VeriLook Surveillance 2.3 SDK

User's guide

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Version: 2.3.0.0.

Release date: 3/31/2014.

Table of Contents

1 Introduction 1

1.1 Supported Platforms 1

1.2 System Requirements 2

1.3 Features 2

1.4 Licensing 3

1.4.1 Single Computer License 3

1.4.2 Enterprise Licensing 3

1.4.3 Volume License Manager 3

2 What's new 5

3 Overview 10

3.1 Required Libraries 10

3.2 Image Support 10

3.2.1 Image 11

3.2.2 Image Format 12

3.2.3 Image File 13

3.2.4 Low-Level Image Input-Output 13

3.3 Matching Threshold and FAR/FRR 13

3.4 Recommendations and Constraints 15

3.5 Cameras 16

3.5.1 IP Cameras 17

3.6 Error codes 17

4 Using 18

4.1 Using Devices 18

4.1.1 Using IP Cameras 18

4.1.1.1 Cisco Video Surveillance IP 4500 18

4.1.1.2 Mobotix DualNight M12 19

4.1.1.3 Prosilica GigE Vision 19

4.1.2 NdmMedia plugin 19

4.1.2.1 Configuring on Microsoft Windows 20

4.1.2.2 Configuring on Linux and Mac OS X 21

Copyright © 2009-2014 Neurotechnology. All rights reserved.

4.1.3 Adding custom device to NDeviceManager 21
4.1.3.1 Naming requirements 22
4.1.3.2 Device manager plugin lifecycle 22
4.1.3.3 Preventing race conditions 23
4.1.3.4 Inteface reference 23
Accesing NDeviceManger 23
IsDriverLoaded 24
AddDevice 24
LoseDevice 24
RemoveDevice 25
Functionality expected by NDeviceManger 25
GetSupportedDeviceTypes 27
UpdateDeviceList 27
GetDeviceParent 27
General functions 28
GetDeviceType 28
GetDeviceIdA 28
GetDeviceIdW 28
GetDeviceDisplayNameA 29
GetDeviceDisplayNameW 29
GetDeviceMakeA 30
GetDeviceMakeW 30
GetDeviceModelA 30
GetDeviceModelW 31
GetDeviceSerialNumberA 31
GetDeviceSerialNumberW 31
Functions for capture devices 32
StartCaptureDeviceCapturing 32
StopCaptureDeviceCapturing 32
IsCaptureDeviceCapturing 33
GetCaptureDeviceMediaType 33
GetCaptureDeviceFormats 34
GetCaptureDeviceCurrentFormat 34
SetCaptureDeviceCurrentFormat 35
Functions for cameras 35
GetCameraFrame 35
Functions for microphones 36
GetMicrophoneSoundSample 36
Functions for biometric devices 36
GetBiometricDeviceBiometricType 36
GetBiometricDeviceVendorId 37

Copyright © 2009-2014 Neurotechnology. All rights reserved.

GetBiometricDeviceProductId 37
IsBiometricDeviceSpoofDetectionSupported 37
GetBiometricDeviceSpoofDetection 38
SetBiometricDeviceSpoofDetection 38
CancelBiometricDevice 39
StartBiometricDeviceSequence 39
EndBiometricDeviceSequence 39
Functions for fingerprint/palm scanners 40
GetFScannerSupportedImpressionTypes 40
GetFScannerSupportedPositions 40
CaptureFScanner 41
Functions for iris scanners 42
GetIrisScannerSupportedPositions 42
CaptureIrisScanner 43
DeviceAdded 44
DeviceLost 44
DeviceRemoved 45
CanUse 45

4.2 Faces 45
4.2.1 Obsolete scores 45
4.2.2 Face Feature Points 46

5 Tutorials 48

6 Samples 49

6.1 C++ (WxWidgets) 49
6.1.1 wxWidgets Compilation 54

6.2 Devices 55
6.2.1 NdmSample - Adding support of custom device to NDeviceManager 55

6.3 .NET 56

7 API Reference 60

7.1 C Reference 60
7.1.1 NBiometrics Library 60
7.1.1.1 NBiometricTypes Unit 61
Functions 64
NBiometricAngleFromRadians Function 65
NBiometricAngleToRadians Function 65

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBiometricAngleToString Function 65
NBiometricQualityIsValid Function 66
NBiometricQualityToString Function 66
NBiometricStatusIsFinal Function 66
NBiometricStatusIsValid Function 67
NBiometricSubtypeIsValid Function 67
NBiometricTypeIsValid Function 67
NEPositionGetAvailableParts Function 67
NEPositionIsKnown Function 68
NEPositionIsLeft Function 68
NEPositionIsOneOf Function 69
NEPositionIsRight Function 69
NEPositionIsRightAndLeft Function 69
NEPositionIsSingleEye Function 70
NEPositionIsTheEye Function 70
NEPositionIsTwoEyes Function 70
NEPositionIsValid Function 70
NEPositionIsValidTheEye Function 71
NFCoreToString Function 71
NFDeltaToString Function 71
NFDoubleCoreToString Function 72
NFImpressionTypeIsCompatibleWith Function 72
NFImpressionTypeIsContact Function 73
NFImpressionTypeIsContactless Function 73
NFImpressionTypeIsFinger Function 73
NFImpressionTypeIsGeneric Function 74
NFImpressionTypeIsLatent Function 74
NFImpressionTypeIsLiveScan Function 74
NFImpressionTypeIsNonliveScan Function 74
NFImpressionTypeIsNonOptical Function 75
NFImpressionTypeIsOneOf Function 75
NFImpressionTypeIsOptical Function 76
NFImpressionTypeIsPalm Function 76
NFImpressionTypeIsPlain Function 76
NFImpressionTypeIsRolled Function 76
NFImpressionTypeIsSwipe Function 77
NFImpressionTypeIsValid Function 77
NFImpressionTypeIsValidFinger Function 77
NFImpressionTypeIsValidPalm Function 78
NFImpressionTypeToFinger Function 78
NFMinutiaNeighborToString Function 78

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFMinutiaToString Function 79
NFPositionGetAvailableParts Function 79
NFPositionIsCompatibleWith Function 80
NFPositionIsCompatibleWithImpressionType Function 80
NFPositionIsFinger Function 80
NFPositionIsFourFingers Function 81
NFPositionIsKnown Function 81
NFPositionIsLeft Function 81
NFPositionIsLeftAndRight Function 81
NFPositionIsOneOf Function 82
NFPositionIsPalm Function 82
NFPositionIsRight Function 83
NFPositionIsSingleFinger Function 83
NFPositionIsTheFinger Function 83
NFPositionIsThePalm Function 83
NFPositionIsThreeFingers Function 84
NFPositionIsTwoFingers Function 84
NFPositionIsValid Function 84
NFPositionIsValidFinger Function 85
NFPositionIsValidPalm Function 85
NFPositionIsValidTheFinger Function 85
NLFeaturePointToString Function 86
Structs, Records, Enums 86
NLFeaturePoint_ Structure 86
NBiometricStatus Enumeration 86
NBiometricSubtype Enumeration 88
NBiometricType Enumeration 89
NEPosition Enumeration 91
NFImpressionType Enumeration 91
NFPosition Enumeration 92
NGender Enumeration 95
NLExpression Enumeration 95
NLProperties Enumeration 96
Types 97
NLFeaturePoint Type 97
Macros 97
N_BIOMETRIC_QUALITY_FAILED Macro 97
N_BIOMETRIC_QUALITY_MAX Macro 97
N_BIOMETRIC_QUALITY_MIN Macro 97
N_BIOMETRIC_QUALITY_UNKNOWN Macro 98
N_PHRASE_ID_UNKNOWN Macro 98

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBiometricAngleToString Macro 98
NBiometricQualityToString Macro 98
NFCoreToString Macro 98
NFDeltaToString Macro 99
NFDoubleCoreToString Macro 99
NFMinutiaNeighborToString Macro 99
NFMinutiaToString Macro 99
NLFeaturePointToString Macro 99
Files 100
NBiometricTypes.h 100
7.1.1.2 NExtractors Unit 103
Files 103
NExtractors.h 103
7.1.1.3 NLAttributes Unit 103
Functions 104
NLAttributesCreate Function 104
Files 104
NLAttributes.h 104
7.1.1.4 NLExtractor Unit 104
Functions 107
NleCompressEx Function 107
NleDetectFace Function 108
NleDetectFaces Function 108
NleDetectFacialFeatures Function 109
NleDetectionDetailsClear Function 109
NleExtract Function 110
NleGeneralizeEx Function 110
NleExtractEndEx Function 111
NleExtractNextEx Function 111
NleCreate Function 112
NleExtractNextEx2 Function 112
NleExtractStart Function 113
NleExtractStartEx Function 113
HNLExtractor 114
NleExtractUsingDetails Function 114
Structs, Records, Enums 114
NleRotation Structure 115
NleTemplateSize Enumeration 115
NleDetectionDetails_ Structure 115
NleEyes Structure 117
NleExtractionStatus Enumeration 117

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NleFace Structure 118
Types 118
NleDetectionDetails Type 118
Macros 119
NLE_FEATURE_POINT_COUNT Macro 120
NleExtractEnd Macro 121
NLEP_BLINK_CONFIDENCE_THRESHOLD Macro 121
NLEP_DARK_GLASSES_CONFIDENCE_THRESHOLD Macro 121
NLEP_DETECT_ALL_FEATURE_POINTS Macro 121
NLEP_DETECT_BASE_FEATURE_POINTS Macro 121
NLEP_FACE_CONFIDENCE_THRESHOLD Macro 122
NLEP_DETECT_BLINK Macro 122
NLEP_DETECT_DARK_GLASSES Macro 122
NLEP_DETECT_EXPRESSION Macro 122
NLEP_DETECT_GENDER Macro 122
NLEP_FACE_QUALITY_THRESHOLD Macro 122
NLEP_DETECT_GLASSES Macro 122
NLEP_DETECT_MORE_FACES Macro 123
NLEP_FAVOR_LARGEST_FACE Macro 123
NLEP_DETECT_MOUTH_OPEN Macro 123
NLEP_EXPRESSION_CONFIDENCE_THRESHOLD Macro 123
NLEP_FACE_DETECTION_SCALE_COUNT Macro 123
NLEP_LIVENESS_THRESHOLD Macro 123
NLEP_FACE_ROLL_ANGLE_BASE Macro 123
NLEP_GENDER_CONFIDENCE_THRESHOLD Macro 124
NLEP_MAX_IOD Macro 124
NLEP_GLASSES_CONFIDENCE_THRESHOLD Macro 124
NLEP_MAX_RECORDS_PER_TEMPLATE Macro 124
NLEP_MAX_ROLL_ANGLE_DEVIATION Macro 124
NLEP_MAX_STREAM_DURATION_IN_FRAMES Macro 124
NLEP_MIN_IOD Macro 124
NLEP_MAX_YAW_ANGLE_DEVIATION Macro 125
NLEP_MOUTH_OPEN_CONFIDENCE_THRESHOLD Macro 125
NLEP_USE_LIVENESS_CHECK Macro 125
NLEP_TEMPLATE_SIZE Macro 125
Files 125
NLExtractor.h 125
NLExtractorParams.h 127
7.1.1.5 NLMatcher Unit 128
Functions 129
NlmCreate Function 129

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NlmIdentifyEnd Function 130
NlmIdentifyNext Function 130
NlmIdentifyStart Function 131
NlmVerify Function 131
HNLMatcher 132
Structs, Records, Enums 132
NlmSpeed Enumeration 132
Macros 133
NLMP_MATCHING_SPEED Macro 133
NLMP_MATCHING_THRESHOLD Macro 133
Files 133
NLMatcher.h 133
NLMatcherParams.h 134
7.1.1.6 NLMatchingDetails Unit 134
Files 134
NLMatchingDetails.h 134
7.1.1.7 NLRecord Unit 134
Functions 135
NLRecordCheck Function 135
NLRecordCreate Function 136
NLRecordCreateFromMemory Function 136
NLRecordGetCbeffProductType Function 137
NLRecordGetCbeffProductTypeMem Function 137
NLRecordGetQuality Function 138
NLRecordGetQualityMem Function 138
NLRecordSetCbeffProductType Function 139
NLRecordGetSizeMem Function 139
NLRecordSetQuality Function 140
HNLRecord 140
Macros 140
NLRecordClone Macro 140
NLRecordGetSize Macro 141
NLRecordSaveToMemory Macro 141
Files 142
NLRecord.h 142
7.1.1.8 NlsMatcher Unit 143
Functions 144
NlsmCreate Function 144
NlsmIdentifyEnd Function 144
NlsmIdentifyNext Function 145
NlsmIdentifyStart Function 145

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NlsmMatchDetailsDeserialize Function 146
NlsmMatchDetailsDeserializeFromStream Function 147
NlsmMatchDetailsFree Function 147
NlsmMatchDetailsSerialize Function 147
NlsmMatchDetailsSerializeToStream Function 148
NlsmVerify Function 148
HNlsMatcher 149
Structs, Records, Enums 149
NlsmFaceMatchDetails Structure 149
NlsmMatchDetails Structure 150
Macros 150
NLSM_PART_NLM Macro 151
NLSM_PART_NONE Macro 151
NLSMP_MATCHING_THRESHOLD Macro 151
Files 151
NlsMatcher.h 151
NlsMatcherParams.h 152
NlsmMatchDetails.h 152
7.1.1.9 NLTemplate Unit 153
Functions 154
NLTemplateAddRecord Function 154
NLTemplateAddRecordFromMemoryEx Function 154
NLTemplateAddRecordCopyEx Function 155
NLTemplateCalculateSize Function 156
NLTemplateAddRecordEx Function 156
NLTemplateCheck Function 157
NLTemplateClearRecords Function 157
NLTemplateCreateFromMemory Function 158
NLTemplateCreateEx Function 159
NLTemplateGetRecord Function 159
NLTemplateGetRecordCapacity Function 160
NLTemplateGetRecordCount Function 160
NLTemplateGetRecordCountMem Function 161
NLTemplatePack Function 161
NLTemplateGetRecordEx Function 162
NLTemplateRemoveRecord Function 163
NLTemplateGetSizeMem Function 163
NLTemplateInsertRecord Function 164
NLTemplateSetRecordCapacity Function 164
HNLTemplate 165
NLTemplateSetRecord Function 165

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLTemplateUnpack Function 165
Macros 166
NLT_MAX_RECORD_COUNT Macro 166
NLT_PROCESS_FIRST_RECORD_ONLY Macro 166
NLTemplateAddRecordCopy Macro 166
NLTemplateAddRecordFromMemory Macro 167
NLTemplateClone Macro 168
NLTemplateCreate Macro 168
NLTemplateGetSize Macro 168
NLTemplateSaveToMemory Macro 169
Files 170
NLTemplate.h 170
7.1.1.10 NMatcher Unit 171
Functions 174
NMCreate Function 174
NMIdentifyNextEx Function 175
NMIdentifyStartEx Function 176
NMIdentifyEnd Function 177
NMIdentifyNext Function 177
NMIdentifyStart Function 178
NMPrepareTemplate Function 179
NMVerifyEx Function 179
NMMatchDetailsDeserialize Function 180
NMMatchDetailsDeserializeFromStream Function 180
NMMatchDetailsFree Function 181
NMVerify Function 181
NMMatchDetailsSerialize Function 182
NMMatchDetailsSerializeToStream Function 183
HNMatcher 183
Structs, Records, Enums 184
NMatchingSpeed Enumeration 184
NMFusionType Enumeration 184
NMMatchDetails Structure 185
Macros 185
NM_FINGERS_MODE_ATMEL_FINGERCHIP Macro 188
NM_FINGERS_MODE_AUTHENTEC_AES2501B Macro 189
NM_FINGERS_MODE_AUTHENTEC_AES4000 Macro 189
NM_FINGERS_MODE_AUTHENTEC_AFS2 Macro 189
NM_FINGERS_MODE_BIOMETRIKA_FX2000 Macro 189
NM_FINGERS_MODE_BIOMETRIKA_FX3000 Macro 189
NM_FINGERS_MODE_BMF_BLP100 Macro 189

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NM_FINGERS_MODE_CROSSMATCH_VERIFIER300 Macro 189
NM_FINGERS_MODE_DIGENT_IZZIX Macro 189
NM_FINGERS_MODE_DIGITALPERSONA_UAREU Macro 189
NM_FINGERS_MODE_ETHENTICA Macro 190
NM_FINGERS_MODE_FUJITSU_MBF200 Macro 190
NM_FINGERS_MODE_FUTRONIC_FS80 Macro 190
NM_FINGERS_MODE_GENERAL Macro 190
NM_FINGERS_MODE_IDENTICATORTECHNOLOGY_DF90 Macro 190
NM_FINGERS_MODE_IDENTIX_DFR2090 Macro 190
NM_FINGERS_MODE_IDENTIX_TOUCHVIEW Macro 190
NM_FINGERS_MODE_KEYTRONIC_SECUREDESKTOP Macro 190
NM_FINGERS_MODE_LIGHTUNING_LTTC500 Macro 190
NM_FINGERS_MODE_NITGEN_FINGKEY_HAMSTER Macro 191
NM_FINGERS_MODE_PRECISEBIOMETRICS_100CS Macro 191
NM_FINGERS_MODE_SECUGEN_HAMSTER Macro 191
NM_FINGERS_MODE_STARTEK_FM200 Macro 191
NM_FINGERS_MODE_TACOMA_CMOS Macro 191
NM_FINGERS_MODE_TESTECH_BIOI Macro 191
NM_FINGERS_MODE_UPEK_TOUCHCHIP Macro 191
NM_PART_FACES Macro 191
NM_PART_FINGERS Macro 192
NM_PART_IRISES Macro 192
NM_PART_NONE Macro 192
NM_PART_PALMS Macro 192
NMP_FACES_MATCHER_BASE Macro 192
NMP_FACES_MATCHING_SPEED Macro 192
NMP_FACES_MATCHING_THRESHOLD Macro 192
NMP_FACES_MATCHING_THRESHOLD_NEW Macro 193
NMP_FINGERS_FILL_FULL_MATCHING_DETAILS Macro 193
NMP_FINGERS_MATCHER_BASE Macro 193
NMP_FINGERS_MATCHING_SPEED Macro 193
NMP_FINGERS_MAXIMAL_ROTATION Macro 193
NMP_FINGERS_MIN_MATCHED_COUNT Macro 193
NMP_FINGERS_MIN_MATCHED_COUNT_THRESHOLD Macro 193
NMP_FINGERS_MODE Macro 194
NMP_FUSION_TYPE Macro 194
NMP_IRISES_MATCHER_BASE Macro 194
NMP_IRISES_MATCHING_SPEED Macro 194
NMP_IRISES_MATCHING_THRESHOLD Macro 194
NMP_IRISES_MAXIMAL_ROTATION Macro 194
NMP_IRISES_MIN_MATCHED_COUNT Macro 194

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMP_IRISES_MIN_MATCHED_COUNT_THRESHOLD Macro 195
NMP_MATCHING_THRESHOLD Macro 195
NMP_PALMS_FILL_FULL_MATCHING_DETAILS Macro 195
NMP_PALMS_MATCHER_BASE Macro 195
NMP_PALMS_MATCHING_SPEED Macro 195
NMP_PALMS_MAXIMAL_ROTATION Macro 195
NMP_PALMS_MIN_MATCHED_COUNT Macro 196
NMP_PALMS_MIN_MATCHED_COUNT_THRESHOLD Macro 196
NMP_VOICES_MATCHER_BASE Macro 196
NMP_VOICES_UNIQUE_PHRASES_ONLY Macro 196
Files 196
NMatcher.h 196
NMatcherParams.h 197
NMMatchDetails.h 199
7.1.1.11 NMatchers Unit 200
Files 200
NMatchers.h 200
7.1.1.12 NMatchingDetails Unit 200
Functions 201
NMatchingDetailsCreateFromMemory Function 201
NMatchingDetailsCreateFromMemoryN Function 202
NMatchingDetailsCreateFromStream Function 202
NMatchingDetailsGetFace Function 203
NMatchingDetailsGetFaceCount Function 203
NMatchingDetailsGetFaceEx Function 204
NMatchingDetailsGetFacesMatchedIndex Function 204
NMatchingDetailsGetFacesScore Function 205
NMatchingDetailsGetFinger Function 206
NMatchingDetailsGetFingerCount Function 206
NMatchingDetailsGetFingerEx Function 207
NMatchingDetailsGetFingersScore Function 207
NMatchingDetailsGetId Function 208
NMatchingDetailsGetIris Function 209
NMatchingDetailsGetIrisCount Function 209
NMatchingDetailsGetIrisesScore Function 210
NMatchingDetailsGetIrisEx Function 211
NMatchingDetailsGetPalm Function 211
NMatchingDetailsGetPalmCount Function 212
NMatchingDetailsGetPalmEx Function 212
NMatchingDetailsGetPalmsScore Function 213
NMatchingDetailsGetVoice Function 214

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMatchingDetailsGetVoiceCount Function 215
NMatchingDetailsGetVoiceEx Function 215
NMatchingDetailsGetVoicesScore Function 215
NMatchingDetailsSaveToMemory Function 217
Macros 217
NMatchingDetailsSaveToStream Macro 217
Files 217
NMatchingDetails.h 218
7.1.1.13 NTemplate Unit 219
Functions 220
NTemplateAddFacesCopyEx Function 221
NTemplateAddFacesEx Function 221
NTemplateAddFacesFromMemoryEx Function 222
NTemplateAddFingersCopyEx Function 222
NTemplateAddFingersEx Function 223
NTemplateAddFingersFromMemoryEx Function 223
NTemplateAddIrisesCopyEx Function 224
NTemplateAddIrisesEx Function 225
NTemplateAddPalmsCopyEx Function 225
NTemplateAddIrisesFromMemoryEx Function 226
NTemplateAddPalmsEx Function 226
NTemplateAddVoicesCopyEx Function 227
NTemplateAddPalmsFromMemoryEx Function 227
NTemplateAddVoices Function 228
NTemplateCheck Function 229
NTemplateClear Function 229
NTemplateAddVoicesFromMemory Function 229
NTemplateCalculateSize Function 230
NTemplateGetFacesEx Function 231
NTemplateCreateEx Function 232
NTemplateCreateFromMemory Function 232
NTemplateGetFingersEx Function 233
NTemplateGetFaces Function 234
NTemplateGetFingers Function 234
NTemplateGetIrisesEx Function 235
NTemplateGetIrises Function 235
NTemplateGetPalms Function 236
NTemplateGetPalmsEx Function 236
NTemplateGetSizeMem Function 237
NTemplateGetVoices Function 237
NTemplatePack Function 238

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTemplateGetVoicesEx Function 239
NTemplateRemoveFaces Function 240
NTemplateRemoveFingers Function 240
NTemplateRemoveIrises Function 241
NTemplateRemovePalms Function 241
NTemplateRemoveVoices Function 242
NTemplateSetFaces Function 242
HNTemplate 243
NTemplateSetFingers Function 243
NTemplateSetIrises Function 243
NTemplateSetPalms Function 243
NTemplateSetVoices Function 244
NTemplateUnpack Function 244
Macros 245
NTemplateAddFaces Macro 245
NTemplateAddFacesCopy Macro 246
NTemplateAddFacesFromMemory Macro 246
NTemplateAddFingers Macro 247
NTemplateAddFingersCopy Macro 247
NTemplateAddFingersFromMemory Macro 248
NTemplateAddIrises Macro 249
NTemplateAddIrisesCopy Macro 249
NTemplateAddIrisesFromMemory Macro 249
NTemplateAddPalms Macro 250
NTemplateAddPalmsCopy Macro 251
NTemplateAddPalmsFromMemory Macro 251
NTemplateAddVoicesCopy Macro 252
NTemplateClone Macro 252
NTemplateCreate Macro 252
NTemplateGetSize Macro 253
NTemplateSaveToMemory Macro 253
Files 254
NTemplate.h 254
7.1.1.14 NTemplates Unit 256
Files 256
NTemplates.h 256
7.1.2 NCore Library 257
7.1.2.1 NArray Unit 259
Functions 261
NArrayCopyTo Function 261
NArrayCopyToBooleanArray Function 262

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NArrayCopyToByteArray Function 262
NArrayCopyToCharArray Function 262
NArrayCopyToDoubleArray Function 263
NArrayCopyToInt16Array Function 263
NArrayCopyToInt32Array Function 264
NArrayCopyToInt64Array Function 264
NArrayCopyToObjectArray Function 264
NArrayCopyToObjectArrayP Function 265
NArrayCopyToP Function 265
NArrayCopyToPointerArray Function 266
NArrayCopyToResultArray Function 266
NArrayCopyToSByteArray Function 267
NArrayCopyToSingleArray Function 267
NArrayCopyToSizeTypeArray Function 267
NArrayCopyToSSizeTypeArray Function 268
NArrayCopyToStringArray Function 268
NArrayCopyToUInt16Array Function 269
NArrayCopyToUInt32Array Function 269
NArrayCopyToUInt64Array Function 269
NArrayCreate Function 270
NArrayCreateCustom Function 270
NArrayCreateFromBooleanArray Function 271
NArrayCreateFromByteArray Function 271
NArrayCreateFromCharArray Function 272
NArrayCreateFromDoubleArray Function 272
NArrayCreateFromInt16Array Function 273
NArrayCreateFromInt32Array Function 273
NArrayCreateFromInt64Array Function 273
NArrayCreateFromObjectArray Function 274
NArrayCreateFromObjectArrayP Function 274
NArrayCreateFromPointerArray Function 275
NArrayCreateFromResultArray Function 275
NArrayCreateFromSByteArray Function 275
NArrayCreateFromSingleArray Function 276
NArrayCreateFromSizeTypeArray Function 276
NArrayCreateFromSSizeTypeArray Function 277
NArrayCreateFromStringArray Function 277
NArrayCreateFromUInt16Array Function 277
NArrayCreateFromUInt32Array Function 278
NArrayCreateFromUInt64Array Function 278
NArrayCreateP Function 279

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NArrayGetLength Function 279
NArrayGetValue Function 279
NArrayGetValueAs Function 280
NArrayGetValueAsBoolean Function 280
NArrayGetValueAsByte Function 281
NArrayGetValueAsChar Function 281
NArrayGetValueAsDouble Function 281
NArrayGetValueAsInt16 Function 282
NArrayGetValueAsInt32 Function 282
NArrayGetValueAsInt64 Function 283
NArrayGetValueAsObject Function 283
NArrayGetValueAsObjectP Function 283
NArrayGetValueAsP Function 284
NArrayGetValueAsPointer Function 284
NArrayGetValueAsResult Function 285
NArrayGetValueAsSByte Function 285
NArrayGetValueAsSingle Function 285
NArrayGetValueAsSizeType Function 286
NArrayGetValueAsSSizeType Function 286
NArrayGetValueAsString Function 287
NArrayGetValueAsUInt16 Function 287
NArrayGetValueAsUInt32 Function 287
NArrayGetValueAsUInt64 Function 288
Macros 288
NArrayCopyToCharArray Macro 288
NArrayCreateFromCharArray Macro 289
NArrayCreateFromStringArray Macro 289
NArrayGetValueAsChar Macro 289
Files 289
NArray.h 289
7.1.2.2 NBinaryReader Unit 291
Functions 293
NBinaryReaderCreate Function 293
NBinaryReaderCreateWithEncoding Function 293
NBinaryReaderGetBaseStream Function 293
NBinaryReaderGetByteOrder Function 294
NBinaryReaderGetNull Function 294
NBinaryReaderGetPosition Function 295
NBinaryReaderPeekChar Function 295
NBinaryReaderRead Function 295
NBinaryReaderReadBoolean Function 296

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBinaryReaderReadByte Function 296
NBinaryReaderReadBytesDst Function 296
NBinaryReaderReadBytesDstN Function 297
NBinaryReaderReadBytesN Function 297
NBinaryReaderReadChar Function 298
NBinaryReaderReadChars Function 298
NBinaryReaderReadCharsDst Function 298
NBinaryReaderReadDouble Function 299
NBinaryReaderReadInt16 Function 299
NBinaryReaderReadInt32 Function 300
NBinaryReaderReadInt64 Function 300
NBinaryReaderReadSByte Function 300
NBinaryReaderReadSingle Function 301
NBinaryReaderReadUInt16 Function 301
NBinaryReaderReadUInt32 Function 301
NBinaryReaderReadUInt64 Function 302
NBinaryReaderSeek Function 302
NBinaryReaderSetPosition Function 302
Macros 303
NBinaryReaderPeekChar Macro 303
NBinaryReaderRead Macro 303
NBinaryReaderReadChar Macro 303
NBinaryReaderReadChars Macro 303
NBinaryReaderReadCharsDst Macro 303
7.1.2.3 NBinaryWriter Unit 303
Functions 304
NBinaryWriterCreate Function 305
NBinaryWriterCreateWithEncoding Function 305
NBinaryWriterFlush Function 305
NBinaryWriterGetBaseStream Function 306
NBinaryWriterGetByteOrder Function 306
NBinaryWriterGetNull Function 306
NBinaryWriterGetPosition Function 307
NBinaryWriterSeek Function 307
NBinaryWriterSetPosition Function 307
NBinaryWriterWriteBoolean Function 308
NBinaryWriterWriteByte Function 308
NBinaryWriterWriteBytes Function 308
NBinaryWriterWriteBytesN Function 309
NBinaryWriterWriteChar Function 309
NBinaryWriterWriteChars Function 309

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBinaryWriterWriteDouble Function 310
NBinaryWriterWriteInt16 Function 310
NBinaryWriterWriteInt32 Function 310
NBinaryWriterWriteInt64 Function 311
NBinaryWriterWriteSByte Function 311
NBinaryWriterWriteSingle Function 311
NBinaryWriterWriteUInt16 Function 312
NBinaryWriterWriteUInt32 Function 312
NBinaryWriterWriteUInt64 Function 312
Macros 313
NBinaryWriterWriteChar Macro 313
NBinaryWriterWriteChars Macro 313
7.1.2.4 NBuffer Unit 313
Functions 314
NBufferCopy Function 314
NBufferCopyFromPtr Function 314
NBufferCopyTo Function 315
NBufferCopyToPtr Function 315
NBufferCopyToPtrRange Function 315
NBufferCreate Function 316
NBufferCreateFromBuffer Function 316
NBufferCreateFromPtr Function 317
NBufferCreateFromPtrEx Function 317
NBufferCreateFromPtrWithFree Function 317
NBufferCreateFromPtrWithFreeProc Function 318
NBufferGetEmpty Function 318
NBufferGetPtr Function 319
NBufferGetSize Function 319
NBufferToPtr Function 319
Macros 320
NBufferCreateFromConstPtr Macro 320
7.1.2.5 NBufferedStream Unit 320
Functions 320
NBufferedStreamCreate Function 320
NBufferedStreamCreateWithBufferSize Function 321
7.1.2.6 NCallback Unit 321
Functions 322
NCallbackClone Function 322
NCallbackCreateCustomRaw Function 322
NCallbackCreateRaw Function 323
NCallbackEquals Function 323

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCallbackFree Function 324
NCallbackGetHashCode Function 324
NCallbackGetParam Function 324
NCallbackGetProcRaw Function 325
NCallbackSet Function 325
Macros 325
NCallbackCreate Macro 325
NCallbackCreateCustom Macro 326
NCallbackGetProc Macro 327
NCallbackInvoke0 Macro 327
NCallbackInvoke1 Macro 327
NCallbackInvoke2 Macro 328
NCallbackInvoke3 Macro 328
NCallbackInvoke4 Macro 328
NCallbackInvoke5 Macro 329
NCallbackInvoke6 Macro 329
NCallbackInvoke7 Macro 330
NCallbackInvoke8 Macro 330
NCallbackInvoke9 Macro 331
NCallbackInvokeRaw Macro 331
Files 332
NCallback.h 332
7.1.2.7 NConsole Unit 333
Functions 334
NConsoleGetError Function 335
NConsoleGetIn Function 335
NConsoleGetOut Function 335
NConsoleReadChar Function 336
NConsoleReadLine Function 336
NConsoleSetError Function 336
NConsoleSetIn Function 336
NConsoleSetOut Function 337
NConsoleWrite Function 337
NConsoleWriteBoolean Function 337
NConsoleWriteBooleanLine Function 338
NConsoleWriteChar Function 338
NConsoleWriteCharLine Function 338
NConsoleWriteChars Function 339
NConsoleWriteCharsLine Function 339
NConsoleWriteDouble Function 339
NConsoleWriteDoubleLine Function 340

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NConsoleWriteEmptyLine Function 340
NConsoleWriteFormat Function 340
NConsoleWriteFormatLine Function 341
NConsoleWriteFormatLineVA Function 341
NConsoleWriteFormatVA Function 341
NConsoleWriteInt32 Function 342
NConsoleWriteInt32Line Function 342
NConsoleWriteInt64 Function 342
NConsoleWriteInt64Line Function 343
NConsoleWriteLine Function 343
NConsoleWriteObject Function 343
NConsoleWriteObjectLine Function 343
NConsoleWritePointer Function 344
NConsoleWritePointerLine Function 344
NConsoleWriteResult Function 344
NConsoleWriteResultLine Function 345
NConsoleWriteSingle Function 345
NConsoleWriteSingleLine Function 345
NConsoleWriteSizeType Function 346
NConsoleWriteSizeTypeLine Function 346
NConsoleWriteSSizeType Function 346
NConsoleWriteSSizeTypeLine Function 347
NConsoleWriteUInt32 Function 347
NConsoleWriteUInt32Line Function 347
NConsoleWriteUInt64 Function 347
NConsoleWriteUInt64Line Function 348
NConsoleWriteValue Function 348
NConsoleWriteValueLine Function 348
NConsoleWriteValueLineP Function 349
NConsoleWriteValueP Function 349
Macros 349
NConsoleReadChar Macro 350
NConsoleWrite Macro 350
NConsoleWriteChar Macro 350
NConsoleWriteCharLine Macro 350
NConsoleWriteChars Macro 350
NConsoleWriteCharsLine Macro 350
NConsoleWriteFormat Macro 351
NConsoleWriteFormatLine Macro 351
NConsoleWriteFormatLineVA Macro 351
NConsoleWriteFormatVA Macro 351

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NConsoleWriteLine Macro 351
7.1.2.8 NCore Unit 352
Functions 352
NCoreAddErrorSuppressedCallback Function 352
NCoreGetInfo Function 353
NCoreAddErrorSuppressed Function 353
NCoreGetCallStack Function 353
NCoreOnExitEx Function 354
NCoreOnStart Function 354
NCoreOnThreadExit Function 354
NCoreOnThreadStart Function 354
NCoreRemoveErrorSuppressedCallback Function 355
NCoreRemoveErrorSuppressed Function 355
NCoreModuleOf 355
Macros 356
NCoreOnExit Macro 356
NCoreOnThreadExitEx Macro 356
Files 356
NCore.h 356
7.1.2.9 NCustomParameterDescriptor Unit 357
Functions 357
NCustomParameterDescriptorCreate Function 357
NCustomParameterDescriptorGetData Function 358
Files 358
NCustomParameterDescriptor.h 358
7.1.2.10 NCustomPropertyDescriptor Unit 358
Functions 359
NCustomPropertyDescriptorGetData Function 359
Files 359
NCustomPropertyDescriptor.h 359
7.1.2.11 NCustomStream Unit 359
Functions 359
NCustomStreamCreate Function 360
7.1.2.12 NDateTime Type 361
Functions 362
NDateTimeAddDays Function 362
NDateTimeAddHours Function 363
NDateTimeAddMilliseconds Function 363
NDateTimeAddMinutes Function 363
NDateTimeAddMonths Function 364
NDateTimeAddSeconds Function 364

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDateTimeAddTicks Function 364
NDateTimeAddTimeSpan Function 365
NDateTimeAddYears Function 365
NDateTimeCompare Function 366
NDateTimeCreate Function 366
NDateTimeCreateDate Function 366
NDateTimeCreateFromTicks Function 367
NDateTimeDaysInMonth Function 367
NDateTimeDecode Function 367
NDateTimeGetDate Function 368
NDateTimeGetDay Function 368
NDateTimeGetDayOfWeek Function 369
NDateTimeGetDayOfYear Function 369
NDateTimeGetHour Function 369
NDateTimeGetMillisecond Function 370
NDateTimeGetMinute Function 370
NDateTimeGetMonth Function 370
NDateTimeGetNow Function 371
NDateTimeGetSecond Function 371
NDateTimeGetTicks Function 371
NDateTimeGetTimeOfDay Function 372
NDateTimeGetToday Function 372
NDateTimeGetUtcNow Function 372
NDateTimeGetYear Function 373
NDateTimeIsDateTimeValid Function 373
NDateTimeIsDateValid Function 373
NDateTimeIsLeapYear Function 374
NDateTimeIsValid Function 374
NDateTimeSubtract Function 374
NDateTimeSubtractTimeSpan Function 375
NDateTimeToLocalTime Function 375
NDateTimeToString Function 375
NDateTimeToUniversalTime Function 376
Structs, Records, Enums 376
NDayOfWeek Enumeration 376
Types 377
NDateTime_ Type 377
Macros 377
N_DATE_TIME_MAX Macro 377
N_DATE_TIME_MIN Macro 377
NDateTimeToString Macro 377

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.13 NDefs Unit 378
Macros 379
N_64 Macro 379
N_ALIGN Macro 379
N_ANSI_C Macro 379
N_API Macro 379
N_API_PTR_RET Macro 379
N_BIG_ENDIAN Macro 380
N_CALL_CONV Macro 380
N_CALLBACK Macro 380
N_CPP Macro 380
N_DEBUG Macro 380
N_DEPRECATED Macro 380
N_FAST_FLOAT Macro 381
N_GCC Macro 381
N_LIB Macro 381
N_LIKELY Macro 381
N_LINUX Macro 381
N_MAC Macro 381
N_MSVC Macro 382
N_NO_ANSI_FUNC Macro 382
N_NO_INLINE Macro 382
N_NO_INT_64 Macro 382
N_NO_RETURN Macro 382
N_NO_THROW Macro 383
N_NO_UNICODE Macro 383
N_PACKED Macro 383
N_RETURN_ADDRESS Macro 383
N_STATIC_ASSERT Macro 383
N_TRACE Macro 383
N_UNICODE Macro 384
N_UNLIKELY Macro 384
N_WINDOWS Macro 384
7.1.2.14 NDescriptor Unit 384
Functions 384
NDescriptorGetAttributes Function 384
NDescriptorGetName Function 385
Files 385
NDescriptor.h 385
7.1.2.15 NDirectory Unit 385
Functions 386

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDirectoryExists Function 386
Macros 386
NDirectoryExists Macro 386
7.1.2.16 NEnum Unit 386
Functions 387
NEnumParseN Function 387
NEnumParseP Function 387
NEnumToStringN Function 387
NEnumToStringP Function 388
NEnumTryParseN Function 388
NEnumTryParseP Function 389
7.1.2.17 NEncoding Enumeration 389
Functions 391
NEncodingConvert Function 391
NEncodingDetect Function 391
NEncodingGetBytes Function 392
NEncodingGetBytesForChars Function 392
NEncodingGetChars Function 393
NEncodingGetMaxByteCount Function 393
NEncodingGetMaxCharCount Function 393
NEncodingGetMinBytesPerChar Function 394
NEncodingGetPreamble Function 394
NEncodingGetPreambleLength Function 395
NEncodingGetStringN Function 395
NEncodingIsSingleByte Function 395
NEncodingIsValid Function 396
Structs, Records, Enums 396
NEncoding_ Enumeration 396
Macros 397
N_ACHAR_ENCODING Macro 397
N_CHAR_ENCODING Macro 397
N_WCHAR_ENCODING Macro 397
NEncodingGetBytes Macro 397
NEncodingGetBytesForChars Macro 397
NEncodingGetChars Macro 398
NEncodingGetMaxByteCount Macro 398
NEncodingGetMaxCharCount Macro 398
7.1.2.18 NError Unit 398
Functions 403
NCallStackEntryCopy Function 403
NCallStackEntryCreate Function 403

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCallStackEntryDispose Function 404
NCallStackEntrySet Function 404
NErrorAppend Function 404
NErrorGetCallStackCount Function 405
NErrorGetCallStackEntry Function 405
NErrorGetCallStackFileEx Function 405
NErrorGetCallStackFunctionEx Function 406
NErrorGetCallStackLength Function 406
NErrorGetCallStackLine Function 406
NErrorGetCallStackLineEx Function 407
NErrorGetCallStackN Function 407
NErrorGetCode Function 407
NErrorGetCodeEx Function 408
NErrorGetDefaultMessageEx Function 408
NErrorGetDefaultMessageN Function 408
NErrorGetExternalCallStackEx Function 409
NErrorGetExternalCallStackN Function 409
NErrorGetExternalError Function 410
NErrorGetExternalErrorEx Function 410
NErrorGetInnerError Function 410
NErrorGetInnerErrorEx Function 411
NErrorGetLast Function 411
NErrorGetLastEx Function 411
NErrorGetLastSysError Function 412
NErrorGetLastWin32Error Function 412
NErrorGetMacErrorMessageEx Function 412
NErrorGetMacErrorMessageN Function 412
NErrorGetMachErrorMessageN Function 413
NErrorGetMessageEx Function 413
NErrorGetMessageN Function 413
NErrorGetParamEx Function 414
NErrorGetParamN Function 414
NErrorGetSysErrorMessageEx Function 414
NErrorGetSysErrorMessageN Function 415
NErrorGetWin32ErrorMessageN Function 415
NErrorReport Function 416
NErrorReportEx Function 416
NErrorSetCom Function 416
NErrorSetComMPN Function 417
NErrorSetComN Function 417
NErrorSetComPN Function 417

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NErrorSetLast Function 418
NErrorSetLastEx Function 418
NErrorSetLastMPN Function 419
NErrorSetLastN Function 419
NErrorSetLastPN Function 419
NErrorSetMac Function 420
NErrorSetMach Function 420
NErrorSetMachMPN Function 421
NErrorSetMachN Function 421
NErrorSetMachPN Function 421
NErrorSetMacMPN Function 422
NErrorSetMacN Function 422
NErrorSetMacPN Function 423
NErrorSetSys Function 423
NErrorSetSysMPN Function 423
NErrorSetSysN Function 424
NErrorSetSysPN Function 424
NErrorSetWin32 Function 425
NErrorSetWin32MPN Function 425
NErrorSetWin32N Function 425
NErrorSetWin32PN Function 426
NErrorSuppress Function 426
Structs, Records, Enums 427
NCallStackEntry_ Structure 427
Macros 427
N_E_ARGUMENT Macro 427
N_E_ABANDONED_MUTEX Macro 427
N_E_ARGUMENT_NULL Macro 427
N_E_ARGUMENT_OUT_OF_RANGE Macro 428
N_E_CLR Macro 428
N_E_COM Macro 428
N_E_ARITHMETIC Macro 428
N_E_CORE Macro 428
N_E_BAD_IMAGE_FORMAT Macro 428
N_E_END_OF_STREAM Macro 429
N_E_EXTERNAL Macro 429
N_E_FAILED Macro 429
N_E_FORMAT Macro 429
N_E_CPP Macro 429
N_E_INDEX_OUT_OF_RANGE Macro 430
N_E_DIRECTORY_NOT_FOUND Macro 430

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_INVALID_OPERATION Macro 430
N_E_DLL_NOT_FOUND Macro 430
N_E_IO Macro 430
N_E_DRIVE_NOT_FOUND Macro 430
N_E_NOT_ACTIVATED Macro 431
N_E_NOT_IMPLEMENTED Macro 431
N_E_ENTRY_POINT_NOT_FOUND Macro 431
N_E_NOT_SUPPORTED Macro 431
N_E_NULL_REFERENCE Macro 431
N_E_FILE_FORMAT Macro 432
N_E_OUT_OF_MEMORY Macro 432
N_E_FILE_LOAD Macro 432
N_E_OVERFLOW Macro 432
N_E_FILE_NOT_FOUND Macro 432
N_E_PARAMETER Macro 432
N_E_PARAMETER_READ_ONLY Macro 433
N_E_SYS Macro 433
N_E_INVALID_CAST Macro 433
N_E_WIN32 Macro 433
N_E_INVALID_ENUM_ARGUMENT Macro 433
N_OK Macro 433
N_E_JVM Macro 434
N_E_MAC Macro 434
NFailed Macro 434
NSucceeded Macro 434
N_E_PATH_TOO_LONG Macro 434
N_E_SECURITY Macro 435
N_E_SOCKET Macro 435
N_E_TIMEOUT Macro 435
NCallStackEntryCreate Macro 435
NE_IS_DIRECTORY_ACCESS Macro 435
NE_MERGE_CALL_STACK Macro 435
NE_NO_CALL_STACK Macro 436
NE_PRESERVE_INNER_ERROR Macro 436
NE_SKIP_ONE_FRAME Macro 436
NErrorAppend Macro 436
NErrorGetCallStackFileEx Macro 436
NErrorGetCallStackFunctionEx Macro 436
NErrorGetClrError Macro 437
NErrorGetComError Macro 437
NErrorGetDefaultMessageEx Macro 437

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NErrorGetExternalCallStack Macro 437
NErrorGetExternalCallStackEx Macro 438
NErrorGetMacErrorMessage Macro 438
NErrorGetMacErrorMessageEx Macro 438
NErrorGetMessageEx Macro 438
NErrorGetParam Macro 439
NErrorGetParamEx Macro 439
NErrorGetSysError Macro 439
NErrorGetSysErrorMessageEx Macro 439
NErrorGetWin32Error Macro 440
NErrorSetCom Macro 440
NErrorSetComMPN Macro 440
NErrorSetComPN Macro 440
NErrorSetLast Macro 440
NErrorSetLastMPN Macro 441
NErrorSetLastPN Macro 441
NErrorSetMac Macro 441
NErrorSetMach Macro 441
NErrorSetMachMPN Macro 441
NErrorSetMachPN Macro 442
NErrorSetMacMPN Macro 442
NErrorSetMacPN Macro 442
NErrorSetSys Macro 442
NErrorSetSysMPN Macro 442
NErrorSetSysPN Macro 442
NErrorSetWin32 Macro 443
NErrorSetWin32MPN Macro 443
NErrorSetWin32PN Macro 443
Files 443
NError.h 443
7.1.2.19 NEvent Unit 448
Functions 449
NEventAddN Function 449
NEventAddRaw Function 449
NEventDispose Function 449
NEventGetCallbacks Function 450
NEventInit Function 450
NEventIsEmpty Function 450
NEventReleaseCallbacks Function 451
NEventRemoveN Function 451
NEventRemoveRaw Function 452

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Macros 452
N_EVENT_RAISE_1 Macro 452
N_EVENT_RAISE_2 Macro 452
N_EVENT_RAISE_3 Macro 453
N_EVENT_RAISE_4 Macro 453
N_EVENT_RAISE_5 Macro 454
N_EVENT_RAISE_6 Macro 454
N_EVENT_RAISE_RAW Macro 455
N_EVENT_SIZE Macro 455
NEventAdd Macro 456
NEventRemove Macro 456
Files 456
NEvent.h 457
7.1.2.20 NExpandableObject Unit 457
Functions 457
NExpandableObjectGetProperties Function 458
Files 458
NExpandableObject.h 458
7.1.2.21 NFile Unit 458
Functions 461
NFileAppendAllLines Function 461
NFileAppendAllLinesPN Function 461
NFileAppendAllLinesWithEncoding Function 461
NFileAppendAllLinesWithEncodingPN Function 462
NFileAppendAllText Function 462
NFileAppendAllTextPN Function 462
NFileAppendAllTextWithEncoding Function 463
NFileAppendAllTextWithEncodingPN Function 463
NFileAppendText Function 464
NFileCreate Function 464
NFileCreateText Function 464
NFileCreateWithBufferSize Function 465
NFileDelete Function 465
NFileExists Function 465
NFileGetAttributes Function 466
NFileOpen Function 466
NFileOpenRead Function 466
NFileOpenText Function 467
NFileOpenWithAccess Function 467
NFileOpenWithAccessAndShare Function 468
NFileOpenWrite Function 468

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFileReadAllBytesCN Function 468
NFileReadAllBytesN Function 469
NFileReadAllLinesCN Function 469
NFileReadAllLinesN Function 469
NFileReadAllLinesWithEncodingCN Function 470
NFileReadAllLinesWithEncodingN Function 470
NFileReadAllTextCN Function 471
NFileReadAllTextN Function 471
NFileReadAllTextWithEncodingCN Function 471
NFileReadAllTextWithEncodingN Function 472
NFileWriteAllBytes Function 472
NFileWriteAllBytesCN Function 473
NFileWriteAllBytesPN Function 473
NFileWriteAllLines Function 473
NFileWriteAllLinesPN Function 474
NFileWriteAllLinesWithEncoding Function 474
NFileWriteAllLinesWithEncodingPN Function 475
NFileWriteAllText Function 475
NFileWriteAllTextPN Function 476
NFileWriteAllTextWithEncoding Function 476
NFileWriteAllTextWithEncodingPN Function 476
Structs, Records, Enums 477
NFileAttributes Enumeration 477
Macros 477
NFileAppendAllText Macro 477
NFileAppendAllTextPN Macro 477
NFileAppendAllTextWithEncoding Macro 478
NFileAppendAllTextWithEncodingPN Macro 478
NFileAppendText Macro 478
NFileCreate Macro 478
NFileCreateText Macro 478
NFileCreateWithBufferSize Macro 478
NFileDelete Macro 479
NFileExists Macro 479
NFileGetAttributes Macro 479
NFileOpen Macro 479
NFileOpenRead Macro 479
NFileOpenText Macro 479
NFileOpenWithAccess Macro 480
NFileOpenWithAccessAndShare Macro 480
NFileOpenWrite Macro 480

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFileReadAllBytesCN Macro 480
NFileReadAllLinesCN Macro 480
NFileReadAllLinesWithEncodingCN Macro 481
NFileReadAllTextCN Macro 481
NFileReadAllTextWithEncodingCN Macro 481
NFileWriteAllBytes Macro 481
NFileWriteAllBytesCN Macro 481
NFileWriteAllLines Macro 481
NFileWriteAllLinesPN Macro 482
NFileWriteAllLinesWithEncoding Macro 482
NFileWriteAllLinesWithEncodingPN Macro 482
NFileWriteAllText Macro 482
NFileWriteAllTextPN Macro 482
NFileWriteAllTextWithEncoding Macro 482
NFileWriteAllTextWithEncodingPN Macro 483
7.1.2.22 NFileStream Unit 483
Functions 483
NFileStreamCreate Function 484
NFileStreamCreateFromOSHandle Function 484
NFileStreamCreateFromOSHandleWithBufferSize Function 484
NFileStreamCreateWithAccess Function 485
NFileStreamCreateWithAccessAndShare Function 485
NFileStreamCreateWithAccessShareAndBufferSize Function 486
NFileStreamGetOSHandle Function 486
Macros 487
NFileStreamCreate Macro 487
NFileStreamCreateWithAccess Macro 487
NFileStreamCreateWithAccessAndShare Macro 487
NFileStreamCreateWithAccessShareAndBufferSize Macro 487
7.1.2.23 NInterlocked Unit 488
Functions 488
NInterlockedAdd Function 488
NInterlockedAnd Function 489
NInterlockedCompareExchange Function 489
NInterlockedCompareExchangePointer Function 489
NInterlockedDecrement Function 490
NInterlockedExchange Function 490
NInterlockedExchangeAdd Function 490
NInterlockedExchangePointer Function 491
NInterlockedIncrement Function 491
NInterlockedOr Function 491

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInterlockedXor Function 492
Macros 492
N_SLOW_INTERLOCKED Macro 492
7.1.2.24 NIOTypes Unit 493
Functions 493
NByteOrderGetSystem Function 493
NByteOrderIsReverse Function 494
NByteOrderIsValid Function 494
NFileAccessIsValid Function 494
NFileModeIsValid Function 495
NFileShareIsValid Function 495
NSeekOriginIsValid Function 495
Structs, Records, Enums 495
NByteOrder Enumeration 496
NFileAccess Enumeration 496
NFileMode Enumeration 496
NFileShare Enumeration 497
NSeekOrigin Enumeration 497
Files 498
NIOTypes.h 498
7.1.2.25 NLibraryInfo Unit 498
Structs, Records, Enums 499
NLibraryInfo_ Structure 499
Macros 499
N_LI_ACTIVATED_MAX_LENGTH Macro 499
N_LI_COMPANY_MAX_LENGTH Macro 500
N_LI_COPYRIGHT_MAX_LENGTH Macro 500
N_LI_PRODUCT_MAX_LENGTH Macro 500
N_LI_TITLE_MAX_LENGTH Macro 500
Files 500
NLibraryInfo.h 500
7.1.2.26 NMemberDescriptor Unit 501
Functions 501
NMemberDescriptorGetComponentType Function 501
Files 501
NMemberDescriptor.h 501
7.1.2.27 NMemory Unit 502
Functions 502
NAlignedFree Function 503
NAlignedOffsetAlloc Function 503
NAlignedOffsetCAlloc Function 503

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NAlignedOffsetReAlloc Function 504
NAlloc Function 504
NCAlloc Function 505
NCompare Function 505
NCopy Function 506
NFill Function 506
NFree Function 506
NFreeEx Function 507
NMemoryTypeIsValid Function 507
NMemoryTypeIsValidForFree Function 507
NMove Function 508
NReAlloc Function 508
Structs, Records, Enums 508
NMemoryType Enumeration 509
Macros 509
NClear Macro 509
Files 509
NMemory.h 509
7.1.2.28 NMemoryStream Unit 510
Functions 511
NMemoryStreamCreate Function 511
NMemoryStreamCreateFromBuffer Function 511
NMemoryStreamCreateFromBufferN Function 512
NMemoryStreamCreateWithCapacity Function 512
NMemoryStreamDetachBuffer Function 512
NMemoryStreamGetBuffer Function 513
NMemoryStreamGetCapacity Function 513
NMemoryStreamGetPositionPtr Function 513
NMemoryStreamSetCapacity Function 514
NMemoryStreamSetPositionPtr Function 514
NMemoryStreamToPtr Function 514
NMemoryStreamWriteTo Function 515
Macros 515
NMemoryStreamCreateFromBufferForRead Macro 515
7.1.2.29 NModule Unit 516
Functions 517
NModuleCheckInit Function 517
NModuleCreate Function 517
NModuleGetActivated Function 518
NModuleGetCompany Function 518
NModuleGetCopyright Function 519

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NModuleGetDependences Function 519
NModuleGetName Function 519
NModuleGetOptions Function 520
NModuleGetProduct Function 520
NModuleGetTitle Function 521
NModuleGetVersionBuild Function 521
NModuleGetVersionMajor Function 521
NModuleGetVersionMinor Function 522
NModuleGetVersionRevision Function 522
NModuleSetCompanyN Function 522
NModuleSetCopyrightN Function 523
NModuleSetNameN Function 523
NModuleSetOptions Function 523
NModuleSetProductN Function 524
NModuleSetTitleN Function 524
NModuleSetVersionBuild Function 524
NModuleSetVersionMajor Function 525
NModuleSetVersionMinor Function 525
NModuleSetVersionRevision Function 525
Structs, Records, Enums 526
NModuleOptions Enumeration 526
Macros 526
N_DECLARE_MODULE Macro 526
N_MODULE_OF Macro 526
NModuleGetActivated Macro 527
NModuleGetCompany Macro 527
NModuleGetCopyright Macro 527
NModuleGetName Macro 527
NModuleGetProduct Macro 527
NModuleGetTitle Macro 527
NModuleSetCompany Macro 528
NModuleSetCopyright Macro 528
NModuleSetName Macro 528
NModuleSetProduct Macro 528
NModuleSetTitle Macro 528
7.1.2.30 NModuleInternal Unit 529
Functions 529
NModuleRegister Function 529
NModuleSetActivatedProc Function 529
NModuleSetDependences Function 530
NModuleSetInit Function 530

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NModuleSetLazyInit Function 530
NModuleSetThreadInit Function 531
NModuleSetThreadUninit Function 531
NModuleSetUninit Function 531
NModuleUnregister Function 532
7.1.2.31 NMonitor Unit 532
Functions 532
NMonitorDispose Function 532
NMonitorEnter Function 533
NMonitorExit Function 533
NMonitorInit Function 533
NMonitorTryEnter Function 534
Macros 534
N_MONITOR_SIZE Macro 534
7.1.2.32 NMutex Unit 534
Functions 535
NMutexCreate Function 535
NMutexCreateFromOSHandle Function 535
NMutexGetOSHandle Function 535
NMutexRelease Function 536
7.1.2.33 NNoDeprecate Unit 536
7.1.2.34 NObject Unit 536
Functions 539
NObjectAddDisposed Function 539
NObjectAddDisposedCallback Function 539
NObjectAddOwnerChanged Function 539
NObjectAddOwnerChangedCallback Function 540
NObjectAddPropertyChanged Function 540
NObjectAddPropertyChangedCallback Function 540
NObjectClone Function 541
NObjectCompareTo Function 541
NObjectCopyProperties Function 542
NObjectEquals Function 542
NObjectFree Function 542
NObjectFreeArray Function 543
NObjectFreeElements Function 543
NObjectGet Function 543
NObjectGetArray Function 543
NObjectGetConcurrent Function 544
NObjectGetFlags Function 544
NObjectGetHashCode Function 545

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObjectGetOwner Function 545
NObjectGetOwnerEx Function 545
NObjectGetParameterWithPartEx Function 546
NObjectGetParameterWithPartExN Function 546
NObjectGetPropertyN Function 547
NObjectGetPropertyNN Function 547
NObjectGetPropertyP Function 548
NObjectGetSize Function 548
NObjectGetSizeMany Function 549
NObjectGetType Function 549
NObjectRef Function 549
NObjectRemoveDisposed Function 550
NObjectRemoveDisposedCallback Function 550
NObjectRemoveOwnerChanged Function 550
NObjectRemoveOwnerChangedCallback Function 551
NObjectRemovePropertyChanged Function 551
NObjectRemovePropertyChangedCallback Function 551
NObjectReset Function 552
NObjectResetProperty Function 552
NObjectSaveManyToMemoryDst Function 552
NObjectSaveManyToMemoryDstN Function 553
NObjectSaveManyToMemoryN Function 553
NObjectSaveManyToStream Function 554
NObjectSaveToMemoryDst Function 554
NObjectSaveToMemoryDstN Function 554
NObjectSaveToMemoryN Function 555
NObjectSaveToStream Function 555
NObjectSet Function 556
NObjectSetConcurrent Function 556
NObjectSetFlags Function 556
NObjectSetParameterWithPartEx Function 557
NObjectSetParameterWithPartExN Function 557
NObjectSetPropertyN Function 558
NObjectSetPropertyNN Function 558
NObjectSetPropertyP Function 559
NObjectToString Function 559
NObjectUnref Function 560
NObjectUnrefArray Function 560
NObjectUnrefElements Function 561
Macros 561
N_OBJECT_REF_RET Macro 561

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObjectCopyParameters Macro 561
NObjectGetParameter Macro 562
NObjectGetParameterEx Macro 562
NObjectGetParameterWithPart Macro 563
NObjectGetPropertyP Macro 563
NObjectResetProperty Macro 564
NObjectSetParameter Macro 564
NObjectSetParameterEx Macro 564
NObjectSetParameterWithPart Macro 565
NObjectSetPropertyP Macro 566
NObjectToString Macro 566
Files 566
NObject.h 566
7.1.2.35 NParameterBag Unit 569
Functions 569
NParameterBagApplyPropertyBag Function 569
NParameterBagContains Function 570
NParameterBagCreate Function 570
NParameterBagGet Function 571
NParameterBagGetAt Function 571
NParameterBagGetCount Function 571
NParameterBagGetKey Function 572
NParameterBagGetValue Function 572
NParameterBagSet Function 572
NParameterBagSetValue Function 573
NParameterBagToPropertyBag Function 573
NParameterBagTryGet Function 573
Macros 574
NParameterBagContains Macro 574
NParameterBagGet Macro 574
NParameterBagSet Macro 574
NParameterBagTryGet Macro 574
Files 575
NParameterBag.h 575
7.1.2.36 NParameters Unit 575
Macros 576
N_TYPE_BOOL Macro 576
N_PC_TYPE_ID Macro 576
N_TYPE_BYTE Macro 576
N_TYPE_DOUBLE Macro 576
N_TYPE_FLOAT Macro 577

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_TYPE_INT Macro 577
N_TYPE_LONG Macro 577
N_TYPE_SBYTE Macro 577
N_TYPE_SHORT Macro 577
N_TYPE_STRING Macro 578
N_TYPE_UINT Macro 578
N_TYPE_ULONG Macro 578
N_TYPE_USHORT Macro 578
NParameterMakeId Macro 578
Files 579
NParameters.h 579
7.1.2.37 NParameterDescriptor Unit 579
Functions 580
NParameterDescriptorGetDefaultValue Function 580
NParameterDescriptorGetFormat Function 580
NParameterDescriptorGetMaxValue Function 580
NParameterDescriptorGetMinValue Function 581
NParameterDescriptorGetParameterType Function 581
NParameterDescriptorGetStdValue Function 581
NParameterDescriptorGetStdValueCount Function 582
Files 582
NParameterDescriptor.h 582
7.1.2.38 NPath Unit 583
Functions 584
NPathChangeExtension Function 584
NPathChangeExtensionPN Function 585
NPathCheck Function 585
NPathCheckN Function 585
NPathCombine Function 586
NPathCombineP1N Function 586
NPathCombineP2N Function 586
NPathGetDirectoryName Function 587
NPathGetExtension Function 587
NPathGetFileName Function 588
NPathGetFileNameWithoutExtension Function 588
NPathGetTempFileName Function 588
NPathGetTempPath Function 589
NPathHasExtension Function 589
NPathIsRooted Function 589
Macros 590
N_MAX_FILE_NAME Macro 590

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_PATH_ALT_DIRECTORY_SEPARATOR Macro 590
N_PATH_DIRECTORY_SEPARATOR Macro 590
N_PATH_IS_CASE_SENSITIVE Macro 590
N_PATH_PATH_SEPARATOR Macro 590
N_PATH_VOLUME_SEPARATOR Macro 591
NPathChangeExtension Macro 591
NPathChangeExtensionPN Macro 591
NPathCheck Macro 591
NPathCombine Macro 591
NPathCombineP1N Macro 591
NPathCombineP2N Macro 592
NPathGetDirectoryName Macro 592
NPathGetExtension Macro 592
NPathGetFileName Macro 592
NPathGetFileNameWithoutExtension Macro 592
NPathHasExtension Macro 593
NPathIsRooted Macro 593
Files 593
NPath.h 593
7.1.2.39 NPlugin Unit 594
Functions 595
NPluginAddChanged Function 595
NPluginAddChangedCallback Function 596
NPluginAddChanging Function 596
NPluginAddChangingCallback Function 596
NPluginDisable Function 597
NPluginEnable Function 597
NPluginGetError Function 597
NPluginGetErrorCode Function 598
NPluginGetFileName Function 598
NPluginGetLoadTime Function 598
NPluginGetModule Function 599
NPluginGetPlugTime Function 599
NPluginGetSelectedInterfaceVersion Function 600
NPluginGetState Function 600
NPluginPlug Function 600
NPluginRemoveChanged Function 601
NPluginRemoveChangedCallback Function 601
NPluginRemoveChanging Function 601
NPluginRemoveChangingCallback Function 602
NPluginUnplug Function 602

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Structs, Records, Enums 603
NPluginState Enumeration 603
Macros 604
NPluginGetFileName Macro 604
Files 604
NPlugin.h 604
7.1.2.40 NPluginInternal Unit 605
Functions 605
NPluginGetSelectedInterface Function 605
Structs, Records, Enums 606
NPluginInterfaceVersionInfo Structure 606
Macros 606
NPluginInterfaceVersionInfoConst Macro 606
7.1.2.41 NPluginManager Unit 607
Functions 609
NPluginManagerAddDisabledPlugin Function 609
NPluginManagerAddDisabledPluginsChanged Function 609
NPluginManagerAddDisabledPluginsChangedCallback Function 610
NPluginManagerAddDisabledPluginsChanging Function 610
NPluginManagerAddDisabledPluginsChangingCallback Function 611
NPluginManagerAddPlugin Function 611
NPluginManagerAddPluginAdded Function 612
NPluginManagerAddPluginAddedCallback Function 612
NPluginManagerAddPluginFromFile Function 612
NPluginManagerAddPluginsChanged Function 613
NPluginManagerAddPluginsChangedCallback Function 613
NPluginManagerAddPluginsChanging Function 613
NPluginManagerAddPluginsChangingCallback Function 614
NPluginManagerAllowsUnplug Function 614
NPluginManagerClearDisabledPlugins Function 615
NPluginManagerEnsurePluginSearchPerformed Function 615
NPluginManagerGetDisabledPlugin Function 616
NPluginManagerGetDisabledPluginCount Function 616
NPluginManagerGetInstances Function 617
NPluginManagerGetInterfaceType Function 617
NPluginManagerGetInterfaceVersions Function 618
NPluginManagerGetPlugin Function 618
NPluginManagerGetPluginByName Function 619
NPluginManagerGetPluginCount Function 619
NPluginManagerGetPluginSearchPath Function 619
NPluginManagerPlugAll Function 620

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPluginManagerRefresh Function 620
NPluginManagerRemoveDisabledPlugin Function 621
NPluginManagerRemoveDisabledPluginAt Function 621
NPluginManagerRemoveDisabledPluginsChanged Function 621
NPluginManagerRemoveDisabledPluginsChangedCallback Function 622
NPluginManagerRemoveDisabledPluginsChanging Function 622
NPluginManagerRemoveDisabledPluginsChangingCallback Function 622
NPluginManagerRemovePluginAdded Function 623
NPluginManagerRemovePluginAddedCallback Function 623
NPluginManagerRemovePluginsChanged Function 624
NPluginManagerRemovePluginsChangedCallback Function 624
NPluginManagerRemovePluginsChanging Function 624
NPluginManagerRemovePluginsChangingCallback Function 625
NPluginManagerSetPluginSearchPath Function 625
NPluginManagerUnplugAll Function 625
Macros 626
NPluginManagerAddDisabledPlugin Macro 626
NPluginManagerAddPlugin Macro 626
NPluginManagerAddPluginFromFile Macro 626
NPluginManagerGetDisabledPlugin Macro 626
NPluginManagerGetInterfaceType Macro 627
NPluginManagerGetPluginSearchPath Macro 627
NPluginManagerRemoveDisabledPlugin Macro 627
NPluginManagerSetPluginSearchPath Macro 627
Files 627
NPluginManager.h 627
7.1.2.42 NPluginModule Unit 629
Functions 630
NPluginModuleGetIncompatiblePlugins Function 630
NPluginModuleCreate Function 631
NPluginModuleGetInterfaceType Function 631
NPluginModuleGetInterfaceVersions Function 631
NPluginModuleSetIncompatiblePluginsN Function 632
NPluginModuleGetPluginName Function 632
NPluginModuleSetInterfaceTypeN Function 633
NPluginModuleSetPluginNameN Function 633
NPluginModuleSetPriority Function 633
NPluginModuleGetPriority Function 634
HNPluginModule 634
Macros 634
NPluginModuleGetIncompatiblePlugins Macro 634

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPluginModuleGetInterfaceType Macro 634
NPluginModuleGetPluginName Macro 635
NPluginModuleSetIncompatiblePlugins Macro 635
NPluginModuleSetInterfaceType Macro 635
NPluginModuleSetPluginName Macro 635
7.1.2.43 NPluginModuleInternal Unit 635
Functions 635
NPluginModuleSetInterfaceVersions Function 636
NPluginModuleSetPlug Function 636
NPluginModuleSetUnplug Function 636
7.1.2.44 NProcessorInfo Unit 637
Functions 638
NProcessorInfoGetCount Function 638
NProcessorInfoGetModelInfo Function 639
NProcessorInfoGetModelNameEx Function 639
NProcessorInfoGetModelNameN Function 639
NProcessorInfoGetVendor Function 640
NProcessorInfoGetVendorNameEx Function 640
NProcessorInfoGetVendorNameN Function 640
NProcessorInfoIs3DNowSupported Function 641
NProcessorInfoIs3DNowSupportedEx Function 641
NProcessorInfoIsLZCntSupported Function 641
NProcessorInfoIsLZCntSupportedEx Function 642
NProcessorInfoIsMmxSupported Function 642
NProcessorInfoIsMmxSupportedEx Function 642
NProcessorInfoIsNeonSupported Function 642
NProcessorInfoIsPopCntSupported Function 643
NProcessorInfoIsPopCntSupportedEx Function 643
NProcessorInfoIsSse2Supported Function 643
NProcessorInfoIsSse2SupportedEx Function 644
NProcessorInfoIsSse3Supported Function 644
NProcessorInfoIsSse3SupportedEx Function 644
NProcessorInfoIsSse41Supported Function 645
NProcessorInfoIsSse41SupportedEx Function 645
NProcessorInfoIsSse42Supported Function 645
NProcessorInfoIsSse42SupportedEx Function 645
NProcessorInfoIsSse4aSupported Function 646
NProcessorInfoIsSse4aSupportedEx Function 646
NProcessorInfoIsSse5Supported Function 646
NProcessorInfoIsSse5SupportedEx Function 647
NProcessorInfoIsSseSupported Function 647

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NProcessorInfoIsSseSupportedEx Function 647
NProcessorInfoIsSsse3Supported Function 648
NProcessorInfoIsSsse3SupportedEx Function 648
Structs, Records, Enums 648
NProcessorVendor Enumeration 648
Macros 649
NProcessorInfoGetModelNameEx Macro 649
NProcessorInfoGetVendorNameEx Macro 649
Files 649
NProcessorInfo.h 649
7.1.2.45 NReDeprecate Unit 651
7.1.2.46 NRWLock Unit 651
Functions 651
NRWLockDispose Function 651
NRWLockEnterRead Function 652
NRWLockEnterWrite Function 652
NRWLockExitRead Function 652
NRWLockExitWrite Function 652
NRWLockInit Function 653
NRWLockTryEnterRead Function 653
NRWLockTryEnterWrite Function 653
7.1.2.47 NSemaphore Unit 654
Functions 654
NSemaphoreCreate Function 654
NSemaphoreCreateFromOSHandle Function 654
NSemaphoreGetOSHandle Function 655
NSemaphoreRelease Function 655
7.1.2.48 NStopwatch Unit 656
Functions 656
NStopwatchCreate Function 656
NStopwatchGetElapsed Function 656
NStopwatchGetElapsedMilliseconds Function 657
NStopwatchGetElapsedTicks Function 657
NStopwatchGetFrequency Function 657
NStopwatchGetTimestamp Function 658
NStopwatchIsHighResolution Function 658
NStopwatchIsRunning Function 658
NStopwatchReset Function 659
NStopwatchRestart Function 659
NStopwatchStart Function 659
NStopwatchStartNew Function 660

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStopwatchStop Function 660
7.1.2.49 NStream Unit 660
Functions 661
NStreamCanRead Function 661
NStreamCanSeek Function 661
NStreamCanWrite Function 662
NStreamCopyTo Function 662
NStreamCopyToWithBufferSize Function 662
NStreamFlush Function 663
NStreamGetLength Function 663
NStreamGetNull Function 664
NStreamGetPosition Function 664
NStreamRead Function 664
NStreamReadByte Function 665
NStreamReadN Function 665
NStreamSeek Function 665
NStreamSetLength Function 666
NStreamSetPosition Function 666
NStreamSynchronized Function 666
NStreamWrite Function 667
NStreamWriteByte Function 667
NStreamWriteN Function 667
7.1.2.50 NStreamReader Unit 668
Functions 669
NStreamReaderCreate Function 669
NStreamReaderCreateFromFile Function 669
NStreamReaderCreateFromFileWithEncoding Function 669
NStreamReaderCreateFromFileWithEncodingAndBufferSize Function 670
NStreamReaderCreateWithEncoding Function 670
NStreamReaderCreateWithEncodingAndBufferSize Function 670
NStreamReaderGetBaseStream Function 671
NStreamReaderGetCurrentEncoding Function 671
NStreamReaderGetNull Function 672
NStreamReaderIsEndOfStream Function 672
Macros 672
NStreamReaderCreateFromFile Macro 672
NStreamReaderCreateFromFileWithEncoding Macro 672
NStreamReaderCreateFromFileWithEncodingAndBufferSize Macro 673
7.1.2.51 NStreamWriter Unit 673
Functions 674
NStreamWriterCreate Function 674

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStreamWriterCreateFromFile Function 674
NStreamWriterCreateFromFileWithEncoding Function 674
NStreamWriterCreateFromFileWithEncodingAndBufferSize Function 675
NStreamWriterCreateWithEncoding Function 675
NStreamWriterCreateWithEncodingAndBufferSize Function 676
NStreamWriterGetAutoFlush Function 676
NStreamWriterGetBaseStream Function 676
NStreamWriterGetNull Function 677
NStreamWriterSetAutoFlush Function 677
Macros 677
NStreamWriterCreateFromFile Macro 677
NStreamWriterCreateFromFileWithEncoding Macro 678
NStreamWriterCreateFromFileWithEncodingAndBufferSize Macro 678
7.1.2.52 NString Unit 678
Format Specifications 685
Functions 687
NStringAppendStrOrChars Function 687
NStringClone Function 687
NStringCompareRangeNA Function 688
NStringCompareRangeNW Function 688
NStringCompareRangeStrOrCharsA Function 689
NStringCompareRangeStrOrCharsW Function 689
NStringConcatArray Function 690
NStringConcatMany Function 690
NStringConcatManyVA Function 691
NStringConcatN Function 691
NStringConcatStrOrChars Function 692
NStringContainsStrN Function 692
NStringContainsStrOrChars Function 692
NStringCopyToStrOrChars Function 693
NStringCreateFromChar Function 693
NStringCreateFromStrOrChars Function 694
NStringCreateWrapper Function 694
NStringEndsWithN Function 694
NStringEndsWithStrOrCharsA Function 695
NStringEndsWithStrOrCharsW Function 695
NStringEqualsN Function 696
NStringEqualsStrOrCharsA Function 696
NStringEqualsStrOrCharsW Function 696
NStringFormat Function 697
NStringFormatVA Function 697

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringFree Function 698
NStringFreeArray Function 698
NStringFreeElements Function 698
NStringGetBuffer Function 698
NStringGetEmpty Function 699
NStringGetHashCode Function 699
NStringGetLength Function 699
NStringHasEmbeddedNull Function 700
NStringIndexOf Function 700
NStringIndexOfAny Function 701
NStringIndexOfAnyFrom Function 701
NStringIndexOfAnyInRange Function 701
NStringIndexOfFrom Function 702
NStringIndexOfInRange Function 702
NStringIndexOfStrFromN Function 703
NStringIndexOfStrInRangeN Function 703
NStringIndexOfStrN Function 704
NStringIndexOfStrOrChars Function 704
NStringIndexOfStrOrCharsFrom Function 705
NStringIndexOfStrOrCharsInRange Function 705
NStringInsertN Function 706
NStringInsertStrOrChars Function 706
NStringIsA Function 706
NStringIsEmpty Function 707
NStringIsW Function 707
NStringIsWhiteSpace Function 707
NStringJoinArray Function 708
NStringJoinArrayVN Function 708
NStringJoinMany Function 709
NStringJoinManyVA Function 709
NStringJoinManyVAVN Function 709
NStringJoinManyVN Function 710
NStringLastIndexOf Function 710
NStringLastIndexOfAny Function 710
NStringLastIndexOfAnyFrom Function 711
NStringLastIndexOfAnyInRange Function 711
NStringLastIndexOfFrom Function 712
NStringLastIndexOfInRange Function 712
NStringLastIndexOfStrFromN Function 713
NStringLastIndexOfStrInRangeN Function 713
NStringLastIndexOfStrN Function 714

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringLastIndexOfStrOrChars Function 714
NStringLastIndexOfStrOrCharsFrom Function 715
NStringLastIndexOfStrOrCharsInRange Function 715
NStringPadLeft Function 716
NStringPadLeftWith Function 716
NStringPadRight Function 717
NStringPadRightWith Function 717
NStringPrependStrOrChars Function 718
NStringPromoteBuffer Function 718
NStringRemove Function 718
NStringRemoveFrom Function 719
NStringReplace Function 719
NStringReplaceStrOrChars Function 720
NStringSet Function 720
NStringSplitWithCount Function 720
NStringSplitWithStrsAndCount Function 721
NStringSplitWithStrsAndCountN Function 721
NStringStartsWithN Function 722
NStringStartsWithStrOrCharsA Function 722
NStringStartsWithStrOrCharsW Function 723
NStringSubstring Function 723
NStringSubstringFrom Function 724
NStringToLower Function 724
NStringToString Function 724
NStringToStringOld Function 725
NStringToStrOrCharArray Function 725
NStringToStrOrCharArrayRange Function 726
NStringToUpper Function 726
NStringTrim Function 726
NStringTrimAny Function 727
NStringTrimEnd Function 727
NStringTrimEndAny Function 727
NStringTrimStart Function 728
NStringTrimStartAny Function 728
Macros 728
N_NEW_LINE Macro 729
N_STRING_SIZE Macro 729
NStringAppend Macro 729
NStringAppendChars Macro 729
NStringAppendStrOrChars Macro 730
NStringCompare Macro 730

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringCompareChars Macro 730
NStringCompareN Macro 731
NStringCompareRange Macro 731
NStringCompareRangeChars Macro 732
NStringCompareRangeN Macro 732
NStringCompareRangeStrOrChars Macro 732
NStringCompareStrOrChars Macro 732
NStringConcat Macro 733
NStringConcatArray Macro 733
NStringConcatChars Macro 733
NStringConcatMany Macro 734
NStringConcatManyVA Macro 734
NStringConcatStrOrChars Macro 734
NStringContainsChars Macro 734
NStringContainsStr Macro 735
NStringContainsStrOrChars Macro 735
NStringCopyTo Macro 735
NStringCopyToString Macro 736
NStringCopyToStrOrChars Macro 736
NStringCreate Macro 736
NStringCreateFromChar Macro 737
NStringCreateFromChars Macro 737
NStringCreateFromStrOrChars Macro 737
NStringCreateWrapper Macro 738
NStringEndsWith Macro 738
NStringEndsWithChars Macro 738
NStringEndsWithStrOrChars Macro 739
NStringEquals Macro 739
NStringEqualsChars Macro 739
NStringEqualsStrOrChars Macro 740
NStringFormat Macro 740
NStringFormatVA Macro 740
NStringGetBuffer Macro 740
NStringGetChar Macro 740
NStringGetLength Macro 740
NStringIndexOf Macro 741
NStringIndexOfAny Macro 741
NStringIndexOfAnyFrom Macro 741
NStringIndexOfAnyInRange Macro 741
NStringIndexOfChars Macro 741
NStringIndexOfCharsFrom Macro 742

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringIndexOfCharsInRange Macro 742
NStringIndexOfFrom Macro 743
NStringIndexOfInRange Macro 743
NStringIndexOfStr Macro 743
NStringIndexOfStrFrom Macro 744
NStringIndexOfStrFromN Macro 744
NStringIndexOfStrInRange Macro 744
NStringIndexOfStrInRangeN Macro 745
NStringIndexOfStrN Macro 745
NStringIndexOfStrOrChars Macro 745
NStringIndexOfStrOrCharsFrom Macro 745
NStringIndexOfStrOrCharsInRange Macro 745
NStringInsert Macro 745
NStringInsertChars Macro 746
NStringInsertN Macro 746
NStringInsertStrOrChars Macro 746
NStringJoinArray Macro 747
NStringJoinArrayVN Macro 747
NStringJoinMany Macro 747
NStringJoinManyVAVN Macro 747
NStringJoinManyVN Macro 747
NStringLastIndexOf Macro 748
NStringLastIndexOfAny Macro 748
NStringLastIndexOfAnyFrom Macro 748
NStringLastIndexOfAnyInRange Macro 748
NStringLastIndexOfChars Macro 748
NStringLastIndexOfCharsFrom Macro 749
NStringLastIndexOfCharsInRange Macro 749
NStringLastIndexOfFrom Macro 750
NStringLastIndexOfInRange Macro 750
NStringLastIndexOfStr Macro 750
NStringLastIndexOfStrFrom Macro 750
NStringLastIndexOfStrFromN Macro 751
NStringLastIndexOfStrInRange Macro 751
NStringLastIndexOfStrInRangeN Macro 752
NStringLastIndexOfStrN Macro 752
NStringLastIndexOfStrOrChars Macro 752
NStringLastIndexOfStrOrCharsFrom Macro 752
NStringLastIndexOfStrOrCharsInRange Macro 752
NStringPadLeft Macro 752
NStringPadLeftWith Macro 753

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringPadRight Macro 753
NStringPadRightWith Macro 753
NStringPrepend Macro 753
NStringPrependChars Macro 754
NStringPrependStrOrChars Macro 754
NStringPromoteBuffer Macro 754
NStringRemove Macro 754
NStringRemoveFrom Macro 754
NStringReplace Macro 755
NStringReplaceChars Macro 755
NStringReplaceStr Macro 755
NStringReplaceStrOrChars Macro 756
NStringSplit Macro 756
NStringSplitWithCount Macro 756
NStringSplitWithStrs Macro 757
NStringSplitWithStrsAndCount Macro 757
NStringSplitWithStrsN Macro 757
NStringStartsWith Macro 758
NStringStartsWithChars Macro 758
NStringStartsWithStrOrChars Macro 759
NStringSubstring Macro 759
NStringSubstringFrom Macro 759
NStringToCharArray Macro 759
NStringToCharArrayRange Macro 759
NStringToNewString Macro 760
NStringToNewStringRange Macro 760
NStringToString Macro 761
NStringToStringOld Macro 761
NStringToStrOrCharArray Macro 761
NStringToStrOrCharArrayRange Macro 761
NStringTrimAny Macro 761
NStringTrimEndAny Macro 762
NStringTrimStartAny Macro 762
Files 762
NString.h 762
7.1.2.53 NStringBuilder Unit 769
Functions 776
NStringBuilderAppend Function 776
NStringBuilderAppendBoolean Function 776
NStringBuilderAppendByte Function 776
NStringBuilderAppendChar Function 777

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringBuilderAppendChars Function 777
NStringBuilderAppendDouble Function 777
NStringBuilderAppendEmptyLine Function 778
NStringBuilderAppendFormat Function 778
NStringBuilderAppendFormatN Function 778
NStringBuilderAppendFormatVA Function 779
NStringBuilderAppendFormatVAN Function 779
NStringBuilderAppendInt16 Function 780
NStringBuilderAppendInt32 Function 780
NStringBuilderAppendInt64 Function 780
NStringBuilderAppendLine Function 781
NStringBuilderAppendLineN Function 781
NStringBuilderAppendN Function 781
NStringBuilderAppendObject Function 782
NStringBuilderAppendPointer Function 782
NStringBuilderAppendResult Function 782
NStringBuilderAppendSByte Function 783
NStringBuilderAppendSingle Function 783
NStringBuilderAppendSizeType Function 783
NStringBuilderAppendSSizeType Function 784
NStringBuilderAppendUInt16 Function 784
NStringBuilderAppendUInt32 Function 784
NStringBuilderAppendUInt64 Function 785
NStringBuilderAppendValue Function 785
NStringBuilderAppendValueP Function 785
NStringBuilderClear Function 786
NStringBuilderCopyToStrOrChars Function 786
NStringBuilderDetachStringN Function 786
NStringBuilderDetachStrOrChars Function 787
NStringBuilderDispose Function 787
NStringBuilderGetCapacity Function 787
NStringBuilderGetChar Function 788
NStringBuilderGetLength Function 788
NStringBuilderGetStringWrapper Function 788
NStringBuilderGetStrOrCharsBuffer Function 789
NStringBuilderInit Function 789
NStringBuilderInitEx Function 790
NStringBuilderInitWithCapacity Function 790
NStringBuilderInitWithStringN Function 790
NStringBuilderInitWithStrOrChars Function 791
NStringBuilderInsert Function 791

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringBuilderInsertBoolean Function 791
NStringBuilderInsertByte Function 792
NStringBuilderInsertChar Function 792
NStringBuilderInsertChars Function 792
NStringBuilderInsertDouble Function 793
NStringBuilderInsertInt16 Function 793
NStringBuilderInsertInt32 Function 794
NStringBuilderInsertInt64 Function 794
NStringBuilderInsertN Function 794
NStringBuilderInsertObject Function 795
NStringBuilderInsertPointer Function 795
NStringBuilderInsertResult Function 796
NStringBuilderInsertSByte Function 796
NStringBuilderInsertSingle Function 796
NStringBuilderInsertSizeType Function 797
NStringBuilderInsertSSizeType Function 797
NStringBuilderInsertUInt16 Function 797
NStringBuilderInsertUInt32 Function 798
NStringBuilderInsertUInt64 Function 798
NStringBuilderInsertValue Function 799
NStringBuilderInsertValueP Function 799
NStringBuilderRemove Function 799
NStringBuilderReplace Function 800
NStringBuilderReplaceInRange Function 800
NStringBuilderReplaceStringInRangeN Function 801
NStringBuilderReplaceStringN Function 801
NStringBuilderReplaceStrOrChars Function 801
NStringBuilderReplaceStrOrCharsInRange Function 802
NStringBuilderSetCapacity Function 802
NStringBuilderSetChar Function 803
NStringBuilderSetLength Function 803
NStringBuilderToString Function 803
NStringBuilderToStringRange Function 804
Macros 804
N_STRING_BUILDER_SIZE Macro 804
NStringBuilderAppend Macro 804
NStringBuilderAppendBoolean Macro 805
NStringBuilderAppendByte Macro 805
NStringBuilderAppendChar Macro 805
NStringBuilderAppendChars Macro 805
NStringBuilderAppendDouble Macro 805

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringBuilderAppendEmptyLine Macro 805
NStringBuilderAppendFormat Macro 806
NStringBuilderAppendFormatN Macro 806
NStringBuilderAppendFormatVA Macro 806
NStringBuilderAppendFormatVAN Macro 806
NStringBuilderAppendInt16 Macro 806
NStringBuilderAppendInt32 Macro 807
NStringBuilderAppendInt64 Macro 807
NStringBuilderAppendLine Macro 807
NStringBuilderAppendLineN Macro 807
NStringBuilderAppendN Macro 807
NStringBuilderAppendObject Macro 807
NStringBuilderAppendPointer Macro 808
NStringBuilderAppendResult Macro 808
NStringBuilderAppendSByte Macro 808
NStringBuilderAppendSingle Macro 808
NStringBuilderAppendSizeType Macro 808
NStringBuilderAppendSSizeType Macro 808
NStringBuilderAppendUInt16 Macro 809
NStringBuilderAppendUInt32 Macro 809
NStringBuilderAppendUInt64 Macro 809
NStringBuilderAppendValue Macro 809
NStringBuilderAppendValueP Macro 809
NStringBuilderClear Macro 810
NStringBuilderCopyTo Macro 810
NStringBuilderCopyToString Macro 810
NStringBuilderCopyToStrOrChars Macro 811
NStringBuilderDetachCharArray Macro 811
NStringBuilderDetachString Macro 811
NStringBuilderDetachStrOrChars Macro 811
NStringBuilderDispose Macro 812
NStringBuilderGetBuffer Macro 812
NStringBuilderGetCapacity Macro 812
NStringBuilderGetChar Macro 812
NStringBuilderGetCharsBuffer Macro 813
NStringBuilderGetLength Macro 813
NStringBuilderGetStringWrapper Macro 813
NStringBuilderGetStrOrCharsBuffer Macro 813
NStringBuilderInit Macro 813
NStringBuilderInitEx Macro 814
NStringBuilderInitWithCapacity Macro 814

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringBuilderInitWithChars Macro 814
NStringBuilderInitWithString Macro 814
NStringBuilderInitWithStringN Macro 815
NStringBuilderInitWithStrOrChars Macro 815
NStringBuilderInsert Macro 815
NStringBuilderInsertBoolean Macro 815
NStringBuilderInsertByte Macro 815
NStringBuilderInsertChar Macro 816
NStringBuilderInsertChars Macro 816
NStringBuilderInsertDouble Macro 816
NStringBuilderInsertInt16 Macro 816
NStringBuilderInsertInt32 Macro 816
NStringBuilderInsertInt64 Macro 816
NStringBuilderInsertN Macro 817
NStringBuilderInsertObject Macro 817
NStringBuilderInsertPointer Macro 817
NStringBuilderInsertResult Macro 817
NStringBuilderInsertSByte Macro 817
NStringBuilderInsertSingle Macro 817
NStringBuilderInsertSizeType Macro 818
NStringBuilderInsertSSizeType Macro 818
NStringBuilderInsertUInt16 Macro 818
NStringBuilderInsertUInt32 Macro 818
NStringBuilderInsertUInt64 Macro 818
NStringBuilderRemove Macro 819
NStringBuilderReplace Macro 819
NStringBuilderReplaceChars Macro 819
NStringBuilderReplaceCharsInRange Macro 819
NStringBuilderReplaceInRange Macro 820
NStringBuilderReplaceString Macro 820
NStringBuilderReplaceStringInRange Macro 820
NStringBuilderReplaceStringN Macro 821
NStringBuilderReplaceStrOrChars Macro 821
NStringBuilderReplaceStrOrCharsInRange Macro 821
NStringBuilderSetCapacity Macro 821
NStringBuilderSetChar Macro 822
NStringBuilderSetLength Macro 822
NStringBuilderToString Macro 822
NStringBuilderToStringRange Macro 822
7.1.2.54 NStringReader Unit 822
Functions 823

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringReaderCreateFromStrOrChars Function 823
NStringReaderCreateN Function 823
Macros 823
NStringReaderCreate Macro 823
NStringReaderCreateFromChars Macro 824
NStringReaderCreateFromStrOrChars Macro 824
7.1.2.55 NStringWriter Unit 824
Functions 825
NStringWriterCreate Function 825
NStringWriterCreateEx Function 825
NStringWriterCreateWithCapacity Function 825
NStringWriterDetachString Function 826
Files 826
NStringWriter.h 826
7.1.2.56 NSyncEvent Unit 827
Functions 827
NSyncEventCreate Function 827
NSyncEventCreateFromOSHandle Function 828
NSyncEventGetOSHandle Function 828
NSyncEventSet Function 828
Macros 829
NSyncEventReset Macro 829
7.1.2.57 NTextReader Unit 829
Functions 830
NTextReaderGetNull Function 830
NTextReaderPeekChar Function 830
NTextReaderReadChar Function 831
NTextReaderReadChars Function 831
NTextReaderReadLine Function 831
NTextReaderReadToEnd Function 832
NTextReaderSynchronized Function 832
Macros 832
NTextReaderPeekChar Macro 832
NTextReaderReadChar Macro 833
NTextReaderReadChars Macro 833
7.1.2.58 NTextWriter Unit 833
Functions 835
NTextWriterFlush Function 835
NTextWriterGetEncoding Function 835
NTextWriterGetNewLine Function 835
NTextWriterGetNull Function 836

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTextWriterSetNewLine Function 836
NTextWriterSynchronized Function 836
NTextWriterWrite Function 837
NTextWriterWriteBoolean Function 837
NTextWriterWriteBooleanLine Function 837
NTextWriterWriteChar Function 838
NTextWriterWriteCharLine Function 838
NTextWriterWriteChars Function 838
NTextWriterWriteCharsLine Function 839
NTextWriterWriteDouble Function 839
NTextWriterWriteDoubleLine Function 839
NTextWriterWriteEmptyLine Function 840
NTextWriterWriteFormat Function 840
NTextWriterWriteFormatLine Function 840
NTextWriterWriteFormatLineVA Function 841
NTextWriterWriteFormatVA Function 841
NTextWriterWriteInt32 Function 842
NTextWriterWriteInt32Line Function 842
NTextWriterWriteInt64 Function 842
NTextWriterWriteInt64Line Function 843
NTextWriterWriteLine Function 843
NTextWriterWriteObject Function 843
NTextWriterWriteObjectLine Function 844
NTextWriterWritePointer Function 844
NTextWriterWritePointerLine Function 844
NTextWriterWriteSingle Function 845
NTextWriterWriteSingleLine Function 845
NTextWriterWriteSizeType Function 845
NTextWriterWriteSizeTypeLine Function 846
NTextWriterWriteSSizeType Function 846
NTextWriterWriteSSizeTypeLine Function 846
NTextWriterWriteUInt32 Function 847
NTextWriterWriteUInt32Line Function 847
NTextWriterWriteUInt64 Function 847
NTextWriterWriteUInt64Line Function 848
NTextWriterWriteValueLineP Function 848
NTextWriterWriteValueP Function 848
Macros 849
NTextWriterSetNewLine Macro 849
NTextWriterWrite Macro 849
NTextWriterWriteChar Macro 849

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTextWriterWriteCharLine Macro 849
NTextWriterWriteChars Macro 850
NTextWriterWriteCharsLine Macro 850
NTextWriterWriteFormat Macro 850
NTextWriterWriteFormatLine Macro 850
NTextWriterWriteFormatLineVA Macro 850
NTextWriterWriteFormatVA Macro 850
NTextWriterWriteLine Macro 851
7.1.2.59 NThread Unit 851
Functions 851
NThreadCreate Function 851
NThreadCreateFromOSHandle Function 852
NThreadCreateParameterized Function 852
NThreadGetCurrent Function 853
NThreadGetCurrentId Function 853
NThreadGetId Function 853
NThreadGetOSHandle Function 854
NThreadJoin Function 854
NThreadSleep Function 854
Files 854
NThread.h 855
7.1.2.60 NTimeSpan Type 855
Functions 856
NTimeSpanAdd Function 856
NTimeSpanCompare Function 857
NTimeSpanCreate Function 857
NTimeSpanCreateFromDays Function 858
NTimeSpanCreateFromHours Function 858
NTimeSpanCreateFromMilliseconds Function 858
NTimeSpanCreateFromMinutes Function 859
NTimeSpanCreateFromSeconds Function 859
NTimeSpanCreateFromTicks Function 859
NTimeSpanDecode Function 860
NTimeSpanDuration Function 860
NTimeSpanGetDays Function 860
NTimeSpanGetHours Function 861
NTimeSpanGetMilliseconds Function 861
NTimeSpanGetMinutes Function 861
NTimeSpanGetSeconds Function 862
NTimeSpanGetTicks Function 862
NTimeSpanGetTotalDays Function 862

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTimeSpanGetTotalHours Function 863
NTimeSpanGetTotalMilliseconds Function 863
NTimeSpanGetTotalMinutes Function 863
NTimeSpanGetTotalSeconds Function 864
NTimeSpanIsTimeValid Function 864
NTimeSpanIsValid Function 865
NTimeSpanNegate Function 865
NTimeSpanSubtract Function 865
NTimeSpanToString Function 866
Types 866
NTimeSpan_ Type 866
Macros 866
N_TIME_SPAN_MAX Macro 866
N_TIME_SPAN_MIN Macro 866
N_TIME_SPAN_TICKS_PER_DAY Macro 867
N_TIME_SPAN_TICKS_PER_HOUR Macro 867
N_TIME_SPAN_TICKS_PER_MILLISECOND Macro 867
N_TIME_SPAN_TICKS_PER_MINUTE Macro 867
N_TIME_SPAN_TICKS_PER_SECOND Macro 867
NTimeSpanToString Macro 868
7.1.2.61 NTls Unit 868
Functions 868
NTlsCreate Function 868
NTlsCreateForObject Function 868
NTlsFree Function 869
NTlsGetObject Function 869
NTlsGetValue Function 869
NTlsSetObject Function 870
NTlsSetValue Function 870
Variables 870
pValue Variable 870
7.1.2.62 NType Unit 871
Functions 873
NTypeAreValuesEqual Function 873
NTypeCompareValues Function 873
NTypeCopyPropertyValues Function 874
NTypeCopyValue Function 874
NTypeDisposeValue Function 875
NTypeDisposeValues Function 875
NTypeFreeValues Function 875
NTypeGetBaseType Function 876

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTypeGetModule Function 876
NTypeGetNameEx Function 876
NTypeGetNameN Function 877
NTypeGetOwnerType Function 877
NTypeGetParameterEx Function 877
NTypeGetPropertyValueN Function 878
NTypeGetPropertyValueP Function 878
NTypeGetRootType Function 879
NTypeGetRootTypeCode Function 879
NTypeGetTypeCode Function 879
NTypeGetUseInsteadType Function 880
NTypeGetValueHashCode Function 880
NTypeGetValueSize Function 881
NTypeHasOwnerType Function 881
NTypeIdentifierToString Function 881
NTypeIdentifierToStringFN Function 881
NTypeIdentifierToStringVN Function 882
NTypeIsAbstract Function 882
NTypeIsAssignableFrom Function 882
NTypeIsBasic Function 882
NTypeIsCallback Function 883
NTypeIsCloneable Function 883
NTypeIsComparable Function 884
NTypeIsDeprecated Function 884
NTypeIsDisposable Function 884
NTypeIsEnum Function 885
NTypeIsEquatable Function 885
NTypeIsFlagsEnum Function 885
NTypeIsHandle Function 886
NTypeIsInstanceOfType Function 886
NTypeIsInstanceOfTypeP Function 886
NTypeIsMemorySerializable Function 887
NTypeIsObject Function 887
NTypeIsParsable Function 887
NTypeIsPrimitive Function 888
NTypeIsPublic Function 888
NTypeIsSealed Function 888
NTypeIsSerializable Function 889
NTypeIsSignNeutral Function 889
NTypeIsStatic Function 889
NTypeIsStruct Function 890

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTypeIsSubclassOf Function 890
NTypeParseValue Function 890
NTypeParseValueVN Function 891
NTypeReset Function 891
NTypeResetPropertyValue Function 892
NTypeSetParameterEx Function 892
NTypeSetPropertyValueN Function 892
NTypeSetPropertyValueP Function 893
NTypeSetValue Function 893
NTypeTryParseValue Function 894
NTypeTryParseValueVN Function 894
NTypeValueToString Function 895
NTypeValueToStringN Function 895
Structs, Records, Enums 896
NTypeCode Enumeration 896
Macros 897
NTypeCopyParameters Macro 897
NTypeGetName Macro 898
NTypeGetNameEx Macro 898
NTypeGetParameter Macro 898
NTypeGetPropertyValueP Macro 898
NTypeParseValue Macro 899
NTypeParseValueVN Macro 899
NTypeResetPropertyValueP Macro 899
NTypeSetParameter Macro 899
NTypeSetPropertyValueP Macro 900
NTypeTryParseValue Macro 900
NTypeTryParseValueVN Macro 900
NTypeValueToString Macro 900
Files 900
NType.h 900
7.1.2.63 NTypeDescriptor Unit 903
Functions 903
NTypeDescriptorGetDefaultPropertyName Function 903
NTypeDescriptorGetDefaultPropertyNameForType Function 903
NTypeDescriptorGetProperties Function 904
NTypeDescriptorGetPropertiesForType Function 904
Files 904
NTypeDescriptor.h 905
7.1.2.64 NTypes Unit 905
Functions 920

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NACharParseN Function 920
NACharParseStrOrChars Function 921
NACharParseVN Function 921
NACharToChars Function 921
NACharToString Function 922
NACharTryParseN Function 922
NACharTryParseStrOrChars Function 923
NACharTryParseVN Function 923
NBooleanParseN Function 924
NBooleanParseStrOrChars Function 924
NBooleanParseVN Function 924
NBooleanToChars Function 925
NBooleanToString Function 925
NBooleanTryParseN Function 926
NBooleanTryParseStrOrChars Function 926
NBooleanTryParseVN Function 927
NCharFromBinDigit Function 927
NCharFromDigit Function 927
NCharFromHexDigit Function 928
NCharFromOctDigit Function 928
NCharIsAscii Function 928
NCharIsBinDigit Function 929
NCharIsDigit Function 929
NCharIsHexDigit Function 929
NCharIsLetter Function 930
NCharIsLower Function 930
NCharIsOctDigit Function 930
NCharIsUpper Function 930
NCharIsWhiteSpace Function 931
NCharParseN Function 931
NCharParseStrOrChars Function 931
NCharParseVN Function 932
NCharToBinDigit Function 932
NCharToChars Function 933
NCharToDigit Function 933
NCharToHexDigit Function 933
NCharToLower Function 934
NCharToOctDigit Function 934
NCharToString Function 934
NCharToUpper Function 935
NCharTryParseN Function 935

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCharTryParseStrOrChars Function 935
NCharTryParseVN Function 936
NComplexToString Function 936
NDoubleGetNaN Function 937
NDoubleGetNegativeInfinity Function 937
NDoubleGetPositiveInfinity Function 937
NDoubleIsInfinity Function 937
NDoubleIsNaN Function 938
NDoubleIsNegativeInfinity Function 938
NDoubleIsPositiveInfinity Function 938
NDoubleParseN Function 939
NDoubleParseStrOrChars Function 939
NDoubleParseVN Function 939
NDoubleToChars Function 940
NDoubleToString Function 940
NDoubleTryParseN Function 941
NDoubleTryParseStrOrChars Function 941
NDoubleTryParseVN Function 942
NGuidParse Function 942
NGuidParseVN Function 943
NGuidToString Function 943
NGuidTryParse Function 943
NGuidTryParseVN Function 944
NIndexPairToString Function 944
NInt16ParseN Function 945
NInt16ParseStrOrChars Function 945
NInt16ParseVN Function 946
NInt16ToChars Function 946
NInt16ToString Function 946
NInt16TryParseN Function 947
NInt16TryParseStrOrChars Function 947
NInt16TryParseVN Function 948
NInt32ParseN Function 948
NInt32ParseStrOrChars Function 949
NInt32ParseVN Function 949
NInt32ToChars Function 949
NInt32ToString Function 950
NInt32TryParseN Function 950
NInt32TryParseStrOrChars Function 951
NInt32TryParseVN Function 951
NInt64ParseN Function 952

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt64ParseStrOrChars Function 952
NInt64ParseVN Function 952
NInt64ToChars Function 953
NInt64ToString Function 953
NInt64TryParseN Function 954
NInt64TryParseStrOrChars Function 954
NInt64TryParseVN Function 955
NInt8ParseN Function 955
NInt8ParseStrOrChars Function 956
NInt8ParseVN Function 956
NInt8ToChars Function 956
NInt8ToString Function 957
NInt8TryParseN Function 957
NInt8TryParseStrOrChars Function 958
NInt8TryParseVN Function 958
NNameStringPairCopy Function 959
NNameStringPairCreate Function 959
NNameStringPairDispose Function 959
NNameStringPairSet Function 960
NOSFamilyGetCurrent Function 960
NPointerParseN Function 960
NPointerParseStrOrChars Function 961
NPointerParseVN Function 961
NPointerToChars Function 961
NPointerToString Function 962
NPointerTryParseN Function 962
NPointerTryParseStrOrChars Function 963
NPointerTryParseVN Function 963
NRangeToString Function 964
NRationalToString Function 964
NResultParseN Function 964
NResultParseStrOrChars Function 965
NResultParseVN Function 965
NResultToChars Function 966
NResultToString Function 966
NResultTryParseN Function 967
NResultTryParseStrOrChars Function 967
NResultTryParseVN Function 967
NSingleGetNaN Function 968
NSingleGetNegativeInfinity Function 968
NSingleGetPositiveInfinity Function 968

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSingleIsInfinity Function 969
NSingleIsNaN Function 969
NSingleIsNegativeInfinity Function 969
NSingleIsPositiveInfinity Function 970
NSingleParseN Function 970
NSingleParseStrOrChars Function 970
NSingleParseVN Function 971
NSingleToChars Function 971
NSingleToString Function 972
NSingleTryParseN Function 972
NSingleTryParseStrOrChars Function 972
NSingleTryParseVN Function 973
NSizeTypeParseN Function 973
NSizeTypeParseStrOrChars Function 974
NSizeTypeParseVN Function 974
NSizeTypeToChars Function 975
NSizeTypeToString Function 975
NSizeTypeTryParseN Function 975
NSizeTypeTryParseStrOrChars Function 976
NSizeTypeTryParseVN Function 976
NSSizeTypeParseN Function 977
NSSizeTypeParseStrOrChars Function 977
NSSizeTypeParseVN Function 978
NSSizeTypeToChars Function 978
NSSizeTypeToString Function 979
NSSizeTypeTryParseN Function 979
NSSizeTypeTryParseStrOrChars Function 979
NSSizeTypeTryParseVN Function 980
NTypesGetPointer Function 980
NUInt16ParseN Function 981
NUInt16ParseStrOrChars Function 981
NUInt16ParseVN Function 981
NUInt16ToChars Function 982
NUInt16ToString Function 982
NUInt16TryParseN Function 983
NUInt16TryParseStrOrChars Function 983
NUInt16TryParseVN Function 984
NUInt32ParseN Function 984
NUInt32ParseStrOrChars Function 984
NUInt32ParseVN Function 985
NUInt32ToChars Function 985

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt32ToString Function 986
NUInt32TryParseN Function 986
NUInt32TryParseStrOrChars Function 987
NUInt32TryParseVN Function 987
NUInt64ParseN Function 988
NUInt64ParseStrOrChars Function 988
NUInt64ParseVN Function 988
NUInt64ToChars Function 989
NUInt64ToString Function 989
NUInt64TryParseN Function 990
NUInt64TryParseStrOrChars Function 990
NUInt64TryParseVN Function 991
NUInt8ParseN Function 991
NUInt8ParseStrOrChars Function 991
NUInt8ParseVN Function 992
NUInt8ToChars Function 992
NUInt8ToString Function 993
NUInt8TryParseN Function 993
NUInt8TryParseStrOrChars Function 994
NUInt8TryParseVN Function 994
NURationalToString Function 995
NVersionRangeContains Function 995
NVersionRangeContainsRange Function 995
NVersionRangeIntersect Function 996
NVersionRangeIntersectsWith Function 996
NVersionRangeToString Function 996
NVersionToString Function 997
NWCharParseN Function 997
NWCharParseStrOrChars Function 998
NWCharParseVN Function 998
NWCharToChars Function 998
NWCharToString Function 999
NWCharTryParseN Function 999
NWCharTryParseStrOrChars Function 1000
NWCharTryParseVN Function 1000
Structs, Records, Enums 1001
NComplex_ Structure 1001
NIndexPair_ Structure 1001
NNameStringPair_ Structure 1001
NRational_ Structure 1002
NURational_ Structure 1002

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NAttributes Enumeration 1002
NOSFamily Enumeration 1003
Types 1004
NAChar Type 1004
NBool Type 1004
NBoolean Type 1004
NByte Type 1005
NChar Type 1005
NComplex Type 1005
NDouble Type 1005
NFloat Type 1005
NHandle Type 1006
NIndexPair Type 1006
NInt Type 1006
NInt16 Type 1006
NInt32 Type 1006
NInt64 Type 1006
NInt8 Type 1007
NLong Type 1007
NNameStringPair Type 1007
NRational Type 1007
NResult Type 1007
NSByte Type 1008
NShort Type 1008
NSingle Type 1008
NSizeType Type 1008
NSSizeType Type 1009
NUInt Type 1009
NUInt16 Type 1009
NUInt32 Type 1009
NUInt64 Type 1009
NUInt8 Type 1009
NULong Type 1010
NURational Type 1010
NUShort Type 1010
NVersion Type 1010
NVersion_ Type 1010
NVersionRange Type 1011
NVersionRange_ Type 1011
Macros 1011
N_DECLARE_HANDLE Macro 1011

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_AW Macro 1011
N_BOOLEAN_FALSE_STRING Macro 1011
N_BOOLEAN_FALSE_STRINGA Macro 1011
N_BOOLEAN_FALSE_STRINGW Macro 1012
N_BOOLEAN_TRUE_STRING Macro 1012
N_BOOLEAN_TRUE_STRINGA Macro 1012
N_BOOLEAN_TRUE_STRINGW Macro 1012
N_BYTE_MAX Macro 1012
N_BYTE_MIN Macro 1013
N_CALLBACK_AW Macro 1013
N_DECLARE_HANDLE_TYPE Macro 1013
N_DECLARE_OBJECT_TYPE Macro 1013
N_DECLARE_STATIC_OBJECT_TYPE Macro 1013
N_DECLARE_TYPE Macro 1014
N_DECLATE_PRIMITIVE Macro 1014
N_DOUBLE_EPSILON Macro 1014
N_DOUBLE_MAX Macro 1014
N_DOUBLE_MIN Macro 1014
N_EMPTY_STRING Macro 1014
N_EMPTY_STRINGA Macro 1015
N_EMPTY_STRINGW Macro 1015
N_FIELD_AW Macro 1015
N_FLOAT_EPSILON Macro 1015
N_FLOAT_INFINITY_STRING Macro 1015
N_FLOAT_INFINITY_STRINGA Macro 1015
N_FLOAT_INFINITY_STRINGW Macro 1016
N_FLOAT_MAX Macro 1016
N_FLOAT_MIN Macro 1016
N_FLOAT_NAN_STRING Macro 1016
N_FLOAT_NAN_STRINGA Macro 1016
N_FLOAT_NAN_STRINGW Macro 1017
N_FUNC_AW Macro 1017
N_INT_MAX Macro 1017
N_INT_MIN Macro 1017
N_INT16_MAX Macro 1017
N_INT16_MIN Macro 1017
N_INT32_MAX Macro 1018
N_INT32_MIN Macro 1018
N_INT64_MAX Macro 1018
N_INT64_MIN Macro 1018
N_INT8_MAX Macro 1018

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_INT8_MIN Macro 1018
N_JOIN_SYMBOLS Macro 1019
N_JOIN_SYMBOLS_ Macro 1019
N_JOIN_SYMBOLS2 Macro 1019
N_JOIN_SYMBOLS2_ Macro 1019
N_LONG_MAX Macro 1019
N_LONG_MIN Macro 1020
N_MACRO_AW Macro 1020
N_PTR_SIZE Macro 1020
N_SBYTE_MAX Macro 1020
N_SBYTE_MIN Macro 1020
N_SHORT_MAX Macro 1020
N_SHORT_MIN Macro 1021
N_SINGLE_EPSILON Macro 1021
N_SINGLE_MAX Macro 1021
N_SINGLE_MIN Macro 1021
N_SIZE_TYPE_MAX Macro 1021
N_SIZE_TYPE_MIN Macro 1021
N_SSIZE_TYPE_MAX Macro 1022
N_SSIZE_TYPE_MIN Macro 1022
N_STRINGIZE Macro 1022
N_STRINGIZE_ Macro 1022
N_STRINGIZE_A Macro 1022
N_STRINGIZE_W Macro 1023
N_STRINGIZEA Macro 1023
N_STRINGIZEW Macro 1023
N_STRUCT_AW Macro 1023
N_SYMBOL_AW Macro 1023
N_T Macro 1023
N_T_A Macro 1024
N_T_W Macro 1024
N_TA Macro 1024
N_TW Macro 1024
N_TYPE_AW Macro 1024
N_TYPE_OF Macro 1024
N_UINT_MAX Macro 1025
N_UINT_MIN Macro 1025
N_UINT16_MAX Macro 1025
N_UINT16_MIN Macro 1025
N_UINT32_MAX Macro 1025
N_UINT32_MIN Macro 1026

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_UINT64_MAX Macro 1026
N_UINT64_MIN Macro 1026
N_UINT8_MAX Macro 1026
N_UINT8_MIN Macro 1026
N_ULONG_MAX Macro 1026
N_ULONG_MIN Macro 1027
N_UNREFERENCED_PARAMETER Macro 1027
N_UNUSED_VARIABLE Macro 1027
N_USHORT_MAX Macro 1027
N_USHORT_MIN Macro 1027
N_VAR_AW Macro 1027
N_WCHAR_SIZE Macro 1028
NACharParse Macro 1028
NACharParseChars Macro 1028
NACharParseStrOrChars Macro 1029
NACharParseVN Macro 1029
NACharToChars Macro 1029
NACharToString Macro 1029
NACharTryParse Macro 1029
NACharTryParseChars Macro 1030
NACharTryParseStrOrChars Macro 1030
NACharTryParseVN Macro 1030
NBooleanParse Macro 1030
NBooleanParseChars Macro 1031
NBooleanParseStrOrChars Macro 1031
NBooleanParseVN Macro 1031
NBooleanToChars Macro 1032
NBooleanToString Macro 1032
NBooleanTryParse Macro 1032
NBooleanTryParseChars Macro 1032
NBooleanTryParseStrOrChars Macro 1033
NBooleanTryParseVN Macro 1033
NCharFromBinDigit Macro 1033
NCharFromDigit Macro 1033
NCharFromHexDigit Macro 1033
NCharFromOctDigit Macro 1034
NCharIsAscii Macro 1034
NCharIsBinDigit Macro 1034
NCharIsDigit Macro 1034
NCharIsHexDigit Macro 1034
NCharIsLetter Macro 1034

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCharIsLower Macro 1035
NCharIsOctDigit Macro 1035
NCharIsUpper Macro 1035
NCharIsWhiteSpace Macro 1035
NCharParse Macro 1035
NCharParseChars Macro 1036
NCharToBinDigit Macro 1036
NCharToDigit Macro 1036
NCharToHexDigit Macro 1036
NCharToLower Macro 1037
NCharToOctDigit Macro 1037
NCharToUpper Macro 1037
NCharTryParse Macro 1037
NCharTryParseChars Macro 1038
NDoubleParse Macro 1038
NDoubleParseChars Macro 1038
NDoubleParseStrOrChars Macro 1039
NDoubleParseVN Macro 1039
NDoubleToChars Macro 1039
NDoubleToString Macro 1039
NDoubleTryParse Macro 1039
NDoubleTryParseChars Macro 1040
NDoubleTryParseStrOrChars Macro 1040
NDoubleTryParseVN Macro 1040
NFalse Macro 1041
NGuidParse Macro 1041
NGuidParseVN Macro 1041
NGuidToString Macro 1041
NGuidTryParse Macro 1041
NGuidTryParseVN Macro 1041
NIndexPairToString Macro 1042
NInt16Parse Macro 1042
NInt16ParseChars Macro 1042
NInt16ParseStrOrChars Macro 1043
NInt16ParseVN Macro 1043
NInt16ToChars Macro 1043
NInt16ToString Macro 1043
NInt16TryParse Macro 1043
NInt16TryParseChars Macro 1044
NInt16TryParseStrOrChars Macro 1044
NInt16TryParseVN Macro 1044

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt32Parse Macro 1044
NInt32ParseChars Macro 1045
NInt32ParseStrOrChars Macro 1045
NInt32ParseVN Macro 1045
NInt32ToChars Macro 1046
NInt32ToString Macro 1046
NInt32TryParse Macro 1046
NInt32TryParseChars Macro 1046
NInt32TryParseStrOrChars Macro 1047
NInt32TryParseVN Macro 1047
NInt64Parse Macro 1047
NInt64ParseChars Macro 1047
NInt64ParseStrOrChars Macro 1048
NInt64ParseVN Macro 1048
NInt64ToChars Macro 1048
NInt64ToString Macro 1048
NInt64TryParse Macro 1048
NInt64TryParseChars Macro 1049
NInt64TryParseStrOrChars Macro 1049
NInt64TryParseVN Macro 1049
NInt8Parse Macro 1050
NInt8ParseChars Macro 1050
NInt8ParseStrOrChars Macro 1050
NInt8ParseVN Macro 1051
NInt8ToChars Macro 1051
NInt8ToString Macro 1051
NInt8TryParse Macro 1051
NInt8TryParseChars Macro 1051
NInt8TryParseStrOrChars Macro 1052
NInt8TryParseVN Macro 1052
NPointerParse Macro 1052
NPointerParseChars Macro 1053
NPointerParseStrOrChars Macro 1053
NPointerParseVN Macro 1053
NPointerToChars Macro 1053
NPointerToString Macro 1054
NPointerTryParse Macro 1054
NPointerTryParseChars Macro 1054
NPointerTryParseStrOrChars Macro 1055
NPointerTryParseVN Macro 1055
NRangeToString Macro 1055

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NRationalToString Macro 1055
NResultParseVN Macro 1055
NResultToChars Macro 1055
NResultToString Macro 1056
NResultTryParseStrOrChars Macro 1056
NResultTryParseVN Macro 1056
NSingleParse Macro 1056
NSingleParseChars Macro 1056
NSingleParseStrOrChars Macro 1057
NSingleParseVN Macro 1057
NSingleToChars Macro 1057
NSingleToString Macro 1057
NSingleTryParse Macro 1058
NSingleTryParseChars Macro 1058
NSingleTryParseStrOrChars Macro 1058
NSingleTryParseVN Macro 1059
NSizeTypeParse Macro 1059
NSizeTypeParseChars Macro 1059
NSizeTypeParseStrOrChars Macro 1059
NSizeTypeParseVN Macro 1060
NSizeTypeToChars Macro 1060
NSizeTypeToString Macro 1060
NSizeTypeTryParse Macro 1060
NSizeTypeTryParseChars Macro 1061
NSizeTypeTryParseStrOrChars Macro 1061
NSizeTypeTryParseVN Macro 1061
NSSizeTypeParse Macro 1061
NSSizeTypeParseChars Macro 1062
NSSizeTypeParseStrOrChars Macro 1062
NSSizeTypeParseVN Macro 1062
NSSizeTypeToChars Macro 1063
NSSizeTypeToString Macro 1063
NSSizeTypeTryParse Macro 1063
NSSizeTypeTryParseChars Macro 1063
NSSizeTypeTryParseStrOrChars Macro 1064
NSSizeTypeTryParseVN Macro 1064
NTrue Macro 1064
NUInt16Parse Macro 1064
NUInt16ParseChars Macro 1065
NUInt16ParseStrOrChars Macro 1065
NUInt16ParseVN Macro 1065

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt16ToChars Macro 1065
NUInt16ToString Macro 1066
NUInt16TryParse Macro 1066
NUInt16TryParseChars Macro 1066
NUInt16TryParseStrOrChars Macro 1067
NUInt16TryParseVN Macro 1067
NUInt32Parse Macro 1067
NUInt32ParseChars Macro 1067
NUInt32ParseStrOrChars Macro 1068
NUInt32ParseVN Macro 1068
NUInt32ToChars Macro 1068
NUInt32ToString Macro 1068
NUInt32TryParse Macro 1068
NUInt32TryParseChars Macro 1069
NUInt32TryParseStrOrChars Macro 1069
NUInt32TryParseVN Macro 1069
NUInt64Parse Macro 1070
NUInt64ParseChars Macro 1070
NUInt64ParseStrOrChars Macro 1070
NUInt64ParseVN Macro 1071
NUInt64ToChars Macro 1071
NUInt64ToString Macro 1071
NUInt64TryParse Macro 1071
NUInt64TryParseChars Macro 1071
NUInt64TryParseStrOrChars Macro 1072
NUInt64TryParseVN Macro 1072
NUInt8Parse Macro 1072
NUInt8ParseChars Macro 1073
NUInt8ParseStrOrChars Macro 1073
NUInt8ParseVN Macro 1073
NUInt8ToChars Macro 1073
NUInt8ToString Macro 1074
NUInt8TryParse Macro 1074
NUInt8TryParseChars Macro 1074
NUInt8TryParseStrOrChars Macro 1075
NUInt8TryParseVN Macro 1075
NULL Macro 1075
NURationalToString Macro 1075
NVersionRangeToString Macro 1075
NVersionToString Macro 1075
NWCharParse Macro 1076

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NWCharParseChars Macro 1076
NWCharParseStrOrChars Macro 1076
NWCharParseVN Macro 1077
NWCharToChars Macro 1077
NWCharToString Macro 1077
NWCharTryParse Macro 1077
NWCharTryParseChars Macro 1077
NWCharTryParseStrOrChars Macro 1078
NWCharTryParseVN Macro 1078
Files 1078
NTypes.h 1078
7.1.2.65 NValue Unit 1094
Functions 1096
NKeyValuePairCopy Function 1096
NKeyValuePairCreate Function 1096
NKeyValuePairDispose Function 1096
NKeyValuePairSet Function 1097
NNameValuePairCopy Function 1097
NNameValuePairCreate Function 1097
NNameValuePairDispose Function 1098
NNameValuePairSet Function 1098
NValueChangeType Function 1098
NValueChangeTypeP Function 1099
NValueCreate Function 1099
NValueCreateCustom Function 1100
NValueCreateCustomN Function 1100
NValueCreateFromBoolean Function 1101
NValueCreateFromByte Function 1101
NValueCreateFromChar Function 1101
NValueCreateFromDouble Function 1102
NValueCreateFromInt16 Function 1102
NValueCreateFromInt32 Function 1102
NValueCreateFromInt64 Function 1103
NValueCreateFromObject Function 1103
NValueCreateFromObjectP Function 1104
NValueCreateFromPointer Function 1104
NValueCreateFromResult Function 1104
NValueCreateFromSByte Function 1105
NValueCreateFromSingle Function 1105
NValueCreateFromSizeType Function 1105
NValueCreateFromSSizeType Function 1106

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NValueCreateFromString Function 1106
NValueCreateFromUInt16 Function 1106
NValueCreateFromUInt32 Function 1107
NValueCreateFromUInt64 Function 1107
NValueCreateP Function 1108
NValueGetAttributes Function 1108
NValueGetFree Function 1108
NValueGetPtr Function 1109
NValueGetSize Function 1109
NValueGetTypeCode Function 1109
NValueGetValueType Function 1110
NValueSet Function 1110
NValueSetP Function 1110
NValueToBoolean Function 1111
NValueToByte Function 1111
NValueToChar Function 1112
NValueToDouble Function 1112
NValueToInt16 Function 1112
NValueToInt32 Function 1113
NValueToInt64 Function 1113
NValueToObject Function 1114
NValueToObjectP Function 1114
NValueToPointer Function 1114
NValueToResult Function 1115
NValueToSByte Function 1115
NValueToSingle Function 1115
NValueToSizeType Function 1116
NValueToSSizeType Function 1116
NValueToString Function 1117
NValueToUInt16 Function 1117
NValueToUInt32 Function 1117
NValueToUInt64 Function 1118
NValueToValue Function 1118
NValueToValueP Function 1119
Structs, Records, Enums 1119
NKeyValuePair_ Structure 1119
NNameValuePair_ Structure 1119
Macros 1120
NNameValuePairCreate Macro 1120
NValueCreateFromChar Macro 1120
NValueCreateFromString Macro 1120

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NValueToChar Macro 1120
Files 1121
NValue.h 1121
7.1.2.66 NWaitObject Unit 1123
Functions 1123
NWaitObjectWaitFor Function 1123
NWaitObjectWaitForTimed Function 1123
Macros 1124
N_INFINITE Macro 1124
7.1.2.67 NWindows Unit 1124
7.1.3 NDevices Library 1124
7.1.3.1 Camera Unit 1126
Functions 1127
CameraGetCurrentFrame Function 1127
CameraGetDevice Function 1127
CameraGetId Function 1128
CameraGetVideoFormat Function 1128
CameraGetVideoFormats Function 1129
CameraIsCapturing Function 1129
CameraSetVideoFormat Function 1130
CameraStartCapturing Function 1130
CameraStopCapturing Function 1130
HCamera 1131
Structs, Records, Enums 1131
CameraVideoFormat Structure 1131
Macros 1132
CameraCopyParameters Macro 1132
CameraGetParameterA Macro 1132
CameraGetParameterW Macro 1132
CAMERAP_AUTOMATIC_SETTINGS Macro 1133
CAMERAP_EXPOSURE Macro 1133
CAMERAP_EXPOSURE_MAX Macro 1133
CAMERAP_EXPOSURE_MIN Macro 1133
CAMERAP_GAIN Macro 1134
CAMERAP_GAIN_MAX Macro 1134
CAMERAP_GAIN_MIN Macro 1134
CAMERAP_IP_CHANNEL_ID Macro 1134
CAMERAP_IP_CHANNEL_NAME Macro 1135
CAMERAP_IP_PASSWORD Macro 1135
CAMERAP_IP_USERNAME Macro 1135
CAMERAP_MIRROR_HORIZONTAL Macro 1135

Copyright © 2009-2014 Neurotechnology. All rights reserved.

CAMERAP_MIRROR_VERTICAL Macro 1135
CAMERAP_VIDEO_DROP_FRAMES Macro 1135
CAMERAP_VIDEO_FILE_NAME Macro 1136
CameraSetParameterA Macro 1136
CameraSetParameterW Macro 1136
Files 1137
Camera.h 1137
CameraParams.h 1137
7.1.3.2 CameraMan Unit 1138
Functions 1139
CameraManGetCamera Function 1139
CameraManGetCameraById Function 1139
CameraManGetCameraCount Function 1140
CameraManGetCameraFromDevice Function 1140
CameraManGetDeviceManager Function 1140
CameraManInitialize Function 1141
CameraManUninitialize Function 1141
Files 1141
CameraMan.h 1141
7.1.3.3 NBiometricDevice Unit 1142
Functions 1142
NBiometricDeviceCancel Function 1142
NBiometricDeviceEndSequence Function 1142
NBiometricDeviceGetBiometricType Function 1143
NBiometricDeviceGetProductId Function 1143
NBiometricDeviceGetSpoofDetection Function 1144
NBiometricDeviceGetVendorId Function 1144
NBiometricDeviceIsSpoofDetectionSupported Function 1144
NBiometricDeviceSetSpoofDetection Function 1145
NBiometricDeviceStartSequence Function 1145
7.1.3.4 NCamera Unit 1145
Functions 1146
NCameraAddStillCaptured Function 1146
NCameraAddStillCapturedCallback Function 1147
NCameraCaptureStill Function 1147
NCameraFocus Function 1148
NCameraGetCurrentStillFormat Function 1148
NCameraGetFocusRegion Function 1148
NCameraGetFrame Function 1149
NCameraGetStillFormats Function 1149
NCameraIsFocusRegionSupported Function 1149

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCameraIsFocusSupported Function 1150
NCameraIsStillCaptureSupported Function 1150
NCameraRemoveStillCaptured Function 1151
NCameraRemoveStillCapturedCallback Function 1151
NCameraResetFocus Function 1151
NCameraSetCurrentStillFormat Function 1152
NCameraSetFocusRegion Function 1152
Structs, Records, Enums 1152
NCameraStatus Enumeration 1152
Macros 1153
NCameraAddIsCapturingChangedCallback Macro 1153
NCameraIsCapturing Macro 1153
NCameraRemoveIsCapturingChangedCallback Macro 1154
NCameraStartCapturing Macro 1154
NCameraStopCapturing Macro 1154
7.1.3.5 NCaptureDevice Unit 1155
Functions 1155
NCaptureDeviceAddIsCapturingChanged Function 1155
NCaptureDeviceAddIsCapturingChangedCallback Function 1155
NCaptureDeviceGetCurrentFormat Function 1156
NCaptureDeviceGetFormats Function 1156
NCaptureDeviceGetMediaType Function 1157
NCaptureDeviceIsCapturing Function 1157
NCaptureDeviceRemoveIsCapturingChanged Function 1157
NCaptureDeviceRemoveIsCapturingChangedCallback Function 1158
NCaptureDeviceSetCurrentFormat Function 1158
NCaptureDeviceStartCapturing Function 1158
NCaptureDeviceStopCapturing Function 1159
7.1.3.6 NDevice Unit 1159
Functions 1160
NDeviceGetChild Function 1160
NDeviceGetChildCount Function 1160
NDeviceGetDeviceType Function 1161
NDeviceGetDisplayName Function 1161
NDeviceGetId Function 1162
NDeviceGetMake Function 1162
NDeviceGetModel Function 1162
NDeviceGetParent Function 1163
NDeviceGetPlugin Function 1163
NDeviceGetSerialNumber Function 1163
NDeviceIsAvailable Function 1164

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDeviceIsDisconnectable Function 1164
NDeviceIsPrivate Function 1165
Structs, Records, Enums 1165
NDeviceType Enumeration 1165
Macros 1166
NDeviceGetDisplayName Macro 1166
NDeviceGetId Macro 1166
NDeviceGetMake Macro 1166
NDeviceGetModel Macro 1167
NDeviceGetSerialNumber Macro 1167
Files 1167
NDevice.h 1167
7.1.3.7 NDeviceManager Unit 1168
Functions 1169
NDeviceManagerAddDeviceAddedCallback Function 1169
NDeviceManagerAddDeviceRemovedCallback Function 1169
NDeviceManagerAddDevicesChangedCallback Function 1170
NDeviceManagerAddDevicesChangingCallback Function 1170
NDeviceManagerAddDevicesRefreshedCallback Function 1170
NDeviceManagerCreate Function 1171
NDeviceManagerGetAutoUpdate Function 1172
NDeviceManagerGetDevice Function 1172
NDeviceManagerGetDeviceById Function 1172
NDeviceManagerGetDeviceCount Function 1173
NDeviceManagerGetDeviceTypes Function 1173
NDeviceManagerGetPluginManager Function 1174
NDeviceManagerRefresh Function 1174
NDeviceManagerRemoveDeviceAddedCallback Function 1174
NDeviceManagerRemoveDeviceRemovedCallback Function 1175
NDeviceManagerRemoveDevicesChangedCallback Function 1175
NDeviceManagerRemoveDevicesChangingCallback Function 1176
NDeviceManagerRemoveDevicesRefreshedCallback Function 1176
Macros 1176
NDeviceManagerGetDeviceById Macro 1176
NDeviceManagerGetInfoA Macro 1177
NDeviceManagerGetInfoW Macro 1177
Files 1177
NDeviceManager.h 1177
7.1.3.8 NDevices Unit 1178
Functions 1178
NDevicesGetInfo Function 1178

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.4 NLicensing Library 1179
7.1.4.1 NLicense Unit 1179
Functions 1180
NLicenseActivateOnline Function 1180
NLicenseGetInfo Function 1181
NLicenseDeactivateOnline Function 1181
NLicenseDeactivateOnlineWithId Function 1181
NLicenseGetInfoForComponent Function 1182
NLicenseGenerateDeactivationId Function 1183
NLicenseGenerateDeactivationIdForLicense Function 1183
NLicenseGenerateId Function 1184
NLicenseIsComponentActivated Function 1184
NLicenseObtain Function 1185
NLicenseObtainComponents Function 1187
NLicenseRelease Function 1189
NLicenseReleaseComponents Function 1190
Structs, Records, Enums 1190
NLicenseInfo_ Structure 1190
Types 1191
NLicenseInfo Type 1191
Macros 1191
NLicenseActivateOnline Macro 1191
NLicenseDeactivateOnline Macro 1191
NLicenseDeactivateOnlineWithId Macro 1191
NLicenseGenerateDeactivationId Macro 1192
NLicenseGenerateDeactivationIdForLicense Macro 1192
NLicenseGenerateId Macro 1192
NLicenseGetInfoForComponent Macro 1192
NLicenseIsComponentActivated Macro 1192
NLicenseObtainComponents Macro 1193
NLicenseReleaseComponents Macro 1193
Files 1193
NLicense.h 1193
7.1.4.2 NLicensing Unit 1194
Functions 1194
NLicensingGetInfo Function 1195
NLicensingModuleOf 1195
Files 1195
NLicensing.h 1195
7.1.4.3 NLicensingService Unit 1195
7.1.5 NMedia Library 1196

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.1 Bmp Unit 1196
Functions 1197
BmpLoadImageFromFile Function 1197
BmpLoadImageFromMemory Function 1198
BmpLoadImageFromStream Function 1198
BmpSaveImageToFile Function 1199
BmpSaveImageToMemory Function 1199
BmpSaveImageToStream Function 1200
Macros 1200
BmpLoadImageFromHBitmap Macro 1200
BmpSaveImageToHBitmap Macro 1200
Files 1200
Bmp.h 1200
7.1.5.2 IHead Unit 1201
Functions 1201
IHeadLoadImageFromFile Function 1201
IHeadLoadImageFromMemory Function 1202
IHeadLoadImageFromStream Function 1203
IHeadSaveImageToFile Function 1203
IHeadSaveImageToMemory Function 1203
IHeadSaveImageToStream Function 1204
Files 1204
IHead.h 1205
7.1.5.3 Jpeg Unit 1205
Functions 1206
JpegInfoGetQuality Function 1206
JpegInfoIsLossless Function 1206
JpegInfoSetLossless Function 1206
JpegInfoSetQuality Function 1207
JpegLoadImageFromFile Function 1207
JpegLoadImageFromMemory Function 1208
JpegLoadImageFromStream Function 1208
JpegSaveImageToFile Function 1209
JpegSaveImageToMemory Function 1209
JpegSaveImageToStream Function 1210
LosslessJpegSaveImageToFile Function 1210
LosslessJpegSaveImageToMemory Function 1211
LosslessJpegSaveImageToStream Function 1211
Macros 1211
JPEG_DEFAULT_QUALITY Macro 1212
Files 1212

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Jpeg.h 1212
7.1.5.4 Jpeg2K Unit 1212
Functions 1213
Jpeg2KInfoGetProfile Function 1213
Jpeg2KInfoGetRatio Function 1214
Jpeg2KInfoSetProfile Function 1214
Jpeg2KInfoSetRatio Function 1214
Jpeg2KLoadImageFromFile Function 1215
Jpeg2KLoadImageFromMemory Function 1215
Jpeg2KLoadImageFromStream Function 1216
Jpeg2KSaveImageToFile Function 1216
Jpeg2KSaveImageToFileWithProfile Function 1217
Jpeg2KSaveImageToMemory Function 1217
Jpeg2KSaveImageToMemoryWithProfile Function 1218
Jpeg2KSaveImageToStream Function 1219
Jpeg2KSaveImageToStreamWithProfile Function 1219
Structs, Records, Enums 1220
Jpeg2KProfile Enumeration 1220
Macros 1220
JPEG_2K_DEFAULT_RATIO Macro 1221
Files 1221
Jpeg2K.h 1221
7.1.5.5 NGeometry Unit 1222
Structs, Records, Enums 1222
NRect_ Structure 1222
NPoint_ Structure 1223
NSize_ Structure 1223
NInterpolationMode Enumeration 1223
NPointD_ Structure 1224
NPointF_ Structure 1224
NRectD_ Structure 1225
NRectF_ Structure 1225
NSizeD_ Structure 1225
NSizeF_ Structure 1226
Types 1226
NPoint Type 1226
NRect Type 1226
NSize Type 1226
NPointD Type 1227
NPointF Type 1227
NRectD Type 1227

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NRectF Type 1227
NSizeD Type 1227
NSizeF Type 1227
Files 1228
NGeometry.h 1228
7.1.5.6 NGrayscaleImage Unit 1228
Functions 1229
NGrayscaleImageGetMaxValue Function 1229
NGrayscaleImageGetMinValue Function 1229
NGrayscaleImageGetPixel Function 1230
NGrayscaleImageSetMaxValue Function 1230
NGrayscaleImageSetMinValue Function 1231
NGrayscaleImageSetPixel Function 1231
Files 1231
NGrayscaleImage.h 1232
7.1.5.7 NImage Unit 1232
Functions 1235
NImageCopy Function 1235
NImageCopyTo Function 1236
NImageCopyData Function 1236
NImageCopyDataPlanes Function 1237
NImageCopyDataPlanesEx Function 1238
NImageCopyFromData Function 1239
NImageCopyFromDataPart Function 1240
NImageCopyFromDataPlanesPart Function 1241
NImageCopyFromDataPlanesPartEx Function 1242
NImageCreateFromFileEx Function 1242
NImageCopyToData Function 1243
NImageCopyToDataPart Function 1244
NImageCopyToDataPlanesPart Function 1245
NImageCopyToDataPlanesPartEx Function 1246
NImageCreate Function 1247
NImageCreateEx Function 1248
NImageCrop Function 1248
NImageCreateFromData Function 1249
NImageFlipDiagonally Function 1250
NImageCreateFromDataEx Function 1251
NImageCreateFromDataPart Function 1251
NImageCreateFromDataPartEx Function 1252
NImageCreateFromMemory Function 1253
NImageFlipHorizontally Function 1254

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NImageCreateFromDataPlanesPart Function 1254
NImageCreateFromStream Function 1255
NImageFlipVertically Function 1256
NImageCreateFromFileExN Function 1256
NImageCreateFromImageEx Function 1257
NImageGetHeight Function 1257
NImageCreateFromImageEx2 Function 1258
NImageCreateFromImagePartEx Function 1258
NImageGetHorzResolution Function 1259
NImageCreateFromImagePartEx2 Function 1260
NImageCreateWrapperForImagePartEx2 Function 1260
NImageCreateWrapper Function 1261
NImageCreateWrapperEx Function 1262
NImageGetStride Function 1263
NImageCreateWrapperForPart Function 1264
NImageGetVertResolution Function 1265
NImageCreateWrapperForPartEx Function 1266
NImageGetWidth Function 1267
NImageCreateWrapperForPlanesPart Function 1267
NImageRotateFlip Function 1268
NImageExportPalette Function 1268
NImageGetResolutionIsAspectRatio Function 1269
NImageSaveToFileEx Function 1269
NImageSaveToMemory Function 1270
NImageGetImageSize Function 1271
NImageSaveToStream Function 1271
NImageSetHorzResolution Function 1272
NImageGetIndex Function 1272
NImageSetResolutionIsAspectRatio Function 1272
NImageGetInfo Function 1273
NImageSetVertResolution Function 1273
NImageGetMaxValue Function 1273
NImageToGrayscale Function 1274
NImageGetMinValue Function 1274
NImageToMonochrome Function 1274
NImageGetPalette Function 1275
NImageToRgb Function 1275
NImageGetPaletteEntry Function 1275
HNImage 1276
NImageGetPaletteLength Function 1276
NImageGetPalettePtr Function 1276

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NImageGetPaletteSize Function 1276
NImageGetPixel Function 1277
NImageGetPixelFormat Function 1277
NImageGetPixelsN Function 1278
NImageGetPixelsPtr Function 1278
NImageGetPlane Function 1278
NImageGetPlaneCount Function 1279
NImageGetPlanePtr Function 1279
NImageGetPlanePtrs Function 1279
NImageGetPlanes Function 1280
NImageGetPlaneSize Function 1280
NImageSetIndex Function 1281
NImageSetMaxValue Function 1281
NImageSetMinValue Function 1281
NImageSetPalette Function 1282
NImageSetPaletteEntry Function 1282
NImageSetPaletteN Function 1282
NImageSetPixel Function 1283
Structs, Records, Enums 1283
NImageRotateFlipType Enumeration 1283
Macros 1284
NI_ALL_DST Macro 1284
NI_ALL_DST_AND_SRC Macro 1284
NI_ALL_SRC Macro 1284
NI_DST_ALPHA_CHANNEL_FIRST Macro 1284
NI_DST_BOTTOM_TO_TOP Macro 1284
NI_DST_SWAP_CHANNELS Macro 1285
NI_READ_UNKNOWN_IMAGE Macro 1285
NI_SRC_ALPHA_CHANNEL_FIRST Macro 1285
NI_SRC_BOTTOM_TO_TOP Macro 1285
NI_SRC_SWAP_CHANNELS Macro 1285
NI_WRITE_UNKNOWN_IMAGE Macro 1285
NImageClone Macro 1286
NImageCreateFromFileA Macro 1286
NImageCreateFromFileEx Macro 1286
NImageCreateFromFileW Macro 1287
NImageCreateFromImage Macro 1287
NImageCreateFromImagePart Macro 1287
NImageCreateWrapperForImagePart Macro 1288
NImageFree Macro 1289
NImageGetPixels Macro 1289

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NImageSaveToFileA Macro 1289
NImageSaveToFileEx Macro 1290
NImageSaveToFileW Macro 1290
Files 1291
NImage.h 1291
7.1.5.8 NImageFile Unit 1294
Functions 1295
NImageFileClose Function 1295
NImageFileCreate Function 1295
NImageFileGetFormat Function 1296
NImageFileGetFormatEx Function 1296
NImageFileIsOpened Function 1297
NImageFileReadImage Function 1297
HNImageFile 1298
Macros 1298
NImageFileFree Macro 1298
Files 1298
NImageFile.h 1298
7.1.5.9 NImageFormat Unit 1299
Functions 1301
NImageFormatCanRead Function 1301
NImageFormatCanWrite Function 1302
NImageFormatCanWriteMultiple Function 1302
NImageFormatCreateInfo Function 1303
NImageFormatGetBmp Function 1303
NImageFormatGetDefaultFileExtensionEx Function 1303
NImageFormatGetBmpEx Function 1304
NImageFormatGetFormat Function 1304
NImageFormatGetDefaultInternetMediaType Function 1305
NImageFormatGetFileFilterEx Function 1305
NImageFormatGetFormatCount Function 1306
NImageFormatGetFormatEx Function 1306
NImageFormatGetIHead Function 1306
NImageFormatGetJpeg Function 1307
NImageFormatGetIHeadEx Function 1307
NImageFormatGetJpeg2K Function 1308
NImageFormatGetInternetMediaType Function 1308
NImageFormatGetNameEx Function 1309
NImageFormatGetPng Function 1309
NImageFormatGetJpeg2KEx Function 1310
NImageFormatOpenFile Function 1310

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NImageFormatGetJpegEx Function 1311
NImageFormatGetTiff Function 1311
NImageFormatGetWsq Function 1311
NImageFormatGetPngEx Function 1312
NImageFormatLoadImageFromFile Function 1312
NImageFormatSaveImagesToFile Function 1313
NImageFormatGetTiffEx Function 1313
NImageFormatSaveImagesToMemory Function 1314
NImageFormatLoadImageFromMemory Function 1314
NImageFormatGetWsqEx Function 1315
NImageFormatLoadImageFromStream Function 1315
NImageFormatSaveImagesToStream Function 1316
NImageFormatOpenFileFromMemory Function 1317
NImageFormatOpenFileFromStream Function 1317
NImageFormatOpenReaderFromFile Function 1318
NImageFormatOpenReaderFromStream Function 1318
NImageFormatSaveImageToFile Function 1319
NImageFormatOpenWriterToStream Function 1319
NImageFormatOpenReaderFromMemory Function 1319
NImageFormatSaveImageToMemory Function 1320
NImageFormatSaveImageToStream Function 1320
NImageFormatOpenWriterToFile Function 1321
NImageFormatSelect Function 1321
NImageFormatSelectByInternetMediaType Function 1322
NImageFormatSelectEx Function 1322
NImageFormatOpenWriterToMemory Function 1323
NImageFormatSelectReaderFromStream Function 1323
NImageFormatSelectReaderFromFile Function 1324
NImageFormatSelectReaderFromMemory Function 1324
HNImageFormat 1324
Macros 1325
NImageFormatGetDefaultFileExtensionA Macro 1325
NImageFormatGetDefaultFileExtensionEx Macro 1325
NImageFormatGetDefaultFileExtensionW Macro 1325
NImageFormatGetDefaultInternetMediaType Macro 1326
NImageFormatGetFileFilterA Macro 1326
NImageFormatGetFileFilterEx Macro 1326
NImageFormatGetFileFilterW Macro 1326
NImageFormatGetInternetMediaType Macro 1327
NImageFormatGetNameA Macro 1327
NImageFormatGetNameEx Macro 1327

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NImageFormatGetNameW Macro 1327
NImageFormatOpenReaderFromFile Macro 1328
NImageFormatOpenWriterToFile Macro 1328
NImageFormatSelectByInternetMediaType Macro 1328
NImageFormatSelectEx Macro 1328
NImageFormatSelectReaderFromFile Macro 1328
Files 1329
NImageFormat.h 1329
7.1.5.10 NImages Unit 1331
Functions 1331
NImagesGetGrayscaleColorWrapperEx Function 1331
NImagesRecolorImage Function 1332
Macros 1332
NImagesGetInfoA Macro 1332
NImagesGetInfoW Macro 1333
NImagesModuleOf Macro 1333
Files 1333
NImages.h 1333
7.1.5.11 NMediaFormat Unit 1334
Functions 1334
NMediaFormatGetMediaSubtype Function 1334
NMediaFormatGetMediaType Function 1334
NMediaFormatIsCompatibleWith Function 1335
NMediaFormatSetMediaSubtype Function 1335
HNMediaFormat 1336
Files 1336
NMediaFormat.h 1336
7.1.5.12 NMediaReader Unit 1336
Functions 1337
NMediaReaderCreate Function 1337
NMediaReaderCreateFromFile Function 1338
NMediaReaderCreateFromUrl Function 1338
NMediaReaderGetLength Function 1339
NMediaReaderGetPosition Function 1340
NMediaReaderGetSource Function 1340
NMediaReaderGetState Function 1340
NMediaReaderReadAudioSample Function 1341
NMediaReaderIsLive Function 1341
NMediaReaderReadVideoSample Function 1342
NMediaReaderPause Function 1342
NMediaReaderSetPosition Function 1342

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMediaReaderStart Function 1343
NMediaReaderStop Function 1343
HNMediaReader 1343
Macros 1343
NMediaReaderCreateFromFile Macro 1344
NMediaReaderCreateFromUrl Macro 1344
Files 1344
NMediaReader.h 1344
7.1.5.13 NMediaSource Unit 1345
Functions 1346
NMediaSourceCreateFromFile Function 1346
NMediaSourceCreateFromUrl Function 1346
NMediaSourceEnumDevices Function 1347
NMediaSourceGetCurrentFormat Function 1347
NMediaSourceGetDisplayName Function 1348
NMediaSourceGetFormats Function 1348
NMediaSourceGetId Function 1349
NMediaSourceGetMediaType Function 1349
NMediaSourceSetCurrentFormat Function 1349
HNmediaSource 1350
Macros 1350
NMediaSourceCreateFromFile Macro 1350
NMediaSourceCreateFromUrl Macro 1350
NMediaSourceGetDisplayName Macro 1350
NMediaSourceGetId Macro 1350
NMS_ALLOW_DUPLICATE_DEVICES Macro 1351
NMS_DO_NOT_USE_DIRECT_SHOW Macro 1351
NMS_DO_NOT_USE_WINDOWS_MEDIA_FOUNDATION Macro 1351
NMS_PREFER_DIRECT_SHOW Macro 1351
Files 1351
NMediaSource.h 1351
7.1.5.14 NMediaTypes Unit 1352
Functions 1353
NMediaTypeIsValid Function 1353
NMediaTypeIsValidSingle Function 1353
NVideoInterlaceModeIsValid Function 1353
Structs, Records, Enums 1354
NMediaState Enumeration 1354
NMediaType Enumeration 1354
NVideoInterlaceMode Enumeration 1354
Files 1355

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMediaTypes.h 1355
7.1.5.15 NMonochromeImage Unit 1355
Functions 1356
NMonochromeImageGetMaxValue Function 1356
NMonochromeImageGetMinValue Function 1356
NMonochromeImageGetPixel Function 1356
NMonochromeImageSetMaxValue Function 1357
NMonochromeImageSetMinValue Function 1357
NMonochromeImageSetPixel Function 1358
Files 1358
NMonochromeImage.h 1358
7.1.5.16 NPixelFormat Unit 1359
Functions 1363
NPixelFormatCalcRowSize Function 1363
NPixelFormatCreate Function 1364
NPixelFormatGetBitsPerPixel Function 1364
NPixelFormatGetBytesPerPixel Function 1365
NPixelFormatGetIndexed Function 1365
NPixelFormatGetNonIndexed Function 1365
NPixelFormatGetNonSeparated Function 1366
NPixelFormatGetPaletteSize Function 1366
NPixelFormatGetPixelType Function 1366
NPixelFormatGetRowSize Function 1367
NPixelFormatGetSeparated Function 1367
NPixelFormatGetWithExtraChannel Function 1367
NPixelFormatIsValid Function 1368
NPixelFormatToString Function 1368
NPixelTypeCanBeIndexed Function 1368
NPixelTypeIsValid Function 1369
Structs, Records, Enums 1369
NPixelType Enumeration 1369
Types 1369
NPixelFormat Type 1369
NPixelFormat_ Type 1370
Macros 1370
NPF_GRAYSCALE_16S Macro 1370
NPF_GRAYSCALE_16U Macro 1370
NPF_GRAYSCALE_1U Macro 1370
NPF_GRAYSCALE_2U Macro 1370
NPF_GRAYSCALE_32F Macro 1371
NPF_GRAYSCALE_32S Macro 1371

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPF_GRAYSCALE_32U Macro 1371
NPF_GRAYSCALE_4U Macro 1371
NPF_GRAYSCALE_64F Macro 1371
NPF_GRAYSCALE_64S Macro 1372
NPF_GRAYSCALE_64U Macro 1372
NPF_GRAYSCALE_8S Macro 1372
NPF_GRAYSCALE_8U Macro 1372
NPF_GRAYSCALE_A_16S Macro 1372
NPF_GRAYSCALE_A_16U Macro 1373
NPF_GRAYSCALE_A_32F Macro 1373
NPF_GRAYSCALE_A_32S Macro 1373
NPF_GRAYSCALE_A_32U Macro 1373
NPF_GRAYSCALE_A_64F Macro 1373
NPF_GRAYSCALE_A_64S Macro 1374
NPF_GRAYSCALE_A_64U Macro 1374
NPF_GRAYSCALE_A_8S Macro 1374
NPF_GRAYSCALE_A_8U Macro 1374
NPF_GRAYSCALE_PA_16S Macro 1374
NPF_GRAYSCALE_PA_16U Macro 1375
NPF_GRAYSCALE_PA_32F Macro 1375
NPF_GRAYSCALE_PA_32S Macro 1375
NPF_GRAYSCALE_PA_32U Macro 1375
NPF_GRAYSCALE_PA_64F Macro 1375
NPF_GRAYSCALE_PA_64S Macro 1376
NPF_GRAYSCALE_PA_64U Macro 1376
NPF_GRAYSCALE_PA_8S Macro 1376
NPF_GRAYSCALE_PA_8U Macro 1376
NPF_RGB_16S Macro 1376
NPF_RGB_16U Macro 1377
NPF_RGB_32F Macro 1377
NPF_RGB_32S Macro 1377
NPF_RGB_32U Macro 1377
NPF_RGB_64F Macro 1377
NPF_RGB_64S Macro 1378
NPF_RGB_64U Macro 1378
NPF_RGB_8S Macro 1378
NPF_RGB_8U Macro 1378
NPF_RGB_8U_INDEXED_1 Macro 1378
NPF_RGB_8U_INDEXED_2 Macro 1379
NPF_RGB_8U_INDEXED_4 Macro 1379
NPF_RGB_8U_INDEXED_8 Macro 1379

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPF_RGB_A_16S Macro 1379
NPF_RGB_A_16U Macro 1379
NPF_RGB_A_32F Macro 1380
NPF_RGB_A_32S Macro 1380
NPF_RGB_A_32U Macro 1380
NPF_RGB_A_64F Macro 1380
NPF_RGB_A_64S Macro 1380
NPF_RGB_A_64U Macro 1381
NPF_RGB_A_8S Macro 1381
NPF_RGB_A_8U Macro 1381
NPF_RGB_A_8U_INDEXED_1 Macro 1381
NPF_RGB_A_8U_INDEXED_2 Macro 1381
NPF_RGB_A_8U_INDEXED_4 Macro 1382
NPF_RGB_A_8U_INDEXED_8 Macro 1382
NPF_RGB_PA_16S Macro 1382
NPF_RGB_PA_16U Macro 1382
NPF_RGB_PA_32F Macro 1382
NPF_RGB_PA_32S Macro 1383
NPF_RGB_PA_32U Macro 1383
NPF_RGB_PA_64F Macro 1383
NPF_RGB_PA_64S Macro 1383
NPF_RGB_PA_64U Macro 1383
NPF_RGB_PA_8S Macro 1384
NPF_RGB_PA_8U Macro 1384
NPF_UNDEFINED Macro 1384
npfGrayscale Macro 1384
npfMonochrome Macro 1384
npfRgb Macro 1385
NPixelFormatToString Macro 1385
Files 1385
NPixelFormat.h 1385
7.1.5.17 NRgb Unit 1389
Functions 1390
NRgbToString Function 1390
Structs, Records, Enums 1390
NRgb_ Structure 1390
Types 1391
NRgb Type 1391
Macros 1391
NRgbToString Macro 1391
Files 1391

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NRgb.h 1391
7.1.5.18 NRgbImage Unit 1392
Functions 1392
NRgbImageGetMaxValue Function 1392
NRgbImageGetMinValue Function 1392
NRgbImageGetPixel Function 1393
NRgbImageSetMaxValue Function 1393
NRgbImageSetMinValue Function 1394
NRgbImageSetPixel Function 1394
Files 1395
NRgbImage.h 1395
7.1.5.19 NVideoFormat Unit 1395
Functions 1397
NVideoFormatCreate Function 1398
NVideoFormatGetFrameRate Function 1398
NVideoFormatGetHeight Function 1398
NVideoFormatGetInterlaceMode Function 1399
NVideoFormatGetPixelAspectRatio Function 1399
NVideoFormatGetWidth Function 1399
NVideoFormatMediaSubtypeToString Function 1400
NVideoFormatSetFrameRate Function 1400
NVideoFormatSetHeight Function 1401
NVideoFormatSetInterlaceMode Function 1401
NVideoFormatSetPixelAspectRatio Function 1401
NVideoFormatSetWidth Function 1402
Macros 1402
N_VIDEO_FORMAT_AI44 Macro 1402
N_VIDEO_FORMAT_ARGB32 Macro 1402
N_VIDEO_FORMAT_AVC1 Macro 1402
N_VIDEO_FORMAT_AYUV Macro 1402
N_VIDEO_FORMAT_DV25 Macro 1403
N_VIDEO_FORMAT_DV50 Macro 1403
N_VIDEO_FORMAT_DVC Macro 1403
N_VIDEO_FORMAT_DVH1 Macro 1403
N_VIDEO_FORMAT_DVHD Macro 1403
N_VIDEO_FORMAT_DVSD Macro 1404
N_VIDEO_FORMAT_DVSL Macro 1404
N_VIDEO_FORMAT_DX50 Macro 1404
N_VIDEO_FORMAT_H264 Macro 1404
N_VIDEO_FORMAT_I420 Macro 1404
N_VIDEO_FORMAT_IYUV Macro 1404

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_VIDEO_FORMAT_L16 Macro 1405
N_VIDEO_FORMAT_L8 Macro 1405
N_VIDEO_FORMAT_M4S2 Macro 1405
N_VIDEO_FORMAT_MJPG Macro 1405
N_VIDEO_FORMAT_MP43 Macro 1405
N_VIDEO_FORMAT_MP4S Macro 1405
N_VIDEO_FORMAT_MP4V Macro 1406
N_VIDEO_FORMAT_MPG1 Macro 1406
N_VIDEO_FORMAT_MSS1 Macro 1406
N_VIDEO_FORMAT_MSS2 Macro 1406
N_VIDEO_FORMAT_NV11 Macro 1406
N_VIDEO_FORMAT_NV12 Macro 1407
N_VIDEO_FORMAT_P010 Macro 1407
N_VIDEO_FORMAT_P016 Macro 1407
N_VIDEO_FORMAT_P210 Macro 1407
N_VIDEO_FORMAT_P216 Macro 1407
N_VIDEO_FORMAT_RGB24 Macro 1407
N_VIDEO_FORMAT_RGB32 Macro 1408
N_VIDEO_FORMAT_RGB555 Macro 1408
N_VIDEO_FORMAT_RGB565 Macro 1408
N_VIDEO_FORMAT_RGB8 Macro 1408
N_VIDEO_FORMAT_UNKNOWN Macro 1408
N_VIDEO_FORMAT_UYVY Macro 1408
N_VIDEO_FORMAT_V210 Macro 1409
N_VIDEO_FORMAT_V216 Macro 1409
N_VIDEO_FORMAT_V410 Macro 1409
N_VIDEO_FORMAT_WMV1 Macro 1409
N_VIDEO_FORMAT_WMV2 Macro 1409
N_VIDEO_FORMAT_WMV3 Macro 1410
N_VIDEO_FORMAT_WVC1 Macro 1410
N_VIDEO_FORMAT_Y210 Macro 1410
N_VIDEO_FORMAT_Y216 Macro 1410
N_VIDEO_FORMAT_Y410 Macro 1410
N_VIDEO_FORMAT_Y416 Macro 1410
N_VIDEO_FORMAT_Y41P Macro 1411
N_VIDEO_FORMAT_Y41T Macro 1411
N_VIDEO_FORMAT_Y42T Macro 1411
N_VIDEO_FORMAT_YUY2 Macro 1411
N_VIDEO_FORMAT_YV12 Macro 1411
N_VIDEO_FORMAT_YVU9 Macro 1411
N_VIDEO_FORMAT_YVYU Macro 1412

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NVideoFormatMediaSubtypeToString Macro 1412
Files 1412
NVideoFormat.h 1412
7.1.5.20 NVideoReader Unit 1414
Functions 1415
NVideoReaderCreateFromFile Function 1415
NVideoReaderGetFrame Function 1415
NVideoReaderGetFrameCount Function 1416
NVideoReaderGetFrameHeight Function 1416
NVideoReaderGetFrameRate Function 1417
NVideoReaderGetFrameWidth Function 1417
Macros 1418
NVideoReaderCreateFromFile Macro 1418
NVideoReaderFree Macro 1418
Files 1418
NVideoReader.h 1418
7.1.5.21 NVideoWriter Module 1419
Functions 1420
NVideoWriterCreateFile Function 1420
NVideoWriterWriteFrame Function 1420
NVideoWriterOptionsCreateWithGui Function 1421
HNVideoWriter 1421
HNVideoWriterOptions 1421
Macros 1421
NVideoWriterCreateFile Macro 1421
NVideoWriterFree Macro 1422
NVideoWriterOptionsFree Macro 1422
Files 1422
NVideoWriter.h 1422
NVideoWriterOptions.h 1423
7.1.5.22 Png Unit 1423
Functions 1423
PngInfoGetCompressionLevel Function 1424
PngInfoSetCompressionLevel Function 1424
PngLoadImageFromFile Function 1424
PngLoadImageFromMemory Function 1425
PngLoadImageFromStream Function 1425
PngSaveImageToFile Function 1426
PngSaveImageToMemory Function 1426
PngSaveImageToStream Function 1427
Macros 1428

Copyright © 2009-2014 Neurotechnology. All rights reserved.

PNG_DEFAULT_COMPRESSION_LEVEL Macro 1428
Files 1428
Png.h 1428
7.1.5.23 Tiff Unit 1429
Functions 1429
TiffLoadImageFromFile Function 1429
TiffLoadImageFromMemory Function 1429
TiffLoadImageFromStream Function 1430
Files 1430
Tiff.h 1430
7.1.5.24 Wsq Unit 1431
Functions 1432
WsqInfoGetBitRate Function 1432
WsqInfoGetImplementationNumber Function 1432
WsqInfoGetNistCom Function 1433
WsqInfoHasNistCom Function 1433
WsqInfoSetBitRate Function 1433
WsqInfoSetHasNistCom Function 1434
WsqLoadImageFromFile Function 1434
WsqLoadImageFromMemory Function 1434
WsqLoadImageFromStream Function 1435
WsqSaveImageToFile Function 1435
WsqSaveImageToMemory Function 1436
WsqSaveImageToStream Function 1437
Macros 1437
WSQ_DEFAULT_BIT_RATE Macro 1437
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_DEBIAN_AMD64 Macro 1437
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_DEBIAN_I386 Macro 1437
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_INTEL Macro 1438
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_INTEL64 Macro 1438
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_POWERPC Macro 1438
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_WIN32_X86 Macro 1438
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_WIN64_X64 Macro 1438
Files 1438
Wsq.h 1438
7.1.6 NSurveillance Library 1439
7.1.6.1 NSurveillance Module 1440
Functions 1443
NsedGetFaces Function 1443
NsedGetBestMatchesEx Function 1443
NsedGetOriginalImage Function 1444

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NsedGetExtractionStatus Function 1444
NsedGetSubjectTemplate Function 1445
NsedGetTemplate Function 1445
NsedGetFrameIndex Function 1445
NsedGetGrayscaleImage Function 1446
NsedGetTraceIndex Function 1446
NsedGetSourceIndex Function 1446
NsedGetDetectionDetails Function 1447
NsedGetTimeStamp Function 1447
NSurveillanceAddEventCallbackEx Function 1447
NSurveillanceAddIsRunningChangedCallback Function 1448
NSurveillanceAddSourceEx Function 1448
NSurveillanceAddSourceEx2 Function 1449
NSurveillanceAddTemplateEx Function 1449
NSurveillanceClearSourcesEx Function 1450
NSurveillanceGetInfo Function 1450
NSurveillanceCreate Function 1451
NSurveillanceGetDeviceManager Function 1451
NSurveillanceGetSourceCountEx Function 1451
NSurveillanceGetSourceEx Function 1452
NSurveillanceGetSourceEx2 Function 1452
NSurveillanceGetTemplateIdsEx Function 1452
NSurveillanceIsRunning Function 1453
NSurveillanceRemoveAllTemplatesEx Function 1453
NSurveillanceRemoveEventCallbackEx Function 1454
NSurveillanceRemoveIsRunningChangedCallback Function 1454
NSurveillanceRemoveSourceEx Function 1455
NSurveillanceRemoveTemplateEx Function 1455
NSurveillanceStartEx Function 1455
NSurveillanceStopEx Function 1456
NsedGetBestMatch Function 1456
NsedGetBestMatchCount Function 1456
Structs, Records, Enums 1457
NSurveillanceEventType Enumeration 1457
Macros 1458
NSP_DETECTION_DETECT_MORE_FACES Macro 1458
NSP_DETECTION_FACE_DETECTION_SCALE_COUNT Macro 1458
NSP_DETECTION_FACE_ROLL_ANGLE_BASE Macro 1458
NSP_DETECTION_MAX_ROLL_ANGLE_DEVIATION Macro 1459
NSP_DETECTION_MAX_YAW_ANGLE_DEVIATION Macro 1459
NSP_EXTRACTION_BLINK_CONFIDENCE_THRESHOLD Macro 1459

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSP_EXTRACTION_DARK_GLASSES_CONFIDENCE_THRESHOLD Macro 1459
NSP_EXTRACTION_DETECT_BLINK Macro 1459
NSP_EXTRACTION_DETECT_DARK_GLASSES Macro 1460
NSP_EXTRACTION_DETECT_EXPRESSION Macro 1460
NSP_EXTRACTION_DETECT_GENDER Macro 1460
NSP_EXTRACTION_DETECT_GLASSES Macro 1460
NSP_EXTRACTION_DETECT_MOUTH_OPEN Macro 1460
NSP_EXTRACTION_ENROLLMENT_TEMPLATE_SIZE Macro 1461
NSP_EXTRACTION_EXPRESSION_CONFIDENCE_THRESHOLD Macro 1461
NSP_EXTRACTION_GENDER_CONFIDENCE_THRESHOLD Macro 1461
NSP_EXTRACTION_GLASSES_CONFIDENCE_THRESHOLD Macro 1461
NSP_EXTRACTION_IDENTIFICATION_TEMPLATE_SIZE Macro 1461
NSP_EXTRACTION_MOUTH_OPEN_CONFIDENCE_THRESHOLD Macro 1462
NSP_MAX_THREAD_COUNT Macro 1462
NSP_IDENTIFICATION_MAX_BEST_MATCH_COUNT Macro 1462
NSP_DETECTION_MAX_IOD Macro 1463
NSP_DETECTION_MIN_IOD Macro 1463
NSP_DETECTION_FACE_CONFIDENCE_THRESHOLD Macro 1463
NSP_EXTRACTION_FACE_QUALITY_THRESHOLD Macro 1463
NSP_EXTRACTION_MAX_RECORDS_PER_TEMPLATE Macro 1464
NSP_IDENTIFICATION_MATCHING_THRESHOLD Macro 1464
Files 1464
NSurveillanceEventDetails.h 1464
NSurveillance.h 1465
NSurveillanceParams.h 1466

7.2 .NET Reference 1468
7.2.1 Neurotec Namespace 1469
7.2.1.1 Classes 1470
ErrorSuppressedEventArgs Class 1470
ErrorSuppressedEventArgs.ErrorSuppressedEventArgs Constructor 1470
ErrorSuppressedEventArgs Properties 1470
ErrorSuppressedEventArgs.Error Property 1470
NArray Class 1471
NArray Methods 1475
FromArray Method 1475
FromPointerArray Method 1485
FromResultArray Method 1485
NArray.GetEnumerator Method 1486
GetValueAs Method 1486
GetValueAsBoolean Method 1488

Copyright © 2009-2014 Neurotechnology. All rights reserved.

GetValueAsByte Method 1489
GetValueAsChar Method 1490
GetValueAsDouble Method 1491
GetValueAsInt16 Method 1492
GetValueAsInt32 Method 1492
GetValueAsInt64 Method 1493
GetValueAsIntPtr Method 1494
GetValueAsObject Method 1495
GetValueAsPointer Method 1496
GetValueAsResult Method 1497
GetValueAsSByte Method 1498
GetValueAsSingle Method 1498
GetValueAsString Method 1499
GetValueAsUInt16 Method 1500
GetValueAsUInt32 Method 1501
GetValueAsUInt64 Method 1502
GetValueAsUIntPtr Method 1503
ToArray Method 1504
ToBooleanArray Method 1506
ToByteArray Method 1506
ToCharArray Method 1507
ToDoubleArray Method 1508
ToInt16Array Method 1509
ToInt32Array Method 1509
ToInt64Array Method 1510
ToIntPtrArray Method 1511
ToObjectArray Method 1511
ToPointerArray Method 1512
ToResultArray Method 1513
ToSByteArray Method 1514
ToSingleArray Method 1515
ToStringArray Method 1515
ToUInt16Array Method 1516
ToUInt32Array Method 1517
ToUInt64Array Method 1517
ToUIntPtrArray Method 1518
NArray Properties 1519
NArray.Length Property 1519
NArray.NativeType Property 1519
NComplex Structure 1519
NComplex.NComplex Constructor 1520

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NComplex Fields 1520
NComplex.ImaginaryOne Field 1520
NComplex.One Field 1520
NComplex.Zero Field 1520
NComplex Methods 1520
ToString Method 1520
NComplex Properties 1521
NComplex.Imaginary Property 1521
NComplex.NativeType Property 1521
NComplex.Real Property 1521
NCore Class 1521
NCore Fields 1522
NCore.DllName Field 1522
NCore Methods 1522
AlignedAlloc Method 1522
AlignedCAlloc Method 1524
NCore.AlignedFree Method 1526
AlignedReAlloc Method 1526
Alloc Method 1528
CAlloc Method 1529
Clear Method 1530
Compare Method 1530
Copy Method 1531
Fill Method 1532
Free Method 1533
Move Method 1534
ReAlloc Method 1535
NCore.Shutdown Method 1535
NCore.ShutdownThread Method 1535
NCore Properties 1536
NCore.NativeModule Property 1536
NCore Events 1536
NCore.ErrorSuppressed Event 1536
NDisposable Class 1536
NDisposable Methods 1536
NDisposable.Dispose Method 1536
NDisposable Properties 1536
NDisposable.IsDisposed Property 1536
NExpandableObject Class 1537
NExpandableObject Properties 1539
NExpandableObject.NativeType Property 1539

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NExpandableObject.Properties Property 1539
NIndexPair Structure 1539
NIndexPair.NIndexPair Constructor 1540
NIndexPair Methods 1540
ToString Method 1540
NIndexPair Properties 1540
NIndexPair.Index1 Property 1541
NIndexPair.Index2 Property 1541
NIndexPair.NativeType Property 1541
NInfo Class 1541
NInfoObject Class 1541
NLibraryInfo Class 1541
NLibraryInfo Methods 1542
NLibraryInfo.GetString Method 1542
NLibraryInfo.Retrieve Method 1542
NLibraryInfo Properties 1542
NLibraryInfo.Activated Property 1542
NLibraryInfo.Company Property 1543
NLibraryInfo.Copyright Property 1543
NLibraryInfo.Product Property 1543
NLibraryInfo.Title Property 1543
NLibraryInfo.Version Property 1543
NModule Class 1543
NModule.NModule Constructor 1546
NModule Methods 1546
NModule.CheckInit Method 1546
NModule.GetDependences Method 1546
NModule.GetLoadedModules Method 1547
NModule Properties 1547
NModule.Activated Property 1547
NModule.Company Property 1547
NModule.Copyright Property 1547
NModule.Name Property 1547
NModule.NativeType Property 1547
NModule.NModuleOptionsNativeType Property 1548
NModule.Options Property 1548
NModule.Product Property 1548
NModule.Title Property 1548
NModule.Version Property 1548
NModule.VersionBuild Property 1548
NModule.VersionMajor Property 1548

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NModule.VersionMinor Property 1548
NModule.VersionRevision Property 1548
NObject Class 1549
NObject Fields 1551
NObject.FlagRefRet Field 1551
NObject Methods 1551
NObject.Clone Method 1551
NObject.CompareTo Method 1551
NObject.CopyParameters Method 1552
NObject.CopyProperties Method 1552
NObject.DisposeArray Method 1552
NObject.Equals Method 1552
FromHandle Method 1552
NObject.FromObject Method 1554
GetArray Method 1554
NObject.GetHashCode Method 1555
NObject.GetNativeType Method 1555
GetObject Method 1555
GetObjects Method 1555
GetParameter Method 1557
NObject.GetProperty Method 1558
GetSize Method 1558
GetSizeMany Method 1558
NObject.GetString Method 1559
Invoke Method 1559
NObject.Ref Method 1560
NObject.Reset Method 1560
NObject.ResetProperty Method 1560
Save Method 1560
SaveMany Method 1561
NObject.SetArray Method 1563
NObject.SetObject Method 1563
SetParameter Method 1563
SetProperty Method 1565
NObject.SetString Method 1565
NObject.ToHandle Method 1566
NObject.ToObject Method 1566
ToString Method 1566
NObject.Unref Method 1567
NObject.UnrefArray Method 1567
NObject.UnrefElements Method 1567

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Properties 1567
NObject.Flags Property 1567
NObject.IsDisposed Property 1567
NObject.NativeType Property 1568
NObject.Owner Property 1568
NObject Delegates 1568
NObject.EventInvoker Delegate 1568
NObject.FromHandlePlain Delegate 1568
NObject.GetArrayPlain Delegate 1568
NObject.GetArrayWithHandle Delegate 1569
NObject.GetObjectPlain Delegate 1569
NObject.GetObjectWithHandle Delegate 1569
NObject.GetOutArrayPlain Delegate 1569
NObject.GetOutArrayWithHandle Delegate 1570
NObject.GetStringPlainN Delegate 1570
NObject.GetStringWithHandleN Delegate 1570
NObject.SetArrayPlain Delegate 1571
NObject.SetArrayWithHandle Delegate 1571
NObject.SetObjectPlain Delegate 1571
NObject.SetObjectWithHandle Delegate 1571
NObject.SetStringPlainN Delegate 1572
NObject.SetStringWithHandleN Delegate 1572
NObject Events 1572
NObject.PropertyChanged Event 1572
NProcessorInfo Class 1572
NProcessorInfo Methods 1573
NProcessorInfo.GetModelInfo Method 1573
NProcessorInfo Properties 1573
NProcessorInfo.Count Property 1573
NProcessorInfo.Is3DNowSupported Property 1574
NProcessorInfo.IsLZCntSupported Property 1574
NProcessorInfo.IsMmxSupported Property 1574
NProcessorInfo.IsNeonSupported Property 1574
NProcessorInfo.IsPopCntSupported Property 1574
NProcessorInfo.IsSse2Supported Property 1574
NProcessorInfo.IsSse3Supported Property 1574
NProcessorInfo.IsSse41Supported Property 1574
NProcessorInfo.IsSse42Supported Property 1574
NProcessorInfo.IsSse4aSupported Property 1575
NProcessorInfo.IsSse5Supported Property 1575
NProcessorInfo.IsSseSupported Property 1575

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NProcessorInfo.IsSsse3Supported Property 1575
NProcessorInfo.ModelName Property 1575
NProcessorInfo.NativeType Property 1575
NProcessorInfo.NProcessorVendorNativeType Property 1575
NProcessorInfo.Vendor Property 1575
NProcessorInfo.VendorName Property 1575
NPropertyBag Class 1576
NPropertyBag Constructor 1579
NPropertyBag.NPropertyBag Constructor () 1579
NPropertyBag.NPropertyBag Constructor (IEnumerable<KeyValuePair<string, object>>) 1579
NPropertyBag Classes 1579
NPropertyBag.KeyCollection Class 1579
NPropertyBag.ValueCollection Class 1580
NPropertyBag Methods 1580
Add Method 1580
NPropertyBag.AddRange Method 1580
NPropertyBag.ApplyTo Method 1580
NPropertyBag.Clear Method 1581
NPropertyBag.Contains Method 1581
NPropertyBag.ContainsKey Method 1581
NPropertyBag.CopyTo Method 1581
NPropertyBag.GetEnumerator Method 1581
Parse Method 1582
Remove Method 1582
NPropertyBag.RemoveAt Method 1583
this Indexer 1583
NPropertyBag.TryGetValue Method 1583
TryParse Method 1584
NPropertyBag Properties 1584
NPropertyBag.Count Property 1584
NPropertyBag.IsReadOnly Property 1585
NPropertyBag.Keys Property 1585
NPropertyBag.NativeType Property 1585
NPropertyBag.Values Property 1585
NRange Structure 1585
NRange.NRange Constructor 1585
NRange Methods 1586
ToString Method 1586
NRange Properties 1586
NRange.From Property 1586
NRange.NativeType Property 1586

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NRange.To Property 1586
NRational Structure 1587
NRational.NRational Constructor 1587
NRational Fields 1587
NRational.Empty Field 1587
NRational Methods 1587
ToString Method 1587
NRational Properties 1588
NRational.Denominator Property 1588
NRational.NativeType Property 1588
NRational.Numerator Property 1588
NType Class 1588
NType Methods 1592
NType.CopyParameters Method 1592
NType.CopyPropertyValues Method 1592
NType.GetParameter Method 1592
NType.GetPropertyValue Method 1593
IdentifierToString Method 1593
NType.IsAssignableFrom Method 1594
NType.IsInstanceOfType Method 1594
NType.IsSubclassOf Method 1594
NType.Reset Method 1594
NType.ResetPropertyValue Method 1595
SetParameter Method 1595
SetPropertyValue Method 1595
NType.ToNativeType Method 1596
ToType Method 1596
NType Properties 1597
NType.BaseType Property 1597
NType.HasOwnerType Property 1597
NType.IsAbstract Property 1597
NType.IsBasic Property 1597
NType.IsCallback Property 1597
NType.IsCloneable Property 1598
NType.IsComparable Property 1598
NType.IsDeprecated Property 1598
NType.IsDisposable Property 1598
NType.IsEnum Property 1598
NType.IsEquatable Property 1598
NType.IsFlagsEnum Property 1598
NType.IsHandle Property 1598

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NType.IsMemorySerializable Property 1598
NType.IsObject Property 1599
NType.IsParsable Property 1599
NType.IsPrimitive Property 1599
NType.IsPublic Property 1599
NType.IsSealed Property 1599
NType.IsSerializable Property 1599
NType.IsSignNeutral Property 1599
NType.IsStatic Property 1599
NType.IsStruct Property 1599
NType.Module Property 1600
NType.Name Property 1600
NType.NativeType Property 1600
NType.NTypeCodeNativeType Property 1600
NType.OwnerType Property 1600
NType.RootType Property 1600
NType.RootTypeCode Property 1600
NType.TypeCode Property 1600
NType.UseInsteadType Property 1600
NType.ValueSize Property 1601
NTypes Class 1601
NTypes Fields 1602
NTypes.CharSet Field 1602
NTypes.StringEncoding Field 1602
NTypes.WCharApi Field 1602
NTypes Methods 1602
CreateCallback Method 1602
NTypes.FreeObjectPointer Method 1603
NTypes.FreeStringArray Method 1603
NTypes.FreeStringElements Method 1603
NTypes.GetObjectFromPointer Method 1603
NTypes.GetPointerForObject Method 1604
NTypes.IsMemoryTypeValid Method 1604
NTypes.IsMemoryTypeValidForFree Method 1604
NTypes Properties 1604
NTypes.ACharNativeType Property 1604
NTypes.BooleanNativeType Property 1605
NTypes.CharNativeType Property 1605
NTypes.DateTimeNativeType Property 1605
NTypes.DoubleNativeType Property 1605
NTypes.GuidNativeType Property 1605

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTypes.Int16NativeType Property 1605
NTypes.Int32NativeType Property 1605
NTypes.Int64NativeType Property 1605
NTypes.Int8NativeType Property 1605
NTypes.IntPtrNativeType Property 1606
NTypes.NameStringPairNativeType Property 1606
NTypes.NativeType Property 1606
NTypes.NAttributesNativeType Property 1606
NTypes.NMemoryTypeNativeType Property 1606
NTypes.NOSFamilyNativeType Property 1606
NTypes.OSFamilyCurrent Property 1606
NTypes.PointerNativeType Property 1606
NTypes.ResultNativeType Property 1606
NTypes.SingleNativeType Property 1607
NTypes.StringNativeType Property 1607
NTypes.TimeSpanNativeType Property 1607
NTypes.UInt16NativeType Property 1607
NTypes.UInt32NativeType Property 1607
NTypes.UInt64NativeType Property 1607
NTypes.UInt8NativeType Property 1607
NTypes.UIntPtrNativeType Property 1607
NTypes.WCharNativeType Property 1607
NURational Structure 1608
NURational.NURational Constructor 1608
NURational Fields 1608
NURational.Empty Field 1608
NURational Methods 1608
ToString Method 1608
NURational Properties 1609
NURational.Denominator Property 1609
NURational.NativeType Property 1609
NURational.Numerator Property 1609
NValue Class 1609
NValue Constructor 1613
NValue.NValue Constructor (IntPtr) 1613
NValue.NValue Constructor (IntPtr, NAttributes) 1613
NValue.NValue Constructor (UIntPtr) 1613
NValue.NValue Constructor (UIntPtr, NAttributes) 1613
NValue.NValue Constructor (bool) 1614
NValue.NValue Constructor (bool, NAttributes) 1614
NValue.NValue Constructor (byte) 1614

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NValue.NValue Constructor (byte, NAttributes) 1614
NValue.NValue Constructor (char) 1614
NValue.NValue Constructor (char, NAttributes) 1615
NValue.NValue Constructor (double) 1615
NValue.NValue Constructor (double, NAttributes) 1615
NValue.NValue Constructor (float) 1615
NValue.NValue Constructor (float, NAttributes) 1615
NValue.NValue Constructor (int) 1616
NValue.NValue Constructor (int, NAttributes) 1616
NValue.NValue Constructor (long) 1616
NValue.NValue Constructor (long, NAttributes) 1616
NValue.NValue Constructor (sbyte) 1616
NValue.NValue Constructor (sbyte, NAttributes) 1617
NValue.NValue Constructor (short) 1617
NValue.NValue Constructor (short, NAttributes) 1617
NValue.NValue Constructor (uint) 1617
NValue.NValue Constructor (uint, NAttributes) 1617
NValue.NValue Constructor (ulong) 1618
NValue.NValue Constructor (ulong, NAttributes) 1618
NValue.NValue Constructor (ushort) 1618
NValue.NValue Constructor (ushort, NAttributes) 1618
NValue Methods 1618
ChangeType Method 1619
FromObject Method 1619
FromPointer Method 1621
FromResult Method 1621
FromString Method 1622
FromType Method 1622
ToBoolean Method 1623
ToByte Method 1624
ToChar Method 1624
ToDouble Method 1625
ToInt16 Method 1626
ToInt32 Method 1627
ToInt64 Method 1627
ToIntPtr Method 1628
ToObject Method 1629
ToPointer Method 1630
ToResult Method 1631
ToSByte Method 1632
ToSingle Method 1632

Copyright © 2009-2014 Neurotechnology. All rights reserved.

ToString Method 1633
ToType Method 1634
ToUInt16 Method 1636
ToUInt32 Method 1636
ToUInt64 Method 1637
ToUIntPtr Method 1638
NValue Properties 1639
NValue.Attributes Property 1639
NValue.IsWrapper Property 1639
NValue.KeyValuePairNativeType Property 1639
NValue.LongSize Property 1639
NValue.NameValuePairNativeType Property 1639
NValue.NativeSize Property 1639
NValue.NativeType Property 1639
NValue.Ptr Property 1639
NValue.Size Property 1639
NValue.TypeCode Property 1640
NValue.ValueType Property 1640
NVersion Structure 1640
NVersion Operator 1641
NVersion.NVersion Constructor (int, int) 1641
NVersion.NVersion Constructor (ushort) 1641
NVersion.NVersion Operator (ushort) 1641
NVersion Fields 1641
NVersion.Empty Field 1641
NVersion Methods 1641
NVersion.CompareTo Method 1642
Equals Method 1642
NVersion.GetHashCode Method 1642
ToString Method 1642
NVersion Operators 1643
NVersion.!= Operator 1643
NVersion.< Operator 1643
NVersion.<= Operator 1644
NVersion.== Operator 1644
NVersion.> Operator 1644
NVersion.>= Operator 1644
NVersion.ushort Operator 1645
NVersion Properties 1645
NVersion.Major Property 1645
NVersion.Minor Property 1645

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NVersion.NativeType Property 1645
NVersionRange Structure 1645
NVersionRange Operator 1646
NVersionRange.NVersionRange Constructor (NVersion, NVersion) 1646
NVersionRange.NVersionRange Constructor (uint) 1646
NVersionRange.NVersionRange Operator (uint) 1646
NVersionRange Fields 1647
NVersionRange.Empty Field 1647
NVersionRange Methods 1647
Contains Method 1647
Equals Method 1647
NVersionRange.GetHashCode Method 1648
Intersect Method 1648
NVersionRange.IntersectsWith Method 1649
ToString Method 1649
NVersionRange Operators 1649
NVersionRange.!= Operator 1650
NVersionRange.== Operator 1650
NVersionRange.uint Operator 1650
NVersionRange Properties 1650
NVersionRange.From Property 1650
NVersionRange.NativeType Property 1650
NVersionRange.To Property 1651
7.2.1.2 Structs, Records, Enums 1651
Neurotec.NMemoryType Enumeration 1651
Neurotec.NModuleOptions Enumeration 1651
Neurotec.NProcessorVendor Enumeration 1652
Neurotec.NTypeCode Enumeration 1652
7.2.1.3 Types 1654
Neurotec.NLibraryGetInfo Type 1654
7.2.2 Neurotec.Biometrics Namespace 1654
7.2.2.1 Classes 1655
NBiometricAttributes Class 1655
NBiometricAttributes Properties 1658
NBiometricAttributes.BiometricType Property 1658
NBiometricAttributes.NativeType Property 1658
NBiometricAttributes.Quality Property 1658
NBiometricAttributes.Status Property 1658
NBiometrics Class 1658
NBiometrics Fields 1658
NBiometrics.DllName Field 1659

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBiometrics Properties 1659
NBiometrics.NativeModule Property 1659
NBiometricTypes Class 1659
NBiometricTypes Fields 1661
NBiometricTypes.PhraseIdUnknown Field 1661
NBiometricTypes.QualityFailed Field 1661
NBiometricTypes.QualityMax Field 1661
NBiometricTypes.QualityMin Field 1662
NBiometricTypes.QualityUnknown Field 1662
NBiometricTypes Methods 1662
NBiometricTypes.AngleFromRadians Method 1662
NBiometricTypes.AngleToRadians Method 1662
AngleToString Method 1662
GetPositionAvailableParts Method 1663
NBiometricTypes.IsBiometricStatusFinal Method 1664
NBiometricTypes.IsBiometricStatusValid Method 1664
NBiometricTypes.IsBiometricSubtypeValid Method 1664
NBiometricTypes.IsBiometricTypeValid Method 1664
NBiometricTypes.IsImpressionTypeCompatibleWith Method 1665
NBiometricTypes.IsImpressionTypeContact Method 1665
NBiometricTypes.IsImpressionTypeContactless Method 1665
NBiometricTypes.IsImpressionTypeFinger Method 1665
NBiometricTypes.IsImpressionTypeGeneric Method 1666
NBiometricTypes.IsImpressionTypeLatent Method 1666
NBiometricTypes.IsImpressionTypeLiveScan Method 1666
NBiometricTypes.IsImpressionTypeNonliveScan Method 1666
NBiometricTypes.IsImpressionTypeNonOptical Method 1667
NBiometricTypes.IsImpressionTypeOneOf Method 1667
NBiometricTypes.IsImpressionTypeOptical Method 1667
NBiometricTypes.IsImpressionTypePalm Method 1667
NBiometricTypes.IsImpressionTypePlain Method 1668
NBiometricTypes.IsImpressionTypeRolled Method 1668
NBiometricTypes.IsImpressionTypeSwipe Method 1668
NBiometricTypes.IsImpressionTypeValid Method 1668
NBiometricTypes.IsImpressionTypeValidFinger Method 1669
NBiometricTypes.IsImpressionTypeValidPalm Method 1669
IsPositionCompatibleWith Method 1669
NBiometricTypes.IsPositionFinger Method 1670
NBiometricTypes.IsPositionFourFingers Method 1670
IsPositionKnown Method 1670
IsPositionLeft Method 1671

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBiometricTypes.IsPositionLeftAndRight Method 1671
IsPositionOneOf Method 1672
NBiometricTypes.IsPositionPalm Method 1672
IsPositionRight Method 1672
NBiometricTypes.IsPositionRightAndLeft Method 1673
NBiometricTypes.IsPositionSingleEye Method 1673
NBiometricTypes.IsPositionSingleFinger Method 1674
NBiometricTypes.IsPositionTheEye Method 1674
NBiometricTypes.IsPositionTheFinger Method 1674
NBiometricTypes.IsPositionThePalm Method 1674
NBiometricTypes.IsPositionThreeFingers Method 1675
NBiometricTypes.IsPositionTwoEyes Method 1675
NBiometricTypes.IsPositionTwoFingers Method 1675
IsPositionValid Method 1675
NBiometricTypes.IsPositionValidFinger Method 1676
NBiometricTypes.IsPositionValidPalm Method 1676
NBiometricTypes.IsPositionValidTheEye Method 1676
NBiometricTypes.IsPositionValidTheFinger Method 1677
NBiometricTypes.IsQualityValid Method 1677
QualityToString Method 1677
NBiometricTypes.ToImpressionTypeFinger Method 1678
NBiometricTypes Properties 1678
NBiometricTypes.NativeType Property 1678
NBiometricTypes.NBiometricStatusNativeType Property 1678
NBiometricTypes.NBiometricSubtypeNativeType Property 1678
NBiometricTypes.NBiometricTypeNativeType Property 1678
NBiometricTypes.NEPositionNativeType Property 1678
NBiometricTypes.NFImpressionTypeNativeType Property 1679
NBiometricTypes.NFMinutiaFormatNativeType Property 1679
NBiometricTypes.NFMinutiaTypeNativeType Property 1679
NBiometricTypes.NFPatternClassNativeType Property 1679
NBiometricTypes.NFPositionNativeType Property 1679
NBiometricTypes.NFRidgeCountsTypeNativeType Property 1679
NBiometricTypes.NGenderNativeType Property 1679
NBiometricTypes.NLExpressionNativeType Property 1679
NBiometricTypes.NLPropertiesNativeType Property 1679
NFCore Structure 1680
NFCore Constructor 1680
NFCore.NFCore Constructor (ushort, ushort) 1680
NFCore.NFCore Constructor (ushort, ushort, double) 1680
NFCore.NFCore Constructor (ushort, ushort, int) 1681

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFCore Methods 1681
ToString Method 1681
NFCore Properties 1681
NFCore.Angle Property 1681
NFCore.NativeType Property 1681
NFCore.RawAngle Property 1682
NFCore.X Property 1682
NFCore.Y Property 1682
NFDelta Structure 1682
NFDelta Constructor 1683
NFDelta.NFDelta Constructor (ushort, ushort) 1683
NFDelta.NFDelta Constructor (ushort, ushort, double, double, double) 1683
NFDelta.NFDelta Constructor (ushort, ushort, int, int, int) 1683
NFDelta Methods 1684
ToString Method 1684
NFDelta Properties 1684
NFDelta.Angle1 Property 1684
NFDelta.Angle2 Property 1685
NFDelta.Angle3 Property 1685
NFDelta.NativeType Property 1685
NFDelta.RawAngle1 Property 1685
NFDelta.RawAngle2 Property 1685
NFDelta.RawAngle3 Property 1685
NFDelta.X Property 1686
NFDelta.Y Property 1686
NFDoubleCore Structure 1686
NFDoubleCore.NFDoubleCore Constructor 1687
NFDoubleCore Methods 1687
ToString Method 1687
NFDoubleCore Properties 1687
NFDoubleCore.NativeType Property 1687
NFDoubleCore.X Property 1687
NFDoubleCore.Y Property 1688
NFMinutia Structure 1688
NFMinutia Constructor 1688
NFMinutia.NFMinutia Constructor (ushort, ushort, NFMinutiaType, byte) 1688
NFMinutia.NFMinutia Constructor (ushort, ushort, NFMinutiaType, byte, byte, byte, byte) 1689
NFMinutia.NFMinutia Constructor (ushort, ushort, NFMinutiaType, double) 1689
NFMinutia.NFMinutia Constructor (ushort, ushort, NFMinutiaType, double, byte, byte, byte) 1689
NFMinutia Methods 1690
ToString Method 1690

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFMinutia Properties 1690
NFMinutia.Angle Property 1690
NFMinutia.Curvature Property 1690
NFMinutia.G Property 1691
NFMinutia.NativeType Property 1691
NFMinutia.Quality Property 1691
NFMinutia.RawAngle Property 1691
NFMinutia.Type Property 1691
NFMinutia.X Property 1692
NFMinutia.Y Property 1692
NFMinutiaNeighbor Structure 1692
NFMinutiaNeighbor.NFMinutiaNeighbor Constructor 1693
NFMinutiaNeighbor Fields 1693
NFMinutiaNeighbor.Empty Field 1693
NFMinutiaNeighbor Methods 1693
ToString Method 1693
NFMinutiaNeighbor Properties 1693
NFMinutiaNeighbor.Index Property 1693
NFMinutiaNeighbor.NativeType Property 1694
NFMinutiaNeighbor.RidgeCount Property 1694
NLAttributes Class 1694
NLAttributes.NLAttributes Constructor 1697
NLAttributes Properties 1697
NLAttributes.NativeType Property 1697
NleDetectionDetails Class 1697
NleDetectionDetails.NleDetectionDetails Constructor 1698
NleDetectionDetails Methods 1698
NleDetectionDetails.Clear Method 1698
NleDetectionDetails Properties 1698
NleDetectionDetails.BlinkConfidence Property 1698
NleDetectionDetails.DarkGlassesConfidence Property 1698
NleDetectionDetails.Expression Property 1698
NleDetectionDetails.ExpressionConfidence Property 1698
NleDetectionDetails.Face Property 1698
NleDetectionDetails.FaceAvailable Property 1698
NleDetectionDetails.Gender Property 1699
NleDetectionDetails.GenderConfidence Property 1699
NleDetectionDetails.GlassesConfidence Property 1699
NleDetectionDetails.LeftEyeCenter Property 1699
NleDetectionDetails.MouthCenter Property 1699
NleDetectionDetails.MouthOpenConfidence Property 1699

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NleDetectionDetails.NoseTip Property 1699
NleDetectionDetails.Points Property 1700
NleDetectionDetails.Properties Property 1700
NleDetectionDetails.RightEyeCenter Property 1700
NleEyes Structure 1700
NleEyes Properties 1700
NleEyes.First Property 1700
NleEyes.FirstConfidence Property 1700
NleEyes.Second Property 1701
NleEyes.SecondConfidence Property 1701
NleFace Structure 1701
NleFace Properties 1701
NleFace.Confidence Property 1701
NleFace.Rectangle Property 1701
NleFace.Rotation Property 1702
NleRotation Structure 1702
NleRotation Properties 1702
NleRotation.Pitch Property 1702
NleRotation.Roll Property 1702
NleRotation.Yaw Property 1702
NLExtractor Class 1702
NLExtractor.NLExtractor Constructor 1708
NLExtractor Fields 1708
NLExtractor.ParameterBlinkConfidenceThreshold Field 1708
NLExtractor.ParameterDarkGlassesConfidenceThreshold Field 1708
NLExtractor.ParameterDetectAllFeaturePoints Field 1709
NLExtractor.ParameterDetectBaseFeaturePoints Field 1709
NLExtractor.ParameterDetectBlink Field 1709
NLExtractor.ParameterDetectDarkGlasses Field 1709
NLExtractor.ParameterDetectExpression Field 1709
NLExtractor.ParameterDetectGender Field 1709
NLExtractor.ParameterDetectGlasses Field 1709
NLExtractor.ParameterDetectMoreFaces Field 1710
NLExtractor.ParameterDetectMouthOpen Field 1710
NLExtractor.ParameterExpressionConfidenceThreshold Field 1710
NLExtractor.ParameterFaceConfidenceThreshold Field 1710
NLExtractor.ParameterFaceDetectionScaleCount Field 1710
NLExtractor.ParameterFaceQualityThreshold Field 1710
NLExtractor.ParameterFaceRollAngleBase Field 1710
NLExtractor.ParameterFavorLargestFace Field 1710
NLExtractor.ParameterGenderConfidenceThreshold Field 1711

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLExtractor.ParameterGlassesConfidenceThreshold Field 1711
NLExtractor.ParameterLivenessThreshold Field 1711
NLExtractor.ParameterMaxIod Field 1711
NLExtractor.ParameterMaxRecordsPerTemplate Field 1711
NLExtractor.ParameterMaxRollAngleDeviation Field 1711
NLExtractor.ParameterMaxStreamDurationInFrames Field 1711
NLExtractor.ParameterMaxYawAngleDeviation Field 1712
NLExtractor.ParameterMinIod Field 1712
NLExtractor.ParameterMouthOpenConfidenceThreshold Field 1712
NLExtractor.ParameterTemplateSize Field 1712
NLExtractor.ParameterUseLivenessCheck Field 1712
NLExtractor Methods 1712
NLExtractor.Compress Method 1712
NLExtractor.DetectFace Method 1712
NLExtractor.DetectFaces Method 1713
NLExtractor.DetectFacialFeatures Method 1713
NLExtractor.Extract Method 1714
NLExtractor.ExtractEnd Method 1714
NLExtractor.ExtractNext Method 1714
NLExtractor.ExtractStart Method 1714
NLExtractor.ExtractUsingDetails Method 1715
NLExtractor.Generalize Method 1715
NLExtractor Properties 1715
NLExtractor.BlinkConfidenceThreshold Property 1715
NLExtractor.DarkGlassesConfidenceThreshold Property 1715
NLExtractor.DetectAllFeaturePoints Property 1716
NLExtractor.DetectBaseFeaturePoints Property 1716
NLExtractor.DetectBlink Property 1716
NLExtractor.DetectDarkGlasses Property 1716
NLExtractor.DetectExpression Property 1716
NLExtractor.DetectGender Property 1716
NLExtractor.DetectGlasses Property 1717
NLExtractor.DetectMoreFaces Property 1717
NLExtractor.DetectMouthOpen Property 1717
NLExtractor.ExpressionConfidenceThreshold Property 1717
NLExtractor.FaceConfidenceThreshold Property 1717
NLExtractor.FaceDetectionScaleCount Property 1718
NLExtractor.FaceQualityThreshold Property 1718
NLExtractor.FaceRollAngleBase Property 1718
NLExtractor.FavorLargestFace Property 1718
NLExtractor.GenderConfidenceThreshold Property 1718

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLExtractor.GlassesConfidenceThreshold Property 1718
NLExtractor.LivenessThreshold Property 1719
NLExtractor.MaxIod Property 1719
NLExtractor.MaxRecordsPerTemplate Property 1719
NLExtractor.MaxRollAngleDeviation Property 1719
NLExtractor.MaxStreamDurationInFrames Property 1720
NLExtractor.MaxYawAngleDeviation Property 1720
NLExtractor.MinIod Property 1720
NLExtractor.MouthOpenConfidenceThreshold Property 1720
NLExtractor.NativeType Property 1720
NLExtractor.NleExtractionStatusNativeType Property 1720
NLExtractor.NleTemplateSizeNativeType Property 1721
NLExtractor.TemplateSize Property 1721
NLExtractor.UseLivenessCheck Property 1721
NLFeaturePoint Structure 1721
NLFeaturePoint Constructor 1722
NLFeaturePoint.NLFeaturePoint Constructor (ushort, ushort, ushort) 1722
NLFeaturePoint.NLFeaturePoint Constructor (ushort, ushort, ushort, byte) 1722
NLFeaturePoint Methods 1722
ToString Method 1722
NLFeaturePoint Properties 1723
NLFeaturePoint.Code Property 1723
NLFeaturePoint.Confidence Property 1723
NLFeaturePoint.NativeType Property 1723
NLFeaturePoint.X Property 1723
NLFeaturePoint.Y Property 1723
NLMatchingDetails Class 1723
NLMatchingDetails Properties 1723
NLMatchingDetails.NativeType Property 1723
NLRecord Class 1724
NLRecord Constructor 1726
NLRecord.NLRecord Constructor () 1726
NLRecord.NLRecord Constructor (NBuffer) 1726
NLRecord.NLRecord Constructor (NBuffer, uint) 1727
NLRecord.NLRecord Constructor (NBuffer, uint, out int) 1727
NLRecord.NLRecord Constructor (byte[]) 1727
NLRecord.NLRecord Constructor (uint) 1727
NLRecord Methods 1728
Check Method 1728
GetQuality Method 1728
GetSize Method 1729

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLRecord Properties 1729
NLRecord.CbeffProductType Property 1729
NLRecord.NativeType Property 1729
NLRecord.Quality Property 1730
NLTemplate Class 1730
NLTemplate Constructor 1733
NLTemplate.NLTemplate Constructor () 1733
NLTemplate.NLTemplate Constructor (NBuffer) 1733
NLTemplate.NLTemplate Constructor (NBuffer, uint) 1733
NLTemplate.NLTemplate Constructor (NBuffer, uint, out int) 1733
NLTemplate.NLTemplate Constructor (byte[]) 1733
NLTemplate.NLTemplate Constructor (uint) 1734
NLTemplate Classes 1734
NLTemplate.RecordCollection Class 1734
NLTemplate Fields 1734
NLTemplate.FlagProcessFirstRecordOnly Field 1734
NLTemplate.MaxRecordCount Field 1734
NLTemplate Methods 1734
NLTemplate.CalculateSize Method 1734
Check Method 1735
GetRecordCount Method 1735
GetSize Method 1736
NLTemplate Properties 1736
NLTemplate.NativeType Property 1736
NLTemplate.Records Property 1736
NMatcher Class 1737
NMatcher.NMatcher Constructor 1742
NMatcher Fields 1742
NMatcher.FingersModeAtmelFingerChip Field 1743
NMatcher.FingersModeAuthentecAes2501B Field 1743
NMatcher.FingersModeAuthentecAes4000 Field 1743
NMatcher.FingersModeAuthentecAFS2 Field 1743
NMatcher.FingersModeBiometrikaFX2000 Field 1743
NMatcher.FingersModeBiometrikaFX3000 Field 1743
NMatcher.FingersModeBmfBlp100 Field 1743
NMatcher.FingersModeCrossMatchVerifier300 Field 1743
NMatcher.FingersModeDigentIzzix Field 1743
NMatcher.FingersModeDigitalPersonaUareU Field 1744
NMatcher.FingersModeEthentica Field 1744
NMatcher.FingersModeFujitsuMbf200 Field 1744
NMatcher.FingersModeFutronicFS80 Field 1744

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMatcher.FingersModeGeneral Field 1744
NMatcher.FingersModeIdenticatorTechnologyDF90 Field 1744
NMatcher.FingersModeIdentixDfr2090 Field 1744
NMatcher.FingersModeIdentixTouchView Field 1744
NMatcher.FingersModeKeytronicSecureDesktop Field 1744
NMatcher.FingersModeLighTuningLttC500 Field 1745
NMatcher.FingersModeNitgenFingkeyHamster Field 1745
NMatcher.FingersModePreciseBiometrics100CS Field 1745
NMatcher.FingersModeSecugenHamster Field 1745
NMatcher.FingersModeStartekFM200 Field 1745
NMatcher.FingersModeTacomaCmos Field 1745
NMatcher.FingersModeTestechBioI Field 1745
NMatcher.FingersModeUpekTouchChip Field 1745
NMatcher.ParameterFacesMatcherBase Field 1745
NMatcher.ParameterFacesMatchingSpeed Field 1746
NMatcher.ParameterFacesMatchingThresholdNew Field 1746
NMatcher.ParameterFingersFillFullMatchingDetails Field 1746
NMatcher.ParameterFingersMatcherBase Field 1746
NMatcher.ParameterFingersMatchingSpeed Field 1746
NMatcher.ParameterFingersMaximalRotation Field 1746
NMatcher.ParameterFingersMinMatchedCount Field 1746
NMatcher.ParameterFingersMinMatchedCountThreshold Field 1746
NMatcher.ParameterFingersMode Field 1747
NMatcher.ParameterFusionType Field 1747
NMatcher.ParameterIrisesMatcherBase Field 1747
NMatcher.ParameterIrisesMatchingSpeed Field 1747
NMatcher.ParameterIrisesMatchingThreshold Field 1747
NMatcher.ParameterIrisesMaximalRotation Field 1747
NMatcher.ParameterIrisesMinMatchedCount Field 1747
NMatcher.ParameterIrisesMinMatchedCountThreshold Field 1747
NMatcher.ParameterMatchingThreshold Field 1748
NMatcher.ParameterPalmsFillFullMatchingDetails Field 1748
NMatcher.ParameterPalmsMatcherBase Field 1748
NMatcher.ParameterPalmsMatchingSpeed Field 1748
NMatcher.ParameterPalmsMaximalRotation Field 1748
NMatcher.ParameterPalmsMinMatchedCount Field 1748
NMatcher.ParameterPalmsMinMatchedCountThreshold Field 1748
NMatcher.ParameterVoicesMatcherBase Field 1749
NMatcher.ParameterVoicesUniquePhrasesOnly Field 1749
NMatcher.PartFaces Field 1749
NMatcher.PartFingers Field 1749

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMatcher.PartIrises Field 1749
NMatcher.PartNone Field 1749
NMatcher.PartPalms Field 1749
NMatcher Methods 1749
NMatcher.IdentifyEnd Method 1749
IdentifyNext Method 1750
IdentifyStart Method 1751
NMatcher.PrepareTemplate Method 1752
Verify Method 1752
NMatcher Properties 1754
NMatcher.FacesMatchingSpeed Property 1754
NMatcher.FacesMatchingThresholdNew Property 1754
NMatcher.FingersMatchingSpeed Property 1754
NMatcher.FingersMaximalRotation Property 1754
NMatcher.FingersMinMatchedCount Property 1754
NMatcher.FingersMinMatchedCountThreshold Property 1755
NMatcher.FingersMode Property 1755
NMatcher.FusionType Property 1755
NMatcher.IrisesMatchingSpeed Property 1755
NMatcher.IrisesMatchingThreshold Property 1755
NMatcher.IrisesMaximalRotation Property 1755
NMatcher.IrisesMinMatchedCount Property 1756
NMatcher.IrisesMinMatchedCountThreshold Property 1756
NMatcher.MatchingThreshold Property 1756
NMatcher.NativeType Property 1756
NMatcher.NMatchingSpeedNativeType Property 1756
NMatcher.NMFusionTypeNativeType Property 1756
NMatcher.PalmsMatchingSpeed Property 1756
NMatcher.PalmsMaximalRotation Property 1756
NMatcher.PalmsMinMatchedCount Property 1757
NMatcher.PalmsMinMatchedCountThreshold Property 1757
NMatcher.VoicesUniquePhrasesOnly Property 1757
NMatchingDetails Class 1757
NMatchingDetails Constructor 1758
NMatchingDetails.NMatchingDetails Constructor (NBuffer) 1758
NMatchingDetails.NMatchingDetails Constructor (NBuffer, uint) 1758
NMatchingDetails.NMatchingDetails Constructor (NBuffer, uint, out int) 1758
NMatchingDetails.NMatchingDetails Constructor (Stream) 1759
NMatchingDetails.NMatchingDetails Constructor (Stream, uint) 1759
NMatchingDetails.NMatchingDetails Constructor (byte[]) 1759
NMatchingDetails Classes 1759

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMatchingDetails.FaceCollection Class 1759
NMatchingDetails.FingerCollection Class 1759
NMatchingDetails.IrisCollection Class 1759
NMatchingDetails.PalmCollection Class 1760
NMatchingDetails.VoiceCollection Class 1760
NMatchingDetails Properties 1760
NMatchingDetails.Faces Property 1760
NMatchingDetails.FacesMatchedIndex Property 1760
NMatchingDetails.FacesScore Property 1760
NMatchingDetails.Fingers Property 1760
NMatchingDetails.FingersScore Property 1760
NMatchingDetails.Irises Property 1761
NMatchingDetails.IrisesScore Property 1761
NMatchingDetails.NativeType Property 1761
NMatchingDetails.Palms Property 1761
NMatchingDetails.PalmsScore Property 1761
NMatchingDetails.Voices Property 1761
NMatchingDetails.VoicesScore Property 1762
NTemplate Class 1762
NTemplate Constructor 1765
NTemplate.NTemplate Constructor () 1765
NTemplate.NTemplate Constructor (NBuffer) 1765
NTemplate.NTemplate Constructor (NBuffer, uint) 1765
NTemplate.NTemplate Constructor (NBuffer, uint, out int) 1765
NTemplate.NTemplate Constructor (byte[]) 1766
NTemplate.NTemplate Constructor (uint) 1766
NTemplate Methods 1766
NTemplate.CalculateSize Method 1766
Check Method 1766
NTemplate.Clear Method 1767
GetSize Method 1767
NTemplate Properties 1767
NTemplate.Faces Property 1768
NTemplate.Fingers Property 1768
NTemplate.Irises Property 1768
NTemplate.NativeType Property 1768
NTemplate.Palms Property 1768
NTemplate.Voices Property 1768
7.2.2.2 Structs, Records, Enums 1768
Neurotec.Biometrics.NBiometricStatus Enumeration 1769
Neurotec.Biometrics.NBiometricSubtype Enumeration 1770

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Neurotec.Biometrics.NBiometricType Enumeration 1771
Neurotec.Biometrics.NEPosition Enumeration 1772
Neurotec.Biometrics.NFImpressionType Enumeration 1773
Neurotec.Biometrics.NFMinutiaFormat Enumeration 1774
Neurotec.Biometrics.NFMinutiaType Enumeration 1774
Neurotec.Biometrics.NFPatternClass Enumeration 1775
Neurotec.Biometrics.NFPosition Enumeration 1776
Neurotec.Biometrics.NFRidgeCountsType Enumeration 1778
Neurotec.Biometrics.NGender Enumeration 1778
Neurotec.Biometrics.NleExtractionStatus Enumeration 1779
Neurotec.Biometrics.NleTemplateSize Enumeration 1779
Neurotec.Biometrics.NLExpression Enumeration 1780
Neurotec.Biometrics.NLProperties Enumeration 1780
Neurotec.Biometrics.NMatchingSpeed Enumeration 1781
Neurotec.Biometrics.NMFusionType Enumeration 1781
7.2.3 Neurotec.Biometrics.Gui Namespace 1782
7.2.3.1 Classes 1782
NEView Class 1782
NEView.NEView Constructor 1783
NEView Properties 1783
NEView.Image Property 1783
NEView.InnerBoundaryColor Property 1783
NEView.InnerBoundaryWidth Property 1783
NEView.OuterBoundaryColor Property 1783
NEView.OuterBoundaryWidth Property 1783
NEView.SegmentationDetails Property 1784
NLView Class 1787
NLView.NLView Constructor 1787
NLView Properties 1787
NLView.BaseFeaturePointSize Property 1787
NLView.DetectionDetails Property 1788
NLView.DrawConfidenceForEyes Property 1788
NLView.DrawFaceConfidence Property 1788
NLView.FaceIds Property 1788
NLView.FaceRectangleColor Property 1788
NLView.FaceRectangleWidth Property 1789
NLView.FeaturePointSize Property 1789
NLView.Image Property 1789
NLView.ShowBaseFeaturePoints Property 1789
NLView.ShowExpression Property 1789
NLView.ShowExpressionConfidence Property 1790

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLView.ShowEyes Property 1790
NLView.ShowEyesConfidence Property 1790
NLView.ShowFaceConfidence Property 1790
NLView.ShowGender Property 1790
NLView.ShowGenderConfidence Property 1790
NLView.ShowMouth Property 1791
NLView.ShowMouthConfidence Property 1791
NLView.ShowNose Property 1791
NLView.ShowNoseConfidence Property 1791
NLView.ShowProperties Property 1791
NLView.ShowPropertiesConfidence Property 1792
7.2.4 Neurotec.Devices Namespace 1792
7.2.4.1 Classes 1792
NBiometricDevice Class 1792
NBiometricDevice Methods 1796
NBiometricDevice.Cancel Method 1796
NBiometricDevice.EndSequence Method 1796
NBiometricDevice.StartSequence Method 1796
NBiometricDevice Properties 1796
NBiometricDevice.BiometricType Property 1796
NBiometricDevice.IsSpoofDetectionSupported Property 1796
NBiometricDevice.NativeType Property 1797
NBiometricDevice.ProductId Property 1797
NBiometricDevice.SpoofDetection Property 1797
NBiometricDevice.VendorId Property 1797
NCamera Class 1797
NCamera Methods 1801
NCamera.CaptureStill Method 1801
NCamera.Focus Method 1801
NCamera.GetCurrentStillFormat Method 1801
NCamera.GetFrame Method 1801
NCamera.GetStillFormats Method 1802
NCamera.ResetFocus Method 1802
NCamera.SetCurrentStillFormat Method 1802
NCamera Properties 1802
NCamera.FocusRegion Property 1802
NCamera.IsFocusRegionSupported Property 1802
NCamera.IsFocusSupported Property 1802
NCamera.IsStillCaptureSupported Property 1803
NCamera.NativeType Property 1803
NCamera.NCameraStatusNativeType Property 1803

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCameraStillCapturedEventArgs Class 1803
NCameraStillCapturedEventArgs.NCameraStillCapturedEventArgs Constructor 1803
NCameraStillCapturedEventArgs Properties 1803
NCameraStillCapturedEventArgs.Id Property 1804
NCameraStillCapturedEventArgs.Properties Property 1804
NCameraStillCapturedEventArgs.Stream Property 1804
NCaptureDevice Class 1804
NCaptureDevice Methods 1807
NCaptureDevice.GetCurrentFormat Method 1807
NCaptureDevice.GetFormats Method 1807
NCaptureDevice.SetCurrentFormat Method 1807
NCaptureDevice.StartCapturing Method 1808
NCaptureDevice.StopCapturing Method 1808
NCaptureDevice Properties 1808
NCaptureDevice.IsCapturing Property 1808
NCaptureDevice.MediaType Property 1808
NCaptureDevice.NativeType Property 1808
NCaptureDevice Events 1808
NCaptureDevice.IsCapturingChanged Event 1808
NDevice Class 1809
NDevice Classes 1811
NDevice.ChildCollection Class 1811
NDevice Properties 1812
NDevice.Children Property 1812
NDevice.DeviceType Property 1812
NDevice.DisplayName Property 1812
NDevice.Id Property 1812
NDevice.IsAvailable Property 1812
NDevice.IsDisconnectable Property 1812
NDevice.IsPrivate Property 1813
NDevice.Make Property 1813
NDevice.Model Property 1813
NDevice.NativeType Property 1813
NDevice.NDeviceTypeNativeType Property 1813
NDevice.Owner Property 1813
NDevice.Parent Property 1813
NDevice.Plugin Property 1814
NDevice.SerialNumber Property 1814
NDeviceManager Class 1814
NDeviceManager Constructor 1817
NDeviceManager.NDeviceManager Constructor () 1817

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDeviceManager.NDeviceManager Constructor (NDeviceType) 1817
NDeviceManager.NDeviceManager Constructor (NDeviceType, bool) 1817
NDeviceManager.NDeviceManager Constructor (NDeviceType, bool, bool) 1818
NDeviceManager Classes 1818
NDeviceManager.DeviceCollection Class 1818
NDeviceManager Methods 1819
ConnectToDevice Method 1819
NDeviceManager.DisconnectFromDevice Method 1819
NDeviceManager.GetConnectToDeviceParameters Method 1820
NDeviceManager.IsConnectToDeviceSupported Method 1820
NDeviceManager.Refresh Method 1820
NDeviceManager Properties 1820
NDeviceManager.AutoUpdate Property 1820
NDeviceManager.Devices Property 1821
NDeviceManager.DeviceTypes Property 1821
NDeviceManager.NativeType Property 1821
NDeviceManager.PluginManager Property 1821
NDeviceManager Events 1821
NDeviceManager.DeviceAdded Event 1821
NDeviceManager.DeviceRemoved Event 1821
NDeviceManager.DevicesChanged Event 1821
NDeviceManager.DevicesChanging Event 1822
NDeviceManager.DevicesRefreshed Event 1822
NDeviceManagerDeviceEventArgs Class 1822
NDeviceManagerDeviceEventArgs.NDeviceManagerDeviceEventArgs Constructor 1822
NDeviceManagerDeviceEventArgs Properties 1822
NDeviceManagerDeviceEventArgs.Device Property 1822
NDevices Class 1822
NDevices Fields 1823
NDevices.DllName Field 1823
NDevices Properties 1823
NDevices.NativeModule Property 1823
7.2.4.2 Structs, Records, Enums 1823
Neurotec.Devices.NCameraStatus Enumeration 1823
Neurotec.Devices.NDeviceType Enumeration 1824
7.2.5 Neurotec.IO Namespace 1825
7.2.5.1 Classes 1825
NBuffer Class 1825
NBuffer Constructor 1828
NBuffer.NBuffer Constructor (IntPtr, UIntPtr) 1828
NBuffer.NBuffer Constructor (IntPtr, UIntPtr, NMemoryType) 1828

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBuffer.NBuffer Constructor (IntPtr, UIntPtr, bool) 1828
NBuffer.NBuffer Constructor (IntPtr, int) 1829
NBuffer.NBuffer Constructor (IntPtr, int, NMemoryType) 1829
NBuffer.NBuffer Constructor (IntPtr, int, bool) 1829
NBuffer.NBuffer Constructor (IntPtr, long) 1829
NBuffer.NBuffer Constructor (IntPtr, long, NMemoryType) 1829
NBuffer.NBuffer Constructor (IntPtr, long, bool) 1830
NBuffer.NBuffer Constructor (NBuffer, UIntPtr, UIntPtr) 1830
NBuffer.NBuffer Constructor (NBuffer, int, int) 1830
NBuffer.NBuffer Constructor (NBuffer, long, long) 1830
NBuffer.NBuffer Constructor (UIntPtr) 1831
NBuffer.NBuffer Constructor (byte[]) 1831
NBuffer.NBuffer Constructor (byte[], int, int) 1831
NBuffer.NBuffer Constructor (int) 1831
NBuffer.NBuffer Constructor (long) 1831
NBuffer Methods 1832
Copy Method 1832
CopyFrom Method 1833
CopyTo Method 1834
NBuffer.ToArray Method 1836
NBuffer.WriteTo Method 1836
NBuffer Properties 1836
NBuffer.Empty Property 1837
NBuffer.LongSize Property 1837
NBuffer.NativeSize Property 1837
NBuffer.NativeType Property 1837
NBuffer.Ptr Property 1837
NBuffer.Size Property 1837
NMemoryStream Class 1837
NMemoryStream Constructor 1841
NMemoryStream.NMemoryStream Constructor () 1841
NMemoryStream.NMemoryStream Constructor (NBuffer) 1841
NMemoryStream.NMemoryStream Constructor (NBuffer, FileAccess, bool) 1841
NMemoryStream.NMemoryStream Constructor (NBuffer, bool) 1841
NMemoryStream.NMemoryStream Constructor (UIntPtr) 1842
NMemoryStream.NMemoryStream Constructor (byte[]) 1842
NMemoryStream.NMemoryStream Constructor (byte[], bool) 1842
NMemoryStream.NMemoryStream Constructor (int) 1842
NMemoryStream.NMemoryStream Constructor (long) 1842
NMemoryStream Methods 1842
NMemoryStream.GetBuffer Method 1843

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMemoryStream.ToArray Method 1843
NMemoryStream.WriteTo Method 1843
NMemoryStream Properties 1843
NMemoryStream.Capacity Property 1843
NMemoryStream.LongCapacity Property 1843
NMemoryStream.NativeCapacity Property 1843
NMemoryStream.NativeType Property 1843
NMemoryStream.PositionPtr Property 1844
NStream Class 1844
NStream Methods 1847
NStream.AsStream Method 1847
CopyTo Method 1847
NStream.Flush Method 1847
NStream.FromStream Method 1847
Read Method 1848
NStream.ReadByte Method 1848
NStream.Seek Method 1848
NStream.SetLength Method 1849
NStream.Synchronized Method 1849
Write Method 1849
NStream.WriteByte Method 1850
NStream Properties 1850
NStream.CanRead Property 1850
NStream.CanSeek Property 1850
NStream.CanWrite Property 1850
NStream.Length Property 1850
NStream.NativeType Property 1850
NStream.Null Property 1850
NStream.Position Property 1851
7.2.6 Neurotec.Licensing Namespace 1851
7.2.6.1 Classes 1851
NLicense Class 1851
NLicense Methods 1852
NLicense.ActivateOnline Method 1852
DeactivateOnline Method 1852
NLicense.DeactivateOnlineWithId Method 1853
GenerateDeactivationId Method 1853
GenerateDeactivationIdForLicense Method 1854
GenerateId Method 1855
NLicense.GetInfo Method 1855
NLicense.GetInfoForComponent Method 1856

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLicense.IsComponentActivated Method 1856
Obtain Method 1856
ObtainComponents Method 1857
NLicense.Release Method 1858
NLicense.ReleaseComponents Method 1858
NLicense Properties 1858
NLicense.NativeType Property 1858
NLicenseInfo Structure 1858
NLicenseInfo.NLicenseInfo Constructor 1859
NLicenseInfo Properties 1859
NLicenseInfo.DistributorId Property 1859
NLicenseInfo.IsObtained Property 1859
NLicenseInfo.NativeType Property 1859
NLicenseInfo.SerialNumber Property 1859
NLicensing Class 1860
NLicensing Fields 1860
NLicensing.DllName Field 1860
NLicensing Properties 1860
NLicensing.NativeModule Property 1860
NLicensingService Class 1860
NLicensingService Methods 1861
NLicensingService.GetBinPath Method 1861
NLicensingService.GetConfPath Method 1861
NLicensingService.GetStatus Method 1861
Install Method 1861
NLicensingService.IsTrial Method 1862
NLicensingService.Start Method 1862
NLicensingService.Stop Method 1862
NLicensingService.Uninstall Method 1862
NLicensingService Properties 1862
NLicensingService.NativeType Property 1862
NLicensingService.NLicensingServiceStatusNativeType Property 1862
7.2.6.2 Structs, Records, Enums 1862
Neurotec.Licensing.NLicensingServiceStatus Enumeration 1862
7.2.7 Neurotec.Plugins Namespace 1863
7.2.7.1 Classes 1863
NPlugin Class 1863
NPlugin Methods 1866
NPlugin.Disable Method 1866
NPlugin.Enable Method 1867
NPlugin.GetError Method 1867

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPlugin.Plug Method 1867
NPlugin.Unplug Method 1867
NPlugin Properties 1867
NPlugin.FileName Property 1867
NPlugin.LoadTime Property 1867
NPlugin.Module Property 1868
NPlugin.NativeType Property 1868
NPlugin.NPluginStateNativeType Property 1868
NPlugin.Owner Property 1868
NPlugin.PlugTime Property 1868
NPlugin.SelectedInterfaceVersion Property 1868
NPlugin.State Property 1868
NPlugin Events 1869
NPlugin.Changed Event 1869
NPlugin.Changing Event 1869
NPluginManager Class 1869
NPluginManager Classes 1872
NPluginManager.DisabledPluginCollection Class 1872
NPluginManager.PluginCollection Class 1872
NPluginManager Methods 1874
NPluginManager.EnsurePluginSearchPerformed Method 1874
NPluginManager.GetInstances Method 1874
NPluginManager.GetInterfaceVersions Method 1874
NPluginManager.PlugAll Method 1874
NPluginManager.Refresh Method 1874
NPluginManager.UnplugAll Method 1874
NPluginManager Properties 1875
NPluginManager.AllowsUnplug Property 1875
NPluginManager.DisabledPlugins Property 1875
NPluginManager.InterfaceType Property 1875
NPluginManager.NativeType Property 1875
NPluginManager.Plugins Property 1875
NPluginManager.PluginSearchPath Property 1875
NPluginManager Events 1876
NPluginManager.DisabledPluginsChanged Event 1876
NPluginManager.DisabledPluginsChanging Event 1876
NPluginManager.PluginAdded Event 1876
NPluginManager.PluginsChanged Event 1876
NPluginManager.PluginsChanging Event 1876
NPluginManagerPluginEventArgs Class 1876
NPluginManagerPluginEventArgs.NPluginManagerPluginEventArgs Constructor 1877

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPluginManagerPluginEventArgs Properties 1877
NPluginManagerPluginEventArgs.Plugin Property 1877
NPluginModule Class 1877
NPluginModule.NPluginModule Constructor 1880
NPluginModule Methods 1880
NPluginModule.GetInterfaceVersions Method 1881
NPluginModule Properties 1881
NPluginModule.IncompatiblePlugins Property 1881
NPluginModule.InterfaceType Property 1881
NPluginModule.NativeType Property 1881
NPluginModule.PluginName Property 1881
NPluginModule.Priority Property 1882
7.2.7.2 Structs, Records, Enums 1882
Neurotec.Plugins.NPluginState Enumeration 1882
7.2.8 Neurotec.Surveillance Namespace 1883
7.2.8.1 Classes 1884
NsedMatch Structure 1884
NsedMatch Properties 1884
NsedMatch.NativeType Property 1884
NsedMatch.Score Property 1884
NsedMatch.TemplateId Property 1884
NSurveillance Class 1885
NSurveillance.NSurveillance Constructor 1890
NSurveillance Fields 1890
NSurveillance.DllName Field 1891
NSurveillance.ParameterDetectionDetectMoreFaces Field 1891
NSurveillance.ParameterDetectionFaceConfidenceThreshold Field 1891
NSurveillance.ParameterDetectionFaceDetectionScaleCount Field 1891
NSurveillance.ParameterDetectionFaceRollAngleBase Field 1891
NSurveillance.ParameterDetectionMaxIod Field 1891
NSurveillance.ParameterDetectionMaxRollAngleDeviation Field 1891
NSurveillance.ParameterDetectionMaxYawAngleDeviation Field 1892
NSurveillance.ParameterDetectionMinIod Field 1892
NSurveillance.ParameterExtractionBlinkConfidenceThreshold Field 1892
NSurveillance.ParameterExtractionDarkGlassesConfidenceThreshold Field 1892
NSurveillance.ParameterExtractionDetectBlink Field 1892
NSurveillance.ParameterExtractionDetectDarkGlasses Field 1892
NSurveillance.ParameterExtractionDetectExpression Field 1892
NSurveillance.ParameterExtractionDetectGender Field 1892
NSurveillance.ParameterExtractionDetectGlasses Field 1893
NSurveillance.ParameterExtractionDetectMouthOpen Field 1893

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSurveillance.ParameterExtractionEnrollmentTemplateSize Field 1893
NSurveillance.ParameterExtractionExpressionConfidenceThreshold Field 1893
NSurveillance.ParameterExtractionFaceQualityThreshold Field 1893
NSurveillance.ParameterExtractionGenderConfidenceThreshold Field 1893
NSurveillance.ParameterExtractionGlassesConfidenceThreshold Field 1894
NSurveillance.ParameterExtractionIdentificationTemplateSize Field 1894
NSurveillance.ParameterExtractionMaxRecordsPerTemplate Field 1894
NSurveillance.ParameterExtractionMouthOpenConfidenceThreshold Field 1894
NSurveillance.ParameterIdentificationMatchingThreshold Field 1894
NSurveillance.ParameterIdentificationMaxBestMatchCount Field 1895
NSurveillance.ParameterMaxThreadCount Field 1895
NSurveillance Methods 1895
NSurveillance.AddEvent Method 1895
AddTemplate Method 1895
NSurveillance.GetTemplateIds Method 1896
NSurveillance.RemoveAllTemplates Method 1896
NSurveillance.RemoveEvent Method 1896
NSurveillance.RemoveTemplate Method 1896
NSurveillance.Start Method 1896
NSurveillance.Stop Method 1897
NSurveillance Properties 1897
NSurveillance.DetectionDetectMoreFaces Property 1897
NSurveillance.DetectionFaceConfidenceThreshold Property 1897
NSurveillance.DetectionFaceDetectionScaleCount Property 1897
NSurveillance.DetectionFaceRollAngleBase Property 1897
NSurveillance.DetectionMaxIod Property 1897
NSurveillance.DetectionMaxRollAngleDeviation Property 1897
NSurveillance.DetectionMaxYawAngleDeviation Property 1898
NSurveillance.DetectionMinIod Property 1898
NSurveillance.DeviceManager Property 1898
NSurveillance.ExtractionBlinkConfidenceThreshold Property 1898
NSurveillance.ExtractionDarkGlassesConfidenceThreshold Property 1898
NSurveillance.ExtractionDetectBlink Property 1898
NSurveillance.ExtractionDetectDarkGlasses Property 1899
NSurveillance.ExtractionDetectExpression Property 1899
NSurveillance.ExtractionDetectGender Property 1899
NSurveillance.ExtractionDetectGlasses Property 1899
NSurveillance.ExtractionDetectMouthOpen Property 1899
NSurveillance.ExtractionEnrollmentTemplateSize Property 1899
NSurveillance.ExtractionExpressionConfidenceThreshold Property 1900
NSurveillance.ExtractionFaceQualityThreshold Property 1900

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSurveillance.ExtractionGenderConfidenceThreshold Property 1900
NSurveillance.ExtractionGlassesConfidenceThreshold Property 1900
NSurveillance.ExtractionIdentificationTemplateSize Property 1900
NSurveillance.ExtractionMaxRecordsPerTemplate Property 1900
NSurveillance.ExtractionMouthOpenConfidenceThreshold Property 1901
NSurveillance.IdentificationMatchingThreshold Property 1901
NSurveillance.IdentificationMaxBestMatchCount Property 1901
NSurveillance.IsRunning Property 1901
NSurveillance.MaxThreadCount Property 1901
NSurveillance.NativeType Property 1901
NSurveillance Events 1901
NSurveillance.Any Event 1901
NSurveillance.FacesDetected Event 1901
NSurveillance.ImageCaptured Event 1902
NSurveillance.IsRunningChanged Event 1902
NSurveillance.SubjectAppeared Event 1902
NSurveillance.SubjectDisappeared Event 1902
NSurveillance.SubjectTrack Event 1902
NSurveillanceEventArgs Class 1902
NSurveillanceEventArgs Properties 1902
NSurveillanceEventArgs.EventDetails Property 1902
NSurveillanceEventArgs.EventType Property 1903
NSurveillanceEventDetails Class 1903
NSurveillanceEventDetails Classes 1906
NSurveillanceEventDetails.BestMatchCollection Class 1906
NSurveillanceEventDetails Methods 1906
NSurveillanceEventDetails.GetFaces Method 1906
NSurveillanceEventDetails.GetSubjectTemplate Method 1906
NSurveillanceEventDetails Properties 1906
NSurveillanceEventDetails.BestMatches Property 1906
NSurveillanceEventDetails.DetectionDetails Property 1906
NSurveillanceEventDetails.ExtractionStatus Property 1907
NSurveillanceEventDetails.FrameIndex Property 1907
NSurveillanceEventDetails.GrayscaleImage Property 1907
NSurveillanceEventDetails.NativeType Property 1907
NSurveillanceEventDetails.OriginalImage Property 1907
NSurveillanceEventDetails.Owner Property 1907
NSurveillanceEventDetails.SourceIndex Property 1907
NSurveillanceEventDetails.Template Property 1907
NSurveillanceEventDetails.TimeStamp Property 1908
NSurveillanceEventDetails.TraceIndex Property 1908

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.8.2 Structs, Records, Enums 1908
Neurotec.Surveillance.NSurveillanceEventType Enumeration 1908
7.2.9 Functions 1909
7.2.9.1 NLicensingServiceGetBinPath Function 1909
7.2.9.2 NLicensingServiceGetConfPath Function 1909
7.2.9.3 NLicensingServiceGetStatus Function 1910
7.2.9.4 NLicensingServiceInstall Function 1910
7.2.9.5 NLicensingServiceIsTrial Function 1910
7.2.9.6 NLicensingServiceStart Function 1911
7.2.9.7 NLicensingServiceStop Function 1911
7.2.9.8 NLicensingServiceUninstall Function 1911
7.2.10 Structs, Records, Enums 1911
7.2.10.1 NLicensingServiceStatus Enumeration 1911
7.2.11 Macros 1912
7.2.11.1 NLicensingServiceInstall Macro 1912

7.3 Change Log 1912
7.3.1 C API Reference 1913
7.3.1.1 NCore Library 1913
7.3.1.2 NLicensing Library 1915
7.3.1.3 NSurveillance Library 1915
7.3.2 .NET API Reference 1916
7.3.2.1 Neurotec Library 1916
7.3.2.2 Neurotec.Biometrics.Gui 1918
7.3.2.3 Neurotec.Licensing 1918
7.3.2.4 Neurotec.Surveillance Changes 1918

8 Support 1920

Index a

Copyright © 2009-2014 Neurotechnology. All rights reserved.

1 Introduction

VeriLook Surveillance 2.3 SDK is a software development kit (SDK) for biometric face identification using live video streams from high-resolution digital surveillance cameras. *VeriLook Surveillance 2.3 SDK* is intended for real-time identification of faces from high-resolution digital surveillance cameras, which can be mounted in airports, retail stores, on the street or in other locations where passers-by may not be looking into the camera. *VeriLook Surveillance 2.3 SDK* can also be used for easy time-attendance control applications when a camera is mounted near an entrance, enabling fast, automatic counting and/or recognition of individuals as they pass through.

VeriLook Surveillance 2.3 SDK enables processing face images in real-time video streams and matches them against a internal database – such as a watch-list of crime suspects or a list of authorized and un-authorized personnel. Once a face is detected, it is tracked in all successive frames of video until it disappears from camera view. Face appearance, tracking and disappearance events are reported to the *VeriLook Surveillance 2.3 SDK*. The face tracking in *VeriLook Surveillance 2.3 SDK* also enables automatic enrollment of faces from a video source and can add new biometric facial templates to the internal database or watch-list on the fly.

The basic functionality of *VeriLook Surveillance 2.3 SDK*:

- Capture images from live video stream
- Detect all near-frontal faces
- Extract facial features into facial templates
- Identify in real time

VeriLook Surveillance 2.3 SDK has live video stream analysis which detects number of specific events and triggers or integrator defined callbacks to inform the system about their occurrences. The following events are implemented:

- Face(s) detected
- New face appeared
- Face is tracked from previous frames
- Tracked face disappeared

Each event contains enough information for detailed logging operations, including: processed image from the stream, time stamp with exact image capturing date and time, coordinates of detected faces, formed facial features template, real-time identification results, etc.

1.1 Supported Platforms

VeriLook Surveillance 2.3 SDK can be used in a number of compilers under any of these operating systems:

- Microsoft Windows 2000/XP/2003/Vista/7
- Linux (based on glibc 2.5 or compatible)

VeriLook Surveillance 2.3 SDK supports platforms based on x86 and x86_64 processors architectures. Libraries for Windows and Linux operating systems are provided.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

1.2 System Requirements

Recommended requirements for system

- At least Intel Core 2 Quad CPU running on 2.66 Ghz or equivalent multi-core processor from other manufacturers
- At least 1 GB RAM
- Microsoft Windows specific:
  - Microsoft Windows 2000/XP/2003/Vista/7/8*
  - .NET framework 2.0
  - Microsoft DirectX 9.0 or later

*** Note**: Windows 8 UI (Windows Store) applications are not supported.

- Linux specific:
  - Linux (based on glibc 2.5 or newer)
  - Video4linux
- One high resolution surveillance camera (about 1 Megapixel)

1.3 Features

- **Real time performance**. *VeriLook Surveillance 2.3 SDK* provides processing face images in real-time video streams and matching them against a internal database – such as a watch-list of crime suspects or a list of authorized and un-authorized personnel. Once a face is detected, it can be tracked in all successive frames of video until it disappears from camera view. Face appearance, tracking and disappearance events are reported to the system. The face tracking in *VeriLook Surveillance 2.3 SDK* also enables automatic enrollment of faces from a video source and can add new biometric facial templates to the internal database or watch-list on the fly.
- **Multiple face tracking**. Once detected, the faces are tracked in all successive frames from the video source until they disappear from camera field of view.
- **Multi-core multi-processor systems supported**. *VeriLook Surveillance 2.3 SDK* enables real-time face detection, extraction and matching by providing embedded parallelization of all *VeriLook 4.0 SDK* functions for improved performance on multi-core, multi-processor systems with up to eight logical processors.
- **Flexibility by adding new cameras**. Support for additional cameras can be requested.
- **Support for different programming languages**. *VeriLook Surveillance 2.3 SDK* supports a wide variety of programming languages and works with a range of high-resolution digital surveillance cameras, including Prosilica [19] GigE Vision cameras.
- **Support for different operating systems**. *VeriLook Surveillance 2.3 SDK* enables the development of fast, reliable and cost-effective biometric facial identification systems for Windows and Linux platforms.
- **VeriLook SDK API support**. *VeriLook Surveillance 2.3 SDK* has all the functionality of VeriLook SDK through VeriLook API.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

1.4 Licensing

To develop a product based on *VeriLook Surveillance 2.3 SDK* technology, an integrator should obtain *VeriLook Surveillance 2.3 SDK*. Integrators can develop only an end-user product using *VeriLook Surveillance 2.3 SDK* and sell/install the product to their own customers.

For more information about licensing visit http://www.neurotechnology.com/licensing_model.html.

Also you can refer to *VeriLook Surveillance 2.3 SDK* Software License Agreement (bin\Documentation\license.html) for all licensing terms and conditions.

1.4.1 Single Computer License

A single computer license allows to install Neurotechnology components on one computer and use it by single CPU (a processor can have any number of cores). Component license will not be lost if computer is reinstalled.

The following license management options are available:

- license activation online by communicating with Neurotechnology's server
- license activation by email
- license activation using volume license manager [3]
- license management using volume license manager [3] on LAN or Internet

Notes

Single computer licenses, provided as serial numbers can not be used on virtual machines.

1.4.2 Enterprise Licensing

Neurotechnology products enterprise license allows an **unlimited use** of Neurotechnology components (for example Extractor or Matcher) in the end-user products within the certain territory, market segment or project. These limitations would be included in the licensing agreement.

The enterprise license price depends on the application size and the number of potential application's users within the designated territory, market segment or project. Neurotechnology products enterprise licenses are provided only for big projects.

1.4.3 Volume License Manager

Volume license manager is used on site by integrators or end users to manage obtained licenses for SDK components. It consists of license management software and a dongle, which is used to store the number of obtained licenses. An integrator or an end-user can use the volume license manager in the following ways:

- **Activating the single computer licenses**. An installation license for a SDK component will be activated for using on a particular computer. The license quantity for the SDK component in the license manager will be decreased by the amount of activated licenses.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- **Managing the single computer licenses on LAN or Internet**. The license manager allows to manage installation licenses for SDK components across the computers on LAN or Internet. The number of managed licenses for a SDK component is limited by the number of licenses in the license manager. No license activation is needed and the license quantity is not decreased. Once issued, the license is assigned to certain computer on the network.
- **Using a license manager as a dongle**. The volume license manager containing at least one license for a SDK component can be used as a dongle that allows to run SDK component installation on a particular computer.

Additional SDK component installation licenses for the license manager can be purchased anytime. Neurotechnology will generate a special update file and send it to you. Then you should enter file to the license manager to add purchased licenses.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

2 What's new

Version 2.3.0.7

- Fixes bugs in NCore [257] library.
- Fixed samples and tutorials.

Version 2.3.0.6

- Fixed rare internal bugs in NCore [257].
- Added support for RTSP cameras that do not support GET PARAMETER RTSP method.

Version 2.3.0.5

- Fixed rare internal bugs in NCore [257].
- Fixed small bugs in samples and tutorials.

Version 2.3.0.4

- Fixed rare internal bugs in NCore [257].
- Fixed memory leak in NLExtractor [104].
- Fixed bug in LicenseManager application, dongle licenses are now displayed correctly.
- Fixed rare bugs in license obtaining over network.

Version 2.3.0.3

- Fixed rare internal bugs in NCore [257].
- Added support for RTSP camera formats.

Version 2.3.0.2

- Fixed SDK redistributable file copy problem (dotNET_Portable folder copying).
- Fixed NLicensing GetInfo function mulfunction (returning error on every call).
- Fixed access violation error when setting invalid thread count to Surveillance object.
- Added RTSP support in Windows 8 Windows Store applications.
- Added new methods to access dongle licenses to NLicenseManager API.

Version 2.3.0.1

- Added support for Nikon cameras.
- Fixed callback crash under certain conditions.
- Fixed NLicensing.ObtainComponents IOException in some Windows 8 applications.
- Fixed SurveillanceSampleWX crash after detecting newly enrolled image.

Version 2.3.0.0

- Fixed rare internal bugs in NCore [257], NBiometrics [60] and NBiometricStandards library.
- Added new face properties detection: glasses, dark glasses, blink (closed eyes), open mouth.
- Added new face expression detection: smile.

Version 2.2.0.5

- Fixed memory leak in NBiometricStandards library.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- Fixed rare internal bugs in NCore [257], NBiometrics [60] and NBiometricStandards library.

Version 2.2.0.4

- Fixed rare internal bugs in NCore [257], NBiometrics [60] library.
- Fixed rare internal bugs in NBiometrics [60] library.
- Fixed several minor bugs in .NET [1468] library wrappers.
- Fixed several bugs in Surveillance samples and tutorials.
- Updated support of RTSP cameras (RTSP cameras are being used throught live555 library).
- Added .NET and .NET portable libraries in `Bin\dotNET` (SDK API wrappers for .NET framework 2.0 and above) and `Bin\dotNET_Portable` (SDK API wrappers for Windows Store application, .NET framework 4.5) directories. Additional NMedia .NET wrappers with additional functionality:
  - `Neurotec.Media_FX35.dll`; `Neurotec.Media_FX45.dll` - Neurotec.Media extensions for .NET 3.5 and 4.5 frameworks.
  - `Neurotec.Media_Win8.dll` - Neurotec.Media extension for Windows 8 and Windows 8 Store applications.
- Updated NImage [1232] API, NImageInfo is now part of NImage [1232].
- Added support for Canon EOS family cameras and updated camera API (see DevicesSample for example).
- Added ability to change device properties and parameters throught NDeviceManager (see DevicesSample [55] for example).
- Added ability to create add network devices (RTSP cameras) to NDeviceManager without specifying connection settings in config file (see DevicesSample [55] for example).

Version 2.2.0.3

- Fixed rare ActivationWizard bug in showing badly formatted diagnostics.
- Fixed rare internal bugs in NCore [257] library.
- Added new NMedia tutorials, which loading of additional information for image.

Version 2.2.0.2

- Fixed license deactivation in Activation Wizard.
- Fixed Surveillance sample (WX, C#, VB.NET) settings error handling.
- Fixed memory leaks in Surveillance Sample (C#, VB.NET).
- Fixed capturing problems (failure to start and select available formats) with web camera on MacOSX.
- Fixed Mobotix [19] IP camera caputring problem - low frames per second count.
- Fixed failure to close communication channel with RTSP cameras.

Version 2.2.0.1

- New face image sharpness quality measure.
- Updated gender classification algorithm, improved accuracy.
- Fixed internal bugs in NCore [257] and NBiometrics library.
- Fixed bugs in Activation Wizard.

Version 2.2.0.0

- Added new Activation Wizard.
- Added new NLicensing library functions [1195] (former NInstaller API).
- Added support for Basler scA1000-30gm and Basler BIP2-1600-25c-DN IP cameras [18].
- Added gender classification.
- Removed obsolete NInstaller API.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Version 2.1.0.8

- Fixed bug with RTSP cameras loosing connection after 20 seconds of capturing.
- Fixed bug with RTSP cameras failing to start capturing after cameras where previously stopped.
- Added new functions to NLicensing library [1179]. New functions allow to generate ids, activate licenses online and deactivate them if needed. New functions will make integrating licensing to client applications easier, see Licensing tutorials [48] for more information.
- NDeviceManagerGetDevice [1172] function an other NDeviceManager and NPluginManager [607] related functions in C API have been changed to return objects with incremented reference count to be thread-safe in general case. NObjectUnref [560] shall be called now on returned objects when they are no longer used to prevent memory leaks. Ex versions for other object returning functions have been added.

Version 2.1.0.7

- Fixed internal bugs in NCore [257] library.
- Fixed multiple ip cameras usage bug in NMedia library.
- Fixed minor memory leaks in Surveillance library.

Version 2.1.0.6

- Fixed rare memory access violation crash on Surveillance exit.

Version 2.1.0.5

- Fixed minor bugs in Surveillance library.
- Fixed license activation bug in LicenseManager API.

Version 2.1.0.4

- Fixed license activation bug in Installer API.
- Updated and optimized DirectShow device enumeration for NMedia library.
- Updated NCore [257] library internals.

Version 2.1.0.3

- Fixed minor bugs in NBiometrics library and other libraries.

Version 2.1.0.2

- Fixed Media library bug - unable to save WSQ image to memory.
- Fixed number of minor Core library bugs.

Version 2.1.0.1

- Fixed Core library bug - not launching under Windows XP.

Version 2.1.0.0

- New VeriLook algorithm for face recognition.

Version 2.0.0.2

- Fixed minor internal bugs.
- Fixed minor stability issues resolved.

Version 2.0.0.1

- Fixed Linux algorithm demo.
- Fixed Surveillance WxWidgets sample bugs.
- Fixed memory leaks and stability issues resolved.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- Fixed MaxTheadCount parameter setting.
- Smoother face detection/identification.
- Enhanced face tracking performance.

Version 2.0.0.0

- Added support for multiple cameras.
- Enhanced subjects tracking.
- CameraMan [1138] library was changed to NDevices [1178] library.
- Added support for enrollment from images in sample application.
- Added support for VeriLook Surveillance internal database synchronization in C# Sample.
- Lower memory usage, higher frame rate, better performance.

Version 1.2.0.0

- Support for adding custom devices [21].
- New NDevice library [1124] for devices management.
- New NMedia library [1196]: former NImages library now is the part of NMedia library.
- New NBiomentrics library [60]: former NExtractors, NMatchers and NTemplates libraries now are the part of this library.
- New NdmSample [55] application which demonstrates how to add custom devices.

Version 1.1.0.3

- Added support for Axis M1114 network camera.

Version 1.1.0.2

- NExtractors library changes:
  - Added support for extractors base license.
  - Face pose estimation algorithm improved.
  - Obsolete defines, functions and members removed.
- NMatchers library changes:
  - Matching of empty NLRecord [134] fixed.
- NImages library changes:
  - Fixed WSQ and IHead [1201] multi-thread issues.
  - Reading of some JP2 files with alpha channel fixed.

Version 1.1.0.1

- Added support for Pixord N606 network cameras.

Version 1.1.0.0

- Added support for Mobotix [19] network cameras.
- Integrated new faces recognition algorithm (the same as in VeriLook 4.0). New algorithm capabilities:
  - Recognition of faces which are rotated in-plane (rolled/tilted left or right) at any angle.
  - Recognition accuracy and speed improved.
  - Matching of different size templates.
- Added support for video files [1439].
- More than one NSurveillance [1439] object can be created. Created Surveillance objects can work with several cameras simultaneously.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- Some functions from NSurveillance [1439] library and methods from Neurotec.Surveillance [1883] became depreciated. See NSurveillance library [1440] form more information.

Version 1.0.0.5

- **ADD**: Support for Cisco [18] IP 4500 network camera.

Version 1.0.0.4

- SDK structure was changed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

3 Overview

3.1 Required Libraries

In order to use VeriLook Surveillance 2.3 SDK these libraries are required[1]:

- NBiometrics [60] (`NBiometrics.dll` file)
- NCore [257] (`NCore.dll` [257] file)
- NDevices [1124] (`NDevices.dll` file)
- NLicensing [1179] (`NLicensing.dll` file)
- NMedia [1196] (`NMedia.dll` file)
- NSurveillance [1439] (`NSurveillance.dll` file)

Also wrappers libraries for .Net are provided[2]:

- Neurotec.Biometrics [1654] (`Neurotec.Biometrics.dll` file)
- Neurotec [1469] (`Neurotec.dll` [1469] file)
- Neurotec.Devices [1792] (`Neurotec.Devices.dll` file)
- Neurotec.Licensing [1851] (`Neurotec.Licensing.dll` [1851] file)
- Neurotec.Media (`Neurotec.Media.dll` file)
- Neurotec.Surveillance [1883] (`Neurotec.Surveillance.dll` [1883] file)

1 - These libraries are for Windows OS only and are located within bin\Win32_x86 and bin\Win64_x64 folders of SDK. Also Lib files are provided and they are located within \lib folder of the SDK. Shared objects for Linux OS are located within *lib\linux_x86* and *lib\linux_x86_64* folders.

2 - These libraries are located within *bin\dotNET* folder of the SDK.

3.2 Image Support

Image support can be divided into the following four parts:

- Image [11]. The base of all image support. Developers should start using this part and take advantage of other parts if it is required.
- Image Format [12]. Declares the supported image formats. Shows how to load and save images in a format-neutral way.
- Image File [13]. Should be used if multiple images are stored in one file and more than one image should be loaded from the file.
- Low-Level Image Input-Output [13]. Should be used to have more control on how images are loaded and saved in particular format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

3.2.1 Image

Image is a rectangular area of pixels (image elements), defined by width, height and pixel format.

Pixel format describes type of color information contained in the image like monochrome, grayscale, true color or palette-based (indexed) and describes pixels storage in memory (how many bits are required to store one pixel).

Image is defined by `HNImage` [1276] handle in `NImage` [1232] module (`NImage` [1232] class in .NET). It is an encapsulation of a memory block that stores image pixels. The memory block is organized as rows that follow each other in top-to-bottom order. The number of rows is equal to height of image. Each row is organized as pixels that follow each other in left-to-right order. The number of pixels in a row is equal to width of image. A pixel format describes how image pixels are stored. See `NImageGetWidth` [1267], `NImageGetHeight` [1257], `NImageGetStride` [1263], `NImageGetPixelFormat` [1277] and `NImageGetPixels` [1289] functions (`Width`, `Height`, `Stride`, `PixelFormat` and `Pixels` properties in .NET) in API Reference for more information.

An image can have horizontal and vertical resolution attributes assigned to it if they are applicable (they are required for fingerprint image, and do not make sense for face image). See `NImageGetHorzResolution` [1259] and `NImageGetVertResolution` [1265] functions (`HorzResolution` and `VertResolution` properties in .NET) in API Reference for more information.

An image can be created either as empty or from existing memory block. See `NImageCreate` [1247], `NImageCreateFromData` [1249] and `NImageCreateWrapper` [1261] functions (`Create`, `FromData` and `GetWrapper` methods in .NET) for more information.

For each value of `NPixelFormat` [1369] (`NPixelFormat` [1369] in .NET) exposed via interface a module (subclass of `NImage` [1232] in .NET) is provided for managing according type of image (getting and setting individual pixels, etc.). See `NGrayscaleImage` [1228], `NMonochromeImage` [1355] and `NRgbImage` [1392] modules (`NGrayscaleImage` [1228], `NMonochromeImage` [1355] and `NRgbImage` [1392] classes in .NET) for more information.

An image can be converted to different pixel format using `NImageCreateFromImage` [1287] function (`FromImage` method in .NET).

Different methods should be used to display an image on different platforms:

- On Windows `BmpSaveImageToHBitmap` [1200] function (`ToHBitmap` method in .NET) can be used to receive a standard Win32 HBITMAP for the image. The reverse process is also possible using `BmpLoadImageFromHBitmap` [1200] function (`FromHBitmap` method in .NET).
- In .NET `ToBitmap` method can be used to receive a standard .NET Bitmap. The reverse process is also possible using `FromBitmap` method.
- On Linux there is no easy method implemented. However, a memory block containing pixels of image could be accessed via `NImageGetPixelFormat` [1277] function (`PixelFormat` method in .NET). The memory block can be used to display the image or convert it to some other representation on any platform.

An image can be stored in file in any supported image format using `NImageSaveToFile` function (`Save` method in .NET).

An image stored in file in any supported image format can be loaded using `NImageCreateFromFile` function (`FromFile` method in .NET).

Files containing more than one image are also supported. See Image File [13] and Image Format [12] sections for more information.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

3.2.2 Image Format

Image format is a specification of image storage in a file. The specification may require to compress/decompress image during writing/reading it to/from a file.

Image format is defined by `HNImageFormat` [1324] handle in `NImageFormat` [1299] module (`NImageFormat` [1299] class in .NET).

There is a number of image formats supported in the *VeriLook Surveillance 2.3 SDK*. Certain formats could not be read from and written to a file on all platforms. See the following table for details.

Image Format	Can read	Can write
BMP	Yes	Yes
NIST IHead [1201]	Yes[2)]	Yes[2)]
JPEG	Yes	Yes
Lossless JPEG	Yes[2)3)4)5)]	Yes[2)3)4)5)]
JPEG 2000	Yes[2)3)4)5)]	Yes[2)3)4)5)]
PNG	Yes	Yes
TIFF	Yes	No
WSQ	Yes[1)]	Yes[1)]

Note: Special formats marked with superscripts in the table are only supported by special components:

- [1)] Fingerprint WSQ component
- [2)] Fingerprint BSS component
- [3)] Face BSS component
- [4)] Iris BSS component
- [5)] Palm Print Client component

Image formats from the table are accessible using these functions:

- `NImageFormatGetBmp` [1303]
- `NImageFormatGetIHead` [1306]
- `NImageFormatGetTiff` [1311]
- `NImageFormatGetWsq` [1311]

For .NET read-only fields `Bmp` [1196], `Gif`, `IHead` [1201], `Jpeg` [1205], `Png` [1423], `Tiff` [1429] and `Wsq` [1431] are used.

To find out which images formats are supported in the VeriLook Surveillance 2.3 SDK in version-independent way these functions should be used:

- `NImageFormatGetFormatCount` [1306]
- `NImageFormatGetFormat` [1304]

Name, file name pattern (file filter) and default file extension of the image format can be retrieved using `NImageFormatGetName`, `NImageFormatGetFileFilter` and `NImageFormatGetDefaultFileExtension` functions (`Name`, `FileFilter` and `DefaultFileExtension` properties in .NET).

To find out which image format should be used to read or write a particular file `NImageFormatSelect` [1321] function

Copyright © 2009-2014 Neurotechnology. All rights reserved.

(Select method in .NET) should be used.

An image can be loaded and saved from/to file or memory buffer using these functions:

- NImageFormatLoadImageFromFile [1312]
- NImageFormatLoadImageFromMemory [1314]
- NImageFormatSaveImageToFile [1319]
- NImageFormatSaveImageToMemory [1320]

(LoadImage and SaveImage methods for .NET). Note that not all image formats support both reading and writing. Use NImageFormatCanRead [1301] and/or NImageFormatCanWrite [1302] function(s) (CanRead and/or CanWrite property(ies) in .NET) to check if the particular image format does.

If image file contains more than one image then image file can be opened using NImageFormatOpenFile [1310] or NImageFormatOpenFileFromMemory [1317] function (OpenFile method in .NET). Image file further can be used to read all images from the file.

If multiple images should be saved in one file NImageFormatSaveImagesToFile [1313] function (SaveImages method in .NET) should be used. Note that not all image formats support writing of multiple images. Use NImageFormatCanWriteMultiple [1302] function (CanWriteMultiple property in .NET) to check if the particular image format does.

3.2.3 Image File

Image file is defined by HNImageFile [1298] handle in NImageFile [1294] module (NImageFile [1294] class in .NET). HNImageFile [1298] handle is an encapsulation of an opened read-only file containing one or more images.

An image file is opened using NImageFileCreate [1295] function (FromFile method in .NET). Also an image file can be opened using image format.

Images are read from image file subsequently calling NImageFileReadImage [1297] function (ReadImage method in .NET) until HNImage [1276] (NImage [1232] in .NET) handle returns NULL [1075] (null in .NET).

3.2.4 Low-Level Image Input-Output

Low-level image I/O is implemented in Bmp [1196], IHead [1201], Tiff [1429] and Wsq [1431] modules (Bmp [1196], IHead [1201], Tiff [1429] and Wsq [1431] classes in .NET).

These modules (classes in .NET) provides functions (static methods in .NET) for loading and saving images in according format (BMP, NIST IHead [1201], TIFF and WSQ).

Those functions (static methods in .NET) can take parameters that precisely control loading and saving of the image in particular formats. For example, bit rate is specified when saving in WSQ format.

3.3 Matching Threshold and FAR/FRR

Biometric features matching algorithm provides similarity score as a result. The higher is score, the higher is probability that features collections are obtained from the same person.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

The strictness of matching algorithms is controlled by NMP_MATCHING_THRESHOLD [195] (NMatcher.MatchingThreshold [1756]) parameter of NMatcher [171]. The higher the threshold value, the more similar feature collections will have to be to yield positive result during matching.

Matching threshold - the minimum score that verification and identification functions accept to assume that the compared finger fingerprints, face or iris belong to the same person.

Matching threshold is linked to false acceptance rate (FAR, different subjects erroneously accepted as of the same) of matching algorithm. The higher is threshold, the lower is FAR and higher FRR (false rejection rate, same subjects erroneously accepted as different) and vice a versa.

Matching threshold for NMatcher [171] should be determined from this table (**note**: thresholds for MINEX mode are provided at the end):

FAR (false acceptance rate)	Matching threshold (score)
100 %	0
10 %	12
1 %	24
0.1 %	36
0.01 %	48
0.001 %	60
0.0001 %	72
0.00001 %	84
0.000001 %	96

or using this formula:

```
Threshold = -12 * log10(FAR); where FAR is NOT percentage value (e.g. 0.1% FAR is 0.001)
```

Matching threshold should be selected according to desired FAR (False Acceptance Rate). FAR is calculated for single match (1:1) and during identification (1:N) false acceptance accumulates. Identification false acceptance probability can be calculated using this formula:

```
(1-(1-FAR/100)^N)*100, where N - DB size
```

For example:

If FAR=0.001% then probability that false acceptance situation will occur during 1:N identification (where N=10 000) is 1-(1-0.00001)^10000=9.52%.

If FAR=0.0001% then probability that false acceptance situation will occur during 1:N identification (where N=10 000) is 1-(1-0.000001)^10000=1.00%.

Matching threshold/FAR should be selected according to the system's development requirements and taking into account mentioned identification false acceptance accumulation.

Matching threshold for NMatcher [171] when MINEX mode is used:

FAR (false acceptance rate)	MINEX matching threshold (score)
10 %	100
1 %	150
0.1 %	200
0.01 %	250
0.001 %	300
0.0001 %	350

Copyright © 2009-2014 Neurotechnology. All rights reserved.

0.00001 %	400

Notes

It is recommended to use NMatcher [171] instead of using separate matchers like NlsMatcher [143], NLMatcher [128], NfsMatcher, NFMatcher, NesMatcher.

3.4 Recommendations and Constraints

Face recognition accuracy of VeriLook Surveillance 2.3 SDK heavily depends on the quality of a face image. **Image quality during enrollment is important**, as it influences the quality of the face template.

There are some basic recommendations and constraints when using face recognition applications based on VeriLook Surveillance SDK.

Cameras and images

- **Similar quality cameras** are recommended for both enrollment and identification. Using the same camera model is even better. Enrollment from photo is also possible if it conforms to other requirements on this page.
- **40 pixels is the recommended minimal distance between eyes** for a face on image or video stream to perform face template extraction. **75 pixels or more** recommended for better face recognition results. Note that this distance should be **native**, not achieved by resizing an image.
- **1 MegaPixel** or better camera resolution is recommended for face enrollment and recognition. Make sure that **native** resolution is provided by a camera, as some cameras or webcams may **scale up** native images to higher resolution without image quality improvement.
- **Check for mirrored face images**, as recognition will fail if a face was enrolled from a mirrored image, and later a non-mirrored face image is used for recognition (or vice versa). This happens as some cameras can be configured to produce mirrored images or may even produce them by default, and different cameras or configurations may be used during enrollment and identification. We recommend to use face images with uniform orientation – all images within a system should be either native or mirrored, but not mixed between each other.
- **Use several images during enrollment**, as it improves facial template quality which results in improvement of recognition quality and reliability.

Lighting

Controlled lighting conditions are recommended:

- **Direct frontal or diffused light** allows equal lighting distribution on each side of the face and from top to bottom with no significant shadows within the face region.
- **Avoid glares** on face skin or glasses that are produced by some types of illumination.

Face posture

The face recognition engine has certain tolerance to face posture:

- head **roll** (tilt) – ±180 degrees (configurable); **±15 degrees** recommended as it is the fastest setting which is usually sufficient for most near-frontal face images.
- head **pitch** (nod) – ±15 degrees from frontal position.
- head **yaw** (bobble) – ±45 degrees from frontal position.
  - **±15 degrees default** value is the fastest setting which is usually sufficient for most near-frontal face images.
  - **30 degrees difference** between a face template in a database and a face image from camera is acceptable.
  - Several views of the same face can be enrolled to the database to cover the whole ±45 degrees yaw range from frontal position.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

The head pitch and yaw tolerance can increase up to ±25 degrees if several images were used during enrollment and these images covered slightly different face views from corresponding directions.

Face quality recommendations for enrollment

- **Neutral face expression** during enrollment is recommended, as non-neutral face expression may affect the accuracy of recognition.
- **Several images with different appearance variants** are recommended for enrollment to assure the quality of recognition in the situations when part of face is covered with glasses or hair:
  - **Eyeglasses** – separate enrollments with and without glasses will assure the best recognition quality for both cases.
  - **Hair style** – some hair styles may cover parts of face, thus hairpins or other means of holding hair off the face are recommended during enrollment.
  - **Facial hair** style changes may require additional enrollments, especially when beard or moustache is grown or shaved off.

Memory and Performance Constraints During Face Tracking

- **Memory usage**. VeriLook Surveillance consumes about 10 MB of memory per minute when tracking one face at a speed of 10 frames per second. The consumed memory is released after the face disappears from a frame.
- **Multiple faces in a frame**. If multiple faces are visible in a frame, tracking performance falls down.
- **Minimal frame rate**. It is recommended to retrieve at least 10 frames per second from a camera. If less than 10 frames are captured, face tracking feature may be not available.

3.5 Cameras

Images should be enrolled and matched using the same camera, as devices have different optical distortions that can influence face recognition performance.

VeriLook Surveillance 2.3 SDK works best with a range of high-resolution digital surveillance cameras. Cameras which have DirectShow support can be used with VeriLook Surveillance 2.3 SDK:

1. **IP cameras**. See the list of supported IP cameras [17].
2. **Web cameras** (at least VGA resolution cameras)
3. **Other cameras**, which support *DirectShow*. For example, these cameras have *DirectShow* support and can be used with *VeriLook Surveillance 2.3 SDK*:
   1. The Imaging Source USB and FireWare cameras
   2. PixeLink FireWire and USB cameras (except GigE)
   3. IDS (Image Development Systems) USB and GigE cameras
   4. Lumenera industrial USB cameras
   5. Viewbits insdustrial CMOS cameras

Note: Support for new cameras can be added NdmMedia API [21].

See Also

Supported IP Cameras [17]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

3.5.1 IP Cameras

The list of supported IP cameras for different operating systems is provided in the following table:

Camera name	Microsoft Windows	Linux
Axis M1114 *	+	+
Mobotix [19] DualNight M12 IP	+	+
Mobotix [19] S14D	+	+
Basler scA1000-30gm **	+	
PiXORD N606 *	+	+
Sony SNC-CS50 *	+	+
Cisco [18] 4500 IP	+	
Prosilica [19] GigE Vision	+	

Other supported cameras:

- Any IP camera, that supports RTSP (Real Time Streaming Protocol):
  - Only RTP over UDP is supported.
  - H.264/MPEG-4 AVC or Motion JPEG should be used for encoding the video stream.
- Any high-resolution digital camera that is accessible using:
  - DirectShow interface for Microsoft Windows platform;
  - GStreamer interface for Linux platform.
- Any other device support can be added by customers using the provided Device Manager Plug-in Framework.

Please refer to the VeriLook Surveillance 2.2 SDK documentation for the detailed information.

For more information on how to use IP cameras read Using IP Cameras [18] section.

* These cameras are configured using NdmMedia plugin [19].

** Requires Basler Pylon SDK

Notes

Only the cameras listed in the table were thoroughly tested and are guaranteed to work.

If you are going to use any other camera than listed in the table, you may want to contact the support [1920] to make sure your camera works properly.

3.6 Error codes

Almost all functions return NResult [1007]. To check whether functions is succeeded the macros NFailed [434] and NSucceeded [434] can be used.

NError [398] (Neurotec [1469] for .Net) defines error codes used in Neurotechnology components.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

4 Using

4.1 Using Devices

The Neurotechnology SDKs provides convenient and unified way for device access. All the devices supported by the SDKs can be discovered through NDeviceManger [1168] Unit (NDeviceManager [1814] Class). Device support is implemented as plugins (dynamic libraries) and plugin management mechanism can be used to control which devices will be enabled. Please see the documentation of the NDevices [1124] library for more information.

For complete device usage examples, please see Devices sample application [55] in \Samples\Devices\ SDK directory and tutorials in \Tutorials\Devices\ directory.

4.1.1 Using IP Cameras

Most of the supported IP cameras can be used from NDeviceManger through NdmMedia plugin [19]. Or used directly through lower level NMedia API [1196] (NMediaReader [1336] and NMediaSource [1345] units). For example NMediaReaderCreateFromUrl [1338] function (NMediaReader.NMediaReader constructor).

Some cameras need special NDeviceManger plugin and their use is described in separate subsections:

- Cisco Video Surveillance IP 4500 [18]
- Mobotix DualNight M12 [19]
- Prosilica GigE Vision [19]

Also these cameras were tested and approved to work with Neurotechnology products:

- **Basler scA1000-30gm** (works only on Windows OS; requires Basler Pylon SDK)
- **Basler BIP2-1600-25c-DN** (works on Windows and Linux using RTSP)

4.1.1.1 Cisco Video Surveillance IP 4500

Cisco Video Surveillance IP 4500 camera is supported (on Windows and Linux) through NdmCisco plugin for NDeviceManger. The camera should be configured by editing the `NdmCisco.conf` file in `\Bin\Win32_x86\Cameras\NdmCisco\`, `\Bin\Win64_x64\Cameras\NdmCisco\` and `\Bin\Linux_x86\Cameras\NdmCisco\` directories of the SDK. Multiple cameras can be specified, by using the following template (for each camera):

```
[Camera]
Ip=192.168.2.150
Username=admin
Password=Admin123
ChannelId=1
ChannelName=Channel1
Id=MyId
DisplayName=MyDisplayName
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

1. *IP* - defines IP address of camera channel
2. *Username* - camera's username. To set camera's username use Cisco software.
3. *Password* - camera's user password.
4. *ChannelId* - Camera's channel ID.
5. *ChannelName* - Name of camera's channel.
6. *Id* - NDeviceManger ID for the camera.
7. *DisplayName* - NDeviceManger display name for the camera.

4.1.1.2 Mobotix DualNight M12

Mobotix DualNight M12 IP camera is supported (on Windows and Linux OS) through NdmMobotix plugin for NDeviceManger. The camera should be configured by editing the `NdmMobotix.conf` file in `\Bin\Win32_x86\Cameras\NdmMobotix\`, `\Bin\Win64_x64\Cameras\NdmMobotix\` and `\Bin\Win64_x64\Linux_x86\NdmMobotix\` directories of the SDK. Multiple cameras can be specified, by using the following template (for each camera):

```
[Camera]
Ip=192.168.2.136
Id=MobotixId
DisplayName=MobotixDisplayName
Channel=right
FrameRate=20
```

1. *IP* - defines IP address of camera channel
2. *Id* - NDeviceManger ID for the camera.
3. *DisplayName* - NDeviceManger display name for the camera.
4. *Channel* - camera's channel.
5. *FrameRate* - camera frame rate. Default value is 20.

4.1.1.3 Prosilica GigE Vision

Prosilica GigE Vision camera is supported (on Windows and Linux OS) through NdmProsilica plugin for NDeviceManger. The plugin is saved as `NdmProsilica.dll` file in `\Bin\Win32_x86\Cameras\NdmProsilica\`, `\Bin\Win64_x64\Cameras\NdmProsilica\` or `\Bin\Linux_x86\Cameras\NdmProsilica\` directories of the SDK.

Notes

This camera does not require any additional configuration.

4.1.2 NdmMedia plugin

NdmMedia library is plugin for NDeviceManger allowing to use Video and Audio devices. It provides audio/video input supported by NMedia API [1196], and allows supported device to be used as NMicrophone and NCamera [1145] objects.

It's configuration is platform dependant and is described in next subsections.

Note: If NDeviceManger interface is not needed , the NMedia API can be used directly (see NMediaReader [1336] and NMediaSource [1345] units).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

4.1.2.1 Configuring on Microsoft Windows

On Windows the plugin is saved as `NdmMedia.dll` file in `Bin\Win32_x86\` or `Bin\Win64_x64\` directories of SDK. The plugin allows to use all Video/Audio devices available through DirectShow and Windows Media Foundation frameworks. Additional virtual devices can be added. These virtual devices would use rtsp stream or local media file for their content.

The plugin is configured through `NdmMedia.ini` configuration file in the same directory. The file contains commented example of the possible configuration, the format is described further. The ';' character can be used to comment out the line.

The file uses usual `.ini` format, where certain parameters are grouped into sections:

Section	Description
[NAudioSource]	Specifies parameters controlling usage of all the local audio devices.
[NVideoSource]	Specifies parameters controlling usage of all the local video devices.
[Camera]	Section defines additional video input device. This can be remote stream (for example from IP camera) or "virtual" device which would provide video from media file.
[Microphone]	Section defines additional audio input device. This can be remote stream or "virtual" device which would provide audio from media file.

The first two sections described ([`NAudioSource`] and [`NVideoSource`]) should appear in the configuration file only once, as their settings control all the local devices. The each of the other sections ([`Camera`] and [`Microphone`]), define a new device and can appear as many times as needed.

NdmMedia library can use devices through two frameworks: DirectShow or Windows Media Foundation. Following flags allow to select which way is preferred and can appear in just described sections:

Flag	Description
UseDirectShow	Flag used to set whether to use DirectShow for video source. `UseDirectShow=False` - disables DirectShow `UseDirectShow=True` - enables DirectShow
UseWindowsMediaFoundation	Flag used to set whether to use Windows Media Foundation for video source. `UseWindowsMediaFoundation=False` - disables Windows Media Foundation `UseWindowsMediaFoundation=True` - enables Windows Media Foundation
PreferDirectShow	Flag used to prefer DirectShow versus Windows Media Foundation. `PreferDirectShow=False` - prefers to use Windows Media Foundation `PreferDirectShow=True` - prefers to use DirectShow vs. Windows Media Foundation
AllowDuplicateDevices	This flag is used to allow enumerating duplicate devices. If this flag was set to true and the same device is accessible through DirectShow and Windows Media Foundation (Windows 7) then this device will be enumerated twice. **Note:** Valid only for [`NAudioSource`] and [`NVideoSource`]) sections

Following parameters define a device and can be used in [`Camera`] and [`Microphone`] sections:

Parameter	Description
Url	Rtsp address of media source (IP camera). For example: `Url=rtsp://192.168.2.120:554/media/video1`
Filename	Media source file name. For example: `Filename=C:\test.wav` **Note**: Only Filename or Url address can be specified at a time, not both.
Id	Camera Id (optional parameter). This Id overrides original camera Id. For example: `Id=My Camera`

Copyright © 2009-2014 Neurotechnology. All rights reserved.

DisplayName	Display name for camera (optional parameter). This parameter overrides default camera display name. For example: `DisplayName=My Camera Display name`

4.1.2.2 Configuring on Linux and Mac OS X

On Linux the plugin is saved as `NdmMedia.so` file in `Lib/Linux_x86/` or `Lib/Linux_x86_64/` directories of SDK.

On Mac OS X the `NeurotecBiometric.dmg` installer saves the plugin as `NdmMedia.dylib` file in `Library/Frameworks/Neurotechnology/` sub directory (of SDK installation target directory).

The plugin allows to use all Video/Audio devices available through gstreamer framework. Additional virtual devices can be added. These virtual devices would use rtsp stream or local media file for their content.

Note: *The current release doesn't support additional virtual devices on Mac OS X.*

Plugin is configured through `NdmMedia.conf` configuration file in the directory of the library. The file contains commented example of the possible configuration, the format is described further. The '#' character can be used to comment out the line.

The file uses `.ini` like format, where certain parameters are grouped into sections. Each sections defines a new device and can appear as many times as needed:

Section	Description
[Camera]	Section defines additional video input device. This can be remote stream (for example from IP camera) or "virtual" device which would provide video from media file.
[Microphone]	Section defines additional audio input device. This can be remote stream or "virtual" device which would provide audio from media file.

The device is described by following parameters in each section:

Parameter	Description
Url	Rtsp address of media source (IP camera). For example: `Url=rtsp://192.168.2.120:554/media/video1`
Filename	Media source file name. For example: `Filename=/tmp/test.wav` **Note**: Only Filename or Url address can be specified at a time, not both.
Id	Camera Id (optional parameter). This Id overrides original camera Id. For example: `Id=My Camera`
DisplayName	Display name for camera (optional parameter). This parameter overrides default camera display name. For example: `DisplayName=My Camera Display name`

4.1.3 Adding custom device to NDeviceManager

The Plug-in framework for the device manager allows developers to create custom device modules for applications based on any of the Neurotechnology SDKs. With this framework it is possible to create custom microphone (audio), video, fingerprint and iris capturing modules. The plug-in is a dynamic library, which is loaded by the device manager and communicates with a particular device. The application does not need to know the specifics of that particular device in order to use it, because all the details are hidden behind the device independent interface.

This section describes interface of device manger plugin. The interface is best understood by example, so please use the

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NdmSample [55] source code as addition to the information contained in following sections.

4.1.3.1 Naming requirements

The plugin's name ***must start with*** "`Ndm`" prefix (which stands for Neurotechnology device manager). The remaining part of the name is up to the author, but should be meaningful. If the plugin provides support for single device it is recommended to use it's name.

For plugin to be loaded by the framework, the ***plugin's dynamic library file should have the name of the plugin***. The library should also contain the ***function returning the handle to the plugin module***. The function's name should be plugin name followed by "`ModuleOf`". For example, the plugin named `NdmSample` should have `NdmSampleModuleOf()` function and be saved as `NdmSample.dll` file (`NdmSample.so` under Linux).

4.1.3.2 Device manager plugin lifecycle

It is useful for plugin writer to understand how plugins are used by the framework on API users activity. There are certain stages of plugin life, defining it's state and available operations (please refer to NdmSample [55] for source example).

Plugin load

Plugin's library file is loaded automatically (from plugin directory [619]) on first use of device manager or upon API users request (NPluginManagerEnsurePluginSearchPerformed [615]/NPluginManagerRefresh [620]/NPluginManagerPlugAll [620] or NPluginManagerAddPluginFromFile [612] functions).

Plugin library should have a module constructor, that is invoked automatically during library load (for example from DllMain() on Windows). The constructor must create a NPluginModule [629] and register it in the framework (by calling NModuleRegister [529] function). The module should support the device manager plugin interface [25] (and register it with NPluginModuleSetInterfaceVersions [636] function).

When the library is loaded (DllMain() finishes), the framework would try to get the handle to loaded plugin. The framework expects to get the handle by calling "<PluginFileBaseName>ModuleOf" function (for example NdmSampleModuleOf for NdmSample.dll). This function is mandatory, and it must match the name of the library file.

Next the interface type and version compatibility will be checked from obtained handle. The framework would also query the type of devices provided by the plugin (GetSupportedDeviceTypes [27] interface function), and would check if all the functions required for given device type are present in the interface.

If an error is encountered at any stage of loading process, the plugin's state [603] will be set accordingly, and can be checked by API user during troubleshooting. The framework would also detect and won't load duplicates [603] (plugins with same name) and incompatible [603] plugins.

If a plugin is successfully loaded it would have npsDisabled [603], npsUnplugged [603] or npsUnused [603] state. Exact state depends on API usage, see the documentation of each state for conditions of each case.

Note: There might be many plugins for different devices and some of them might not be used. Plugin writer should ensure that loaded plugin consumes minimal resources if it is not actually used, i.e. before is plugged as described next.

Plugin Plug

Framework will try to plug the loaded plugin only if both following conditions are true:

- The plugin is not disabled [603] and activated by an API user (NPluginPlug [600] or NPluginManagerPlugAll [620]/NPluginManagerRefresh [620] functions). The activated plugin will have npsUnused [603] state.
- There is a device manager that enumerates the devices of the type provided by the plugin.

Once the both conditions are met the framework will call the plug function (registered by NPluginModuleSetPlug [636]() from the module constructor). The function should prepare the module for work. It should check the availability of physical

Copyright © 2009-2014 Neurotechnology. All rights reserved.

devices and register [24] them within the device manager. Please also note that not only freshly loaded plugins, but also previously unplugged ones can be connected. The module should be designed for multiple plug/unplug cycles.

Plugin's state becomes npsPlugged [603] if plug function completes successfully or npsPluggingError [603] if the function fails with error.

Plugin Use

The devices registered in plug function will be presented to the API user in appropriate category, and the user can get the device handle and perform operation supported by the device. Like getting various attributes of the device or performing capturing. Appropriate interface [25] functions will be called during the use of the device.

The device manager will periodically ask the plugged plugin to update the list [27] of provided devices. The plugin should check if registered physical devices are still available and if new devices have appeared. The plugin can also update the device list at any time (for example if device's driver allows to get the availability change events).

Plugin Unplug

To save resources, the framework will unplug the plugin when it is no longer needed. This happens when the last device manager enumerating the device is freed. This will also happen when the API user unplugs the plugin explicitly (NPluginUnplug [602]/NPluginManagerUnplugAll [625] functions).

The unplug function (registered by NPluginModuleSetUnplug [636]() from the module constructor) will be called. The function should unregister devices that were provided to device manager and clean-up resources allocated in NdmSamplePlug() and further operation.

Note: Users can unplug the plugin at any time (for example the device is still in use). Special care must be taked to implement the proper synchronisation.

4

4.1.3.3 Preventing race conditions

The functions of plugin interface [25] are called both by device manger and by device users. This can happen at different times and from different threads. Special care should be taken in plugin code to maintain consistency of data structures through use of critical sections, and to avoid deadlocks at the same time. There are two common issues to take into account:

- API users might have obsolete handles to devices (for example, device have already been removed or module unpluged/repluged). Plugin functions performing device operation should check this and return proper error code. One design pattern for achieving this is to maintain the internal list of present devices. See NdmSample [55].
- Plugin interface function can be called from the implementation of AddDevice [24] and RemoveDevice [25] functions, so to avoid deadlock possibility, no lock should be held when registering/removing device with these two functions.

4.1.3.4 Inteface reference

The interface is declared in `NdmInterface.h` file. The interface is bidirectional, i.e. the plugin provides certain functionality to be used by device manager and vice versa. The functions for each usage direction are grouped into separate structure:

- NdmHostInterfaceV1 [23] – used by plugin to access device manger;
- NdmInterfaceV1 [25] – used by device manger to access plugin.

4.1.3.4.1 Accesing NDeviceManger

NDeviceManager manages the list of available devices, and provides them to the users. It is responsibility of a device plugin, to inform the NDeviceManger of provided device availability changes. Plugin can perform this by calling the NDeviceManger functions available in NdmHostInterfaceV1 structure:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Function	Description
IsDriverLoaded [24]	*[Optional]* Allows to check if Windows OS driver required by plugin has been loaded.
AddDevice [24]	Registers device in device managers.
LoseDevice [24]	Asks device managers to mark device as unavailable [1164].
RemoveDevice [25]	Removes device from device managers.

Note that there might be multiple instances of NDeviceManger used by the API user. The AddDevice/LoseDevice/RemoveDevice functions described in this table need to be called only once, and the framework will update all the device manager instances. It will also ensure that this update will happen at safe time, as requested by the device manager user. For example the device will not be removed from

4.1.3.4.1.1 IsDriverLoaded

Special function used by some plugins on Windows OS. Allows to check if Windows OS driver required by plugin has been loaded.

```
NResult (N_CALLBACK IsDriverLoaded)(const NChar * szName, NBool * pValue);
```

Parameters

szName	[in] Name of the driver to check.
pValue [870]	[out] Pointer to variable that will be updated with the result.

Return value

Function returns N_OK [433] on success or one of the error codes defined in NErrors.h.

4.1.3.4.1.2 AddDevice

Registers device in device managers.

```
void (N_CALLBACK AddDevice)(NHandle hDevice, HNPlugin hPlugin, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
hPlugin	[in] Handle to current plugin.
pParam	[in] Pointer to optional argument (currently not used).

Remarks

Device manager might call some functions provided [25] by plugin. To avoid deadlock, the plugin should not hold any locks when calling AddDevice.

4.1.3.4.1.3 LoseDevice

Asks device managers to mark device as unavailable [1164].

```
void (N_CALLBACK LoseDevice)(NHandle hDevice, HNPlugin hPlugin, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
hPlugin	[in] Handle to current plugin.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

pParam	[in] Pointer to optional argument (currently not used).

Remarks

It is recommended to inform device managers about device unavailability as soon as this was discovered. The LoseDevice will not call any plugin interface [25] functions, so it is save to call this function from any context. The RemoveDevice [25] should also be called, but only after internal device structure is updated and locks released.

4.1.3.4.1.4 RemoveDevice

Removes device from device managers.

```
void (N_CALLBACK RemoveDevice)(NHandle hDevice, HNPlugin hPlugin, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
hPlugin	[in] Handle to current plugin.
pParam	[in] Pointer to optional argument (currently not used).

Remarks

Device manager might call some functions provided [25] by plugin. To avoid deadlock, the plugin should not hold any locks when calling RemoveDevice. Instead the plugin should use LoseDevice [24] as soon as possible and call RemoveDevice after internal device structure is updated and locks released.

4

4.1.3.4.2 Functionality expected by NDeviceManger

NDeviceManager will need to query the plugin for types of supported devices to provides them to the users. The plugin also must provide function for device operation and capability queries. The are also few utility functions what are used by device manager. All those functions are grouped in plugin interface structure. Note that the interface is for all possible device types [1124], and not all of the functions have to be implemented in particular plugin (the unused function pointers should be initialized to NULL [1075]). The complete list of functions is provided in the table below and described in separate subsections (for descriptions of used notations see the notes below the table):

Function	Description
GetSupportedDeviceTypes [27]	Returns information about which device types are provided by the plugin. The returned value determines which of the remaining functions will have to be implemented.
UpdateDeviceList [27]	Should recheck the availability of physical devices.
GetDeviceType [28]	*[Rcmd]* Returns device type.
GetDeviceIdA [28]	Returns specific device property.
GetDeviceIdW [28]	Returns specific device property.
GetDeviceDisplayNameA [29]	Returns specific device property.
GetDeviceDisplayNameW [29]	Returns specific device property.
GetDeviceMakeA [30]	*[Opt]* Returns specific device property.
GetDeviceMakeW [30]	*[Opt]* Returns specific device property.
GetDeviceModelA [30]	*[Opt]* Returns specific device property.
GetDeviceModelW [31]	*[Opt]* Returns specific device property.
GetDeviceSerialNumberA [31]	*[Opt]* Returns specific device property.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

GetDeviceSerialNumberW [31]	*[Opt]* Returns specific device property.
StartCaptureDeviceCapturing [32]	*[Cap]* Prepare capture device for acquiring images/sound.
GetCameraFrame [35]	*[Cam]* Get the most recent captured image.
StopCaptureDeviceCapturing [32]	*[Cap]* Finish acquiring images/sound.
IsCaptureDeviceCapturing [33]	*[Cap]* Check if capture device is active (ready to produce data).
GetCaptureDeviceMediaType [33]	*[Cap, Opt]* Returns specific device property.
GetCaptureDeviceFormats [34]	*[Cap, Opt]* Returns specific device property.
GetCaptureDeviceCurrentFormat [34]	*[Cap, Opt]* Returns specific device property.
SetCaptureDeviceCurrentFormat [35]	*[Cap, Opt]* Sets device property.
GetMicrophoneSoundSample [36]	*[Mic]* Get recent sound sample.
GetBiometricDeviceBiometricType [36]	*[Bio, Rcmd]* Returns specific device property.
GetBiometricDeviceVendorId [37]	*[Bio]* Returns specific device property.
GetBiometricDeviceProductId [37]	*[Bio]* Returns specific device property.
IsBiometricDeviceSpoofDetectionSupported [37]	*[Bio,Opt]* Returns specific device property.
GetBiometricDeviceSpoofDetection [38]	*[Bio, Opt]* Returns specific device property.
SetBiometricDeviceSpoofDetection [38]	*[Bio, Opt]* Sets device property.
StartBiometricDeviceSequence [39]	*[Bio, Opt]* Prepare device for capturing of several biometric objects.
EndBiometricDeviceSequence [39]	*[Bio, Opt]* Finish capturing sequence.
CancelBiometricDevice [39]	*[Bio]* Stops capturing of biometric device.
GetFScannerSupportedImpressionTypes [40]	*[FSc, Rcmd]* Returns specific device property.
GetFScannerSupportedPositions [40]	*[FSc, Rcmd]* Returns specific device property.
CaptureFScanner [41]	*[FSc]* Start capturing of fingerprint/palm.
GetIrisScannerSupportedPositions [42]	*[ISc, Rcmd]* Returns specific device property.
CaptureIrisScanner [43]	*[ISc]* Start capturing of irises.
GetDeviceParent [27]	*[Opt]* Reserved for more complex plugins.
CanUse [45]	*[Opt]* Reserved for more complex plugins.
DeviceAdded [44]	*[Opt]* Reserved for more complex plugins.
DeviceLost [44]	*[Opt]* Reserved for more complex plugins.
DeviceRemoved [45]	*[Opt]* Reserved for more complex plugins.

The annotations in the table above mark the type of device which require particular function (see NDevices [1124] library for device type hierarchy diagram):

- *[Opt]* – function is optional (the pointer in the interface structure can be set to NULL [1075] if particular functionality is not be needed);
- *[Rcmd]* – function is not mandatory, but is nevertheless recommended;
- *[Cap]* – function needed for capture devices [1155] only.
  - *[Cam]* – function needed for NCamera [1145] only.
  - *[Mic]* – function needed for NMicrophone only.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- *[Bio]* – function needed for biometric devices [1142] only: i.e finger, palm or iris scanners;
  - *[FSc]* – function needed for NFScanner only: i.e. finger or palm scanners;
  - *[ISc]* – function needed for iris scanners only.

Functions returning strings come in two variants, the one with the name ending with "W" use wide character unicode strings, and the one with "A" ending use ASCII character codes.

4.1.3.4.2.1 GetSupportedDeviceTypes

Returns information about which device types are provided by the plugin.

```
NDeviceType (N_CALLBACK GetSupportedDeviceTypes)();
```

Return value

Function should return bit mask combination with all supported device types [1165].

Remarks

- The function is Mandatory.
- The returned value determines which parts of the interface [25] will have to be supported.

4.1.3.4.2.2 UpdateDeviceList

Should recheck the availability of physical devices. The new devices should be register and the disconnected removed.

```
void (N_CALLBACK UpdateDeviceList)(void);
```

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function is Optional. Can be omitted if plugin knows better when to trigger device update cycle.
- If available, the function will be called periodically by the framework. The plugin can also update the device list at any time (for example if device's driver allows to get the availability change events). In such case (if plugin uses custom way of making the list of available devices up to date) the function can be omitted.

4.1.3.4.2.3 GetDeviceParent

Returns the parent device of a given device.

```
NResult (N_CALLBACK GetDeviceParent)(NHandle hDevice, HNPlugin * phParentPlugin, NHandle *
phParentHandle);
```

Parameters

hDevice	[in] Handle to the device.
phParentPlugin	[out] Handle parent's plugin.
phParentHandle	[out] Handle to the parent device.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function is Optional. Used for multi-biometric devices and in some complex plugins.
- The hDevice handle might be obsolete and should be checked for validity.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

4.1.3.4.2.4 General functions

General functions common to all device types (NDevice [1159] unit).

4.1.3.4.2.4.1 GetDeviceType

Returns the type of a device.

```
NResult (N_CALLBACK GetDeviceType)(NHandle hDevice, NDeviceType * pValue);
```

Parameters

hDevice	[in] Handle to the device.
pValue [870]	[out] Address of the buffer where requested value should be saved.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NDeviceGetDeviceType [1161] is called.
- The function is Recommended. If the function missing, the default value of ndtNone will be used, and only basic device functionality will be usable.
- The hDevice handle might be obsolete and should be checked for validity.

4.1.3.4.2.4.2 GetDeviceIdA

Returns the device ID visible to API user (NDeviceGetId [1162] function).

```
NResult (N_CALLBACK GetDeviceIdA)(NHandle hDevice, NAChar * szValue, NInt valueSize);
```

Parameters

hDevice	[in] Handle to the device.
szValue	[out] Address of the buffer where requested value should be saved.
valueSize	[in] The size of the given buffer.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function is Mandatory.
- The hDevice handle might be obsolete and should be checked for validity.
- The requested value should use the ASCII characters.

4.1.3.4.2.4.3 GetDeviceIdW

Returns the device property visible to API user (NDeviceGetId [1162] function).

```
NResult (N_CALLBACK GetDeviceIdW)(NHandle hDevice, NWChar * szValue, NInt valueSize);
```

Parameters

hDevice	[in] Handle to the device.
szValue	[out] Address of the buffer where requested value should be saved.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

valueSize	[in] The size of the given buffer.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function is Mandatory.
- The hDevice handle might be obsolete and should be checked for validity.
- The requested value should use the wide characters.

4.1.3.4.2.4.4 GetDeviceDisplayNameA

Returns the device property visible to API user (NDeviceGetDisplayName [1161] function).

```
NResult (N_CALLBACK GetDeviceDisplayNameA)(NHandle hDevice, NAChar * szValue, NInt
valueSize);
```

Parameters

hDevice	[in] Handle to the device.
szValue	[out] Address of the buffer where requested value should be saved.
valueSize	[in] The size of the given buffer.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function is Mandatory.
- The hDevice handle might be obsolete and should be checked for validity.
- The requested value should use the ASCII characters.

4.1.3.4.2.4.5 GetDeviceDisplayNameW

Returns the device property visible to API user (NDeviceGetDisplayName [1161] function).

```
NResult (N_CALLBACK GetDeviceDisplayNameW)(NHandle hDevice, NWChar * szValue, NInt
valueSize);
```

Parameters

hDevice	[in] Handle to the device.
szValue	[out] Address of the buffer where requested value should be saved.
valueSize	[in] The size of the given buffer.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function is Mandatory.
- The hDevice handle might be obsolete and should be checked for validity.
- The requested value should use the wide characters.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

4.1.3.4.2.4.6 GetDeviceMakeA

Returns the device property visible to API user (NDeviceGetMake [1162] function).

```
NResult (N_CALLBACK GetDeviceMakeA)(NHandle hDevice, NAChar * szValue, NInt valueSize);
```

Parameters

hDevice	[in] Handle to the device.
szValue	[out] Address of the buffer where requested value should be saved.
valueSize	[in] The size of the given buffer.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function is Optional.
- The hDevice handle might be obsolete and should be checked for validity.
- The requested value should use the ASCII characters.

4.1.3.4.2.4.7 GetDeviceMakeW

Returns the device property visible to API user (NDeviceGetMake [1162] function).

```
NResult (N_CALLBACK GetDeviceMakeW)(NHandle hDevice, NWChar * szValue, NInt valueSize);
```

Parameters

hDevice	[in] Handle to the device.
szValue	[out] Address of the buffer where requested value should be saved.
valueSize	[in] The size of the given buffer.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function is Optional.
- The hDevice handle might be obsolete and should be checked for validity.
- The requested value should use the wide characters.

4.1.3.4.2.4.8 GetDeviceModelA

Returns the device property visible to API user (NDeviceGetModel [1162] function).

```
NResult (N_CALLBACK GetDeviceModelA)(NHandle hDevice, NAChar * szValue, NInt valueSize);
```

Parameters

hDevice	[in] Handle to the device.
szValue	[out] Address of the buffer where requested value should be saved.
valueSize	[in] The size of the given buffer.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

- The function is Optional.
- The hDevice handle might be obsolete and should be checked for validity.
- The requested value should use the ANCII characters.

4.1.3.4.2.4.9 GetDeviceModelW

Returns the device property visible to API user (NDeviceGetModel [1162] function).

```
NResult (N_CALLBACK GetDeviceModelW)(NHandle hDevice, NWChar * szValue, NInt valueSize);
```

Parameters

hDevice	[in] Handle to the device.
szValue	[out] Address of the buffer where requested value should be saved.
valueSize	[in] The size of the given buffer.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function is Optional.
- The hDevice handle might be obsolete and should be checked for validity.
- The requested value should use the wide characters.

4.1.3.4.2.4.10 GetDeviceSerialNumberA

Returns the device property visible to API user (NDeviceGetSerialNumber [1163] function).

```
NResult (N_CALLBACK GetDeviceSerialNumberA)(NHandle hDevice, NAChar * szValue, NInt
valueSize);
```

Parameters

hDevice	[in] Handle to the device.
szValue	[out] Address of the buffer where requested value should be saved.
valueSize	[in] The size of the given buffer.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function is Optional.
- The hDevice handle might be obsolete and should be checked for validity.
- The requested value should use the ASCII characters.

4.1.3.4.2.4.11 GetDeviceSerialNumberW

Returns the device property visible to API user (NDeviceGetSerialNumber [1163] function).

```
NResult (N_CALLBACK GetDeviceSerialNumberW)(NHandle hDevice, NWChar * szValue, NInt
valueSize);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

hDevice	[in] Handle to the device.
szValue	[out] Address of the buffer where requested value should be saved.
valueSize	[in] The size of the given buffer.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function is Optional.
- The hDevice handle might be obsolete and should be checked for validity.
- The requested value should use the wide characters.

4.1.3.4.2.5 Functions for capture devices

General functions used by capture devices (NCaptureDevice [1155] unit).

4.1.3.4.2.5.1 StartCaptureDeviceCapturing

Prepare device for providing of the data.

```
NResult (N_CALLBACK StartCaptureDeviceCapturing)(NHandle hDevice,
NdmIsCaptureDeviceCapturingChangedProc pIsCapturingChanged, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
pIsCapturingChanged	[in] Device manager callback. Must be invoked after capturing is started and stopped.
pParam	[in] The capturing context. The value must also be passed to the callback.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NCaptureDeviceStartCapturing [1158] is called.
- The function is Mandatory for ndtCaptureDevice.
- The hDevice handle might be obsolete and should be checked for validity.
- The pIsCaptureDeviceCapturingChanged callback must be performed not only if stop is requested by the API (call to StopCaptureDeviceCapturing [32]) but also if capturing is stopped due to other reason (like error, stream termination, device disconnection or plugin unplug).
- Different contexts (pParam values) correspond to devices accessed from different device managers. Consequently pParam should be remembered and checked in other capturing function, otherwise the device usages through different device managers will interfere.
- The plugin should take care that the internal buffers don't grow indefinitely. I.e. must drop the old data if the GetCameraFrame [35] or GetMicrophoneSoundSample [36] are not called fast enough.

4.1.3.4.2.5.2 StopCaptureDeviceCapturing

The device will not have to provide data in near future.

```
NResult (N_CALLBACK StopCaptureDeviceCapturing)(NHandle hDevice, void * pParam);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

hDevice	[in] Handle to the device.
pParam	[in] Context argument (should be the same as given to StartCaptureDeviceCapturing).

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NCaptureDeviceStopCapturing [1159] is called.
- The function is Mandatory for ndtCaptureDevice.
- The hDevice handle might be obsolete and should be checked for validity.
- Different contexts (pParam values) correspond to devices accessed from different device managers. Consequently pParam should be checked to be equal to pParam passed to StartCaptureDeviceCapturing [32], otherwise the device usages through different device managers will interfere.

4.1.3.4.2.5.3 IsCaptureDeviceCapturing

Check if the device is capturing.

```
NResult (N_CALLBACK IsCaptureDeviceCapturing)(NHandle hDevice, NBool * pValue, void *
pParam);
```

Parameters

hDevice	[in] Handle to the device.
pValue [870]	[out] Pointer to the variable where requested value should be saved.
pParam	[in] Context argument (should be the same as given to StartCaptureDeviceCapturing).

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NCaptureDeviceIsCapturing [1157] is called.
- The function is Mandatory for ndtCaptureDevice.
- The hDevice handle might be obsolete and should be checked for validity.
- Different contexts (pParam values) correspond to devices accessed from different device managers. Consequently pParam should be checked to be equal to pParam passed to StartCaptureDeviceCapturing [32], otherwise the device usages through different device managers will interfere.

4.1.3.4.2.5.4 GetCaptureDeviceMediaType

Returns device media type.

```
NResult (N_CALLBACK GetCaptureDeviceMediaType)(NHandle hDevice, NMediaType * pValue);
```

Parameters

hDevice	[in] Handle to the device.
pValue [870]	[out] Pointer to the variable where requested value should be saved.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NCaptureDeviceGetMediaType [1157] is called.
- The function is Optional for ndtCaptureDevice. The default value will be infered from GetDeviceType if the function is missing.
- The hDevice handle might be obsolete and should be checked for validity.

4.1.3.4.2.5.5 GetCaptureDeviceFormats

Returns the list of formats supported by the capture device.

```
NResult (N_CALLBACK GetCaptureDeviceFormats)(NHandle hDevice, HNMediaFormat * *
parhFormats, NInt * pFormatCount);
```

Parameters

hDevice	[in] Handle to the device.
parhFormats	[out] Pointer which will be updated with the address of allocated array.
pFormatCount	[out] The number of elements in returned array.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NCaptureDeviceGetFormats [1156] is called.
- The function is Recommended for ndtCaptureDevice. If present, SetCaptureDeviceCurrentFormat is also required. Empty array will be used if the function is missing.
- The hDevice handle might be obsolete and should be checked for validity.
- The function must allocate the array with NAlloc [504] function and each element with NAudioFormatCreate/NVideoFormatCreate [1398] (depending on device type). The caller will be responsible for freeing this allocation.

4.1.3.4.2.5.6 GetCaptureDeviceCurrentFormat

Returns the list of formats supported by the capture device.

```
NResult (N_CALLBACK GetCaptureDeviceCurrentFormat)(NHandle hDevice, HNMediaFormat *
phFormat);
```

Parameters

hDevice	[in] Handle to the device.
phFormat	[out] Address which will receive the handle to structure describing current format.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NCaptureDeviceGetCurrentFormat [1156] is called.
- The function is Recommended for ndtCaptureDevice. If present, SetCaptureDeviceCurrentFormat is also required. NULL [1075] will be used if the function is missing.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- The hDevice handle might be obsolete and should be checked for validity.
- The function must allocate the phFormat with NAudioFormatCreate/NVideoFormatCreate [1398] (depending on device type). The caller will be responsible for freeing this allocation.

4.1.3.4.2.5.7 SetCaptureDeviceCurrentFormat

Select the format that will be used during capturing.

```
NResult (N_CALLBACK SetCaptureDeviceCurrentFormat)(NHandle hDevice, HNMediaFormat hFormat,
void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
phFormat	[in] Handle to the structure describing requested format.
pParam	[in] Context argument (should be the same as given to StartCaptureDeviceCapturing).

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NCaptureDeviceSetCurrentFormat [1158] is called.
- The function is Recommended for ndtCaptureDevice.
- The hDevice handle might be obsolete and should be checked for validity.
- Different contexts (pParam values) correspond to devices accessed from different device managers. Consequently pParam should be checked to be equal to pParam passed to StartCaptureDeviceCapturing [32], otherwise the device usages through different device managers will interfere.

4.1.3.4.2.6 Functions for cameras

Functions used by cameras (NCamera [1145] unit).

4.1.3.4.2.6.1 GetCameraFrame

Get the next recent frame.

```
NResult (N_CALLBACK GetCameraFrame)(NHandle hDevice, HNImage * phImage, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
phImage	[out] Pointer to variable which will receive the handle of the captured image.
pParam	[in] Context argument (should be the same as given to StartCaptureDeviceCapturing).

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NCameraGetFrame [1149] is called.
- The function is Mandatory for ndtCamera.
- The hDevice handle might be obsolete and should be checked for validity.
- Different contexts (pParam values) correspond to devices accessed from different device managers. Consequently pParam should be checked to be equal to pParam passed to StartCaptureDeviceCapturing [32], otherwise the device usages through different device managers will interfere.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- Plugin writer should implement some sort of buffering to store 1 or 2 most recent frames. The buffer should be updated so that most recent frames are available to the user (the new data received from the device replaces old one). This also assures that buffers won't grow indefinitely if the user won't be able to call GetCamerFrame at high rate.

4.1.3.4.2.7 Functions for microphones

Functions used by microphones (NMicrophone unit).

4.1.3.4.2.7.1 GetMicrophoneSoundSample

Returns the recent sound sample.

```
NResult (N_CALLBACK GetMicrophoneSoundSample)(NHandle hDevice, HNSoundBuffer *
phSoundBuffer, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
phSoundBuffer	[out] Address which will receive the handle to captured sound buffer.
pParam	[in] Context argument (should be the same as given to StartCaptureDeviceCapturing).

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NMicrophoneGetSoundSample is called.
- The function is Mandatory for ndtMicrophone.
- The hDevice handle might be obsolete and should be checked for validity.
- Different contexts (pParam values) correspond to devices accessed from different device managers. Consequently pParam should be checked to be equal to pParam passed to StartCaptureDeviceCapturing [32], otherwise the device usages through different device managers will interfere.
- The resulting buffer should be allocated (with NAlloc [504]) and filled by using functions from NSoundBuffer unit.
- Plugin writer should implement some sort of buffering to store recent sound input (1-2 seconds). The buffer should be updated so that new data received from the device replaces obsolete one (so that buffers don't grow if user won't call GetMicrophoneSoundSample function after starting the capturing). This is similar requirement for camera devices, though cameras should only keep 1-2 recent frames (compared to 1-2 seconds of sound).

4.1.3.4.2.8 Functions for biometric devices

General functions used by biometric devices (NBiometricDevice [1792] unit).

4.1.3.4.2.8.1 GetBiometricDeviceBiometricType

Returns the biometric type of a device (NBiometricDeviceGetBiometricType [1143] function).

```
NResult (N_CALLBACK GetBiometricDeviceBiometricType)(NHandle hDevice, NBiometricType *
pValue);
```

Parameters

hDevice	[in] Handle to the device.
pValue [870]	[out] Pointer to the variable where requested value should be saved.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

- The function will be invoked when API function NBiometricDeviceGetBiometricType [1143] is called.
- The function is Recommended for ndtBiometricDevice. The default value will be infered from GetDeviceType if the function is missing.
- The hDevice handle might be obsolete and should be checked for validity.

4.1.3.4.2.8.2 GetBiometricDeviceVendorId

Returns the biometric device's vendor id (NBiometricDeviceGetVendorId [1144] function).

```
NResult (N_CALLBACK GetBiometricDeviceVendorId)(NHandle hDevice, NUShort * pValue);
```

Parameters

hDevice	[in] Handle to the device.
pValue [870]	[out] Pointer to the variable where requested value should be saved.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NBiometricDeviceGetVendorId [1144] is called.
- The function is Optional for ndtBiometricDevice. If present, GetBiometricDeviceProductId is also required.
- The hDevice handle might be obsolete and should be checked for validity.

4.1.3.4.2.8.3 GetBiometricDeviceProductId

Returns the biometric device's product id (NBiometricDeviceGetProductId [1143] function).

```
NResult (N_CALLBACK GetBiometricDeviceProductId)(NHandle hDevice, NUShort * pValue);
```

Parameters

hDevice	[in] Handle to the device.
pValue [870]	[out] Pointer to the variable where requested value should be saved.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NBiometricDeviceGetProductId [1143] is called.
- The function is Optional for ndtBiometricDevice. If present, GetBiometricDeviceVendorId is also required.
- The hDevice handle might be obsolete and should be checked for validity.

4.1.3.4.2.8.4 IsBiometricDeviceSpoofDetectionSupported

Used to check if device supports detection of spoof input (NBiometricDeviceIsSpoofDetectionSupported [1144] function).

```
NResult (N_CALLBACK IsBiometricDeviceSpoofDetectionSupported)(NHandle hDevice, NBool *
pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

hDevice	[in] Handle to the device.
pValue [870]	[out] Pointer to the variable where requested value should be saved.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NBiometricDeviceIsSpoofDetectionSupported [1144] is called.
- The function is Optional for ndtBiometricDevice. If present, GetBiometricDeviceSpoofDetection and SetBiometricDeviceSpoofDetection are also required.
- The hDevice handle might be obsolete and should be checked for validity.

4.1.3.4.2.8.5 GetBiometricDeviceSpoofDetection

Used to check if device's spoof detection is turned on (NBiometricDeviceGetSpoofDetection [1144] function).

```
NResult (N_CALLBACK GetBiometricDeviceSpoofDetection)(NHandle hDevice, NBool * pValue);
```

Parameters

hDevice	[in] Handle to the device.
pValue [870]	[out] Pointer to the variable where requested value should be saved.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NBiometricDeviceGetSpoofDetection [1144] is called.
- The function is Optional for ndtBiometricDevice. If present, IsBiometricDeviceSpoofDetectionSupported and SetBiometricDeviceSpoofDetection are also required.
- The hDevice handle might be obsolete and should be checked for validity.

4.1.3.4.2.8.6 SetBiometricDeviceSpoofDetection

Used to change device spoof detection (NBiometricDeviceSetSpoofDetection [1145] function).

```
NResult (N_CALLBACK SetBiometricDeviceSpoofDetection)(NHandle hDevice, NBool value);
```

Parameters

hDevice	[in] Handle to the device.
value	[in] NTrue [1064] should enable spoof detection, NFalse [1041] should disable it.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NBiometricDeviceSetSpoofDetection [1145] is called.
- The function is Optional for ndtBiometricDevice. If present, IsBiometricDeviceSpoofDetectionSupported and

Copyright © 2009-2014 Neurotechnology. All rights reserved.

GetBiometricDeviceSpoofDetection are also required.

- The hDevice handle might be obsolete and should be checked for validity.

4.1.3.4.2.8.7 CancelBiometricDevice

Stop capturing of biometric device.

```
NResult (N_CALLBACK CancelBiometricDevice)(NHandle hDevice, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
pParam	[in] Context argument (should be the same as given to CaptureIrisScanner/CaptureFScanner).

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NBiometricDeviceCancel [1142] is called. The function should interrupt the capturing (and wait untill the capturing is actually stopped).
- The function is Mandatory for ndtBiometricDevice.
- The hDevice handle might be obsolete and should be checked for validity.
- Different contexts (pParam values) correspond to devices accessed from different device managers. Consequently pParam should be checked to be equal to pParam passed to CaptureFScanner [41] or CaptureIrisScanner [43], otherwise the device usages through different device managers will interfere.

4.1.3.4.2.8.8 StartBiometricDeviceSequence

Prepare the device for multiple capture operations.

```
NResult (N_CALLBACK StartBiometricDeviceSequence)(NHandle hDevice, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
pParam	[in] The capturing context.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NBiometricDeviceStartSequence [1145] is called.
- The function is Optional for ndtBiometricDevice. If present, EndBiometricDeviceSequence is also required.
- The function should not start capturing, but just prepare the device for multiple capturing operations (wake it from sleep state, turn on the backlight, etc.). The function should also mark the device as prepared for capturing, so that next call to capturing function doesn't need to perform this preparations (can run faster).
- The hDevice handle might be obsolete and should be checked for validity.
- Different contexts (pParam values) correspond to devices accessed from different device managers. Consequently pParam should be checked in all capturing functions. The equal value should also be passed to CaptureFScanner [41] or CaptureIrisScanner [43], otherwise the device usages through different device managers will interfere.

4.1.3.4.2.8.9 EndBiometricDeviceSequence

Ends multiple images capturing session.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
NResult (N_CALLBACK EndBiometricDeviceSequence)(NHandle hDevice, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
pParam	[in] Context argument (should be the same as given to StartBiometricDeviceSequence).

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NBiometricDeviceEndSequence [1142] is called.
- The function is Optional for ndtBiometricDevice. If present, StartBiometricDeviceSequence is also required.
- The function could put the device to idle state (power saving). The function should also mark the device as not prepared for capturing, so that next call to capturing function would need to prepare the device, perform capturing and go back to power saving state.
- The function should also cancel (finish) capturing if it haven't been completed yet.
- The hDevice handle might be obsolete and should be checked for validity.
- Different contexts (pParam values) correspond to devices accessed from different device managers. Consequently pParam should be checked to be equal to pParam passed to StartBiometricDeviceSequence [39], otherwise the device usages through different device managers will interfere.

4.1.3.4.2.9 Functions for fingerprint/palm scanners

Functions used by fingerprint/palm scanners (NFScanner unit).

4.1.3.4.2.9.1 GetFScannerSupportedImpressionTypes

Returns the list of supported impression types.

```
NResult (N_CALLBACK GetFScannerSupportedImpressionTypes)(NHandle hDevice, NFImpressionType
* arValue, NInt valueLength);
```

Parameters

hDevice	[in] Handle to the device.
arValue	[out] Address of the caller allocated buffer where requested value should be saved.
valueLength	[in] The size of the given buffer.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NFScannerGetSupportedImpressionTypes is called.
- The function is Recommended for ndtFScanner. The default value of nfitLiveScanPlain or nfitLiveScanPalm (according to NDeviceType [1165]) will be used if the function is missing.
- The hDevice handle might be obsolete and should be checked for validity.

4.1.3.4.2.9.2 GetFScannerSupportedPositions

Returns the list of supported finger/palm positions.

```
NResult (N_CALLBACK GetFScannerSupportedPositions)(NHandle hDevice, NFPosition * arValue,
NInt valueLength);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

hDevice	[in] Handle to the device.
arValue	[out] Address of the called allocated buffer where requested value should be saved.
valueLength	[in] The size of the given buffer.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NFScannerGetSupportedPositions is called.
- The function is Recommended for ndtFScanner. The default value of nfpUnknown or nfpUnknownPalm (according to NDeviceType [1165]) will be used if the function is missing.
- The hDevice handle might be obsolete and should be checked for validity.

4.1.3.4.2.9.3 CaptureFScanner

Returns the image containing requested biometric data.

```
NResult (N_CALLBACK CaptureFScanner)(NHandle hDevice, NFImpressionType impressionType,
NFPosition position, const NFPosition * arMissingPositions, NInt missingPositionCount,
NBool automatic, NInt timeoutMilliseconds, NBiometricStatus * pStatus, HNFAttributes * *
parhObjects, NInt * pObjectCount, HNImage * phImage, NdmPreviewFScannerProc pPreview, void
* pParam);
```

Parameters

hDevice	[in] Handle to the device.
impressionType	[in] Requested impression type.
position	[in] Requested position value.
arMissingPositions	[in] Array of missing fingerprint positions (e.g. amputated). Parameter is used to mark missing fingers which will not be present in requested multi-finger scan (e.g. position == nfpPlainRightFourFingers).
missingPositionCount	[in] Count of missing positions in scanned image.
automatic	[in] Value which allows the scanner to decide about the image quality. When NTrue [1064] is set scanning device should checks image quality and decide when this image should be returned to the user. When NFalse [1041] is set user will control image quality and capturing process himself. In this case user will change the device status in pPreview callback function.
timeoutMilliseconds	[in] Maximum time (milliseconds) for the operation to complete (-1 to have no limit).
pStatus	[out] Status of the operation (after function returns).
parhObjects	[out] Array which contains information about objects (fingerprints, palmprint parts) in scanned image. The same objects should also be given to pPreview callback.
pObjectCount	[out] Object count in *parhObjects*.
phImage	[out] Captured image.
pPreview	[in] The callback function which allows the user to see/check non-final capturing data (see notes below).
pParam	[in] The capturing context. The value must also be passed to the callback.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

- The function will be invoked when API functions NFScannerCapture or NFScannerCaptureEx are called.
- The function is Mandatory for ndtFScanner.
- The hDevice handle might be obsolete and should be checked for validity.
- The function should try to capture image of satisfying quality with requested features for up to *timeoutMilliseconds* argument. However the operation should be stopped when user calls CancelBiometricDevice [39] function.
- The pStatus should show the status of the operation (was it successful/timeout/canceled).
- The parhObjects array should be allocated with NAlloc [504] and filled by using functions from NFAttributes unit.
- Different contexts (pParam values) correspond to devices accessed from different device managers. Consequently pParam should be remembered and checked in other capturing function, otherwise the device usages through different device managers will interfere.

Preview callback

Preview callback allows the user to get the intermediate capturing data. This can be handy to give the user feedback. So the user could improve the quality of the captured image or to just see the progress. Alternative usage of the preview function is when user controls the quality of the captured data (*automatic* argument of CaptureFScanner function set to NFalse [1041]). Therefore plugin writer should call the preview callback whenever the new scanned image or status from the device is received.

```
typedef void (N_CALLBACK NdmPreviewFScannerProc)(NHandle hDevice, HNImage hImage,
NBiometricStatus * pStatus, const HNFAttributes * arhObjects, NInt objectCount, void *
pParam);
```

Parameters

hDevice	[in] Handle to the device.
hImage	[in] Image captured so far.
pStatus	[in/out] Current status of the operation. The value can be changed by the user, the plugin writer should check the value after function returns. The capturing should be stopped if status is final (NBiometricStatusIsFinal [66] function).
parhObjects	[in] Array which contains information about objects (fingerprints, palmprint parts) in scanned image.
pObjectCount	[in] Object count in *parhObjects*.
pParam	[in] The capturing context. The value given to CaptureFScanner must be passed to the callback.

4.1.3.4.2.10 Functions for iris scanners

Functions used by iris scanners (NIrisScanner unit).

4.1.3.4.2.10.1 GetIrisScannerSupportedPositions

Returns the list of supported iris positions.

```
NResult (N_CALLBACK GetIrisScannerSupportedPositions)(NHandle hDevice, NEPosition *
arValue, NInt valueLength);
```

Parameters

hDevice	[in] Handle to the device.
arValue	[out] Address of the called allocated buffer where requested value should be saved.
valueLength	[in] The size of the given buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API function NIrisScannerGetSupportedPositions is called.
- The function is Recommended for ndtIrisScanner. The default value of nepUnknown will be used if the function is missing.
- The hDevice handle might be obsolete and should be checked for validity.

4.1.3.4.2.10.2 CaptureIrisScanner

Returns the image containing requested biometric data.

```
NResult (N_CALLBACK CaptureIrisScanner)(NHandle hDevice, NEPosition position, const
NEPosition * arMissingPositions, NInt missingPositionCount, NBool automatic, NInt
timeoutMilliseconds, NBiometricStatus * pStatus, HNEAttributes * * parhObjects, NInt *
pObjectCount, HNImage * phImage, NdmPreviewIrisScannerProc pPreview, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
impressionType	[in] Requested impression type.
position	[in] Requested position value.
arMissingPositions	[in] Array of missing eye positions (e.g. half-blind people without one eye).
missingPositionCount	[in] Count of missing positions in scanned image.
automatic	[in] Value which allows the scanner to decide about the image quality. When NTrue [1064] is set scanning device should checks image quality and decide when this image should be returned to the user. When NFalse [1041] is set user will control image quality and capturing process himself. In this case user will change the device status in *pPreview* callback function.
timeoutMilliseconds	[in] Maximum time (milliseconds) for the operation to complete (-1 to have no limit).
pStatus	[out] Status of the operation (after function returns).
parhObjects	[out] Array which contains information about objects (irises present in scanned image). The same objects should also be given to *pPreview* callback.
pObjectCount	[out] Object count in *parhObjects*.
phImage	[out] Captured image.
pPreview	[in] The callback function which allows the user to see/check non-final capturing data (see notes below).
pParam	[in] The capturing context. The value must also be passed to the callback.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

- The function will be invoked when API functions NIrisScannerCapture or NIrisScannerCaptureEx are called.
- The function is Mandatory for ndtIrisScanner.
- The hDevice handle might be obsolete and should be checked for validity.
- The function should try to capture image of satisfying quality with requested features for up to *timeoutMilliseconds* argument. However the operation should be stopped when user calls CancelBiometricDevice [39] function.
- The pStatus should show the status of the operation (was it successful/timeout/canceled).
- The parhObjects array should be allocated with NAlloc [504] and filled by using functions from NEAttributes unit.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- Different contexts (pParam values) correspond to devices accessed from different device managers. Consequently pParam should be remembered and checked in other capturing function, otherwise the device usages through different device managers will interfere. If the capture sequence is used, the value should also be checked to match one given to StartBiometricDeviceSequence [39].

Preview callback

Preview callback allows the user to get the intermediate capturing data. This can be handy to give the user feedback. So the operator could improve the quality of the captured image or to just see the progress. Alternative usage of the preview function is when user controls the quality of the captured data (*automatic* argument of CaptureIrisScanner function set to NFalse [1041]).

Therefore plugin writer should call the preview callback whenever the new scanned image or status from the device is received.

```
typedef void (N_CALLBACK NdmPreviewIrisScannerProc)(NHandle hDevice, HNImage hImage,
NBiometricStatus * pStatus, const HNEAttributes * arhObjects, NInt objectCount, void *
pParam);
```

Parameters

hDevice	[in] Handle to the device.
hImage	[in] Image captured so far.
pStatus	[in/out] Current status of the operation. The value can be changed by the user, the plugin writer should check the value after function returns. The capturing should be stopped if status is final (NBiometricStatusIsFinal [66] function).
parhObjects	[in] Array which contains information about objects (irises present in scanned image).
pObjectCount	[in] Object count in *parhObjects*.
pParam	[in] The capturing context. The value given to CaptureIrisScanner must be passed to the callback.

4.1.3.4.2.11 DeviceAdded

Invoked when device is added to each instance of device manager.

```
void (N_CALLBACK DeviceAdded)(NHandle hDevice, HNPlugin hPlugin, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
hPlugin	[in] Handle to current plugin.
pParam	[in] Pointer to device manager handle.

Remarks

The function is reserved for more complex plugins and can be omitted.

4.1.3.4.2.12 DeviceLost

```
void (N_CALLBACK DeviceLost)(NHandle hDevice, HNPlugin hPlugin, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
hPlugin	[in] Handle to current plugin.
pParam	[in] Pointer to device manager handle.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

The function is reserved for more complex plugins and can be omitted.

4.1.3.4.2.13 DeviceRemoved

```
void (N_CALLBACK DeviceRemoved)(NHandle hDevice, HNPlugin hPlugin, void * pParam);
```

Parameters

hDevice	[in] Handle to the device.
hPlugin	[in] Handle to current plugin.
pParam	[in] Pointer to device manager handle.

Remarks

The function is reserved for more complex plugins and can be omitted.

4.1.3.4.2.14 CanUse

```
NBool (N_CALLBACK CanUse)(HNPlugin hOtherPlugin);
```

Parameters

hOtherPlugin	[in] Handle to other plugin.

Return value

Function should return N_OK [433] on success or one of the error codes defined in NErrors.h.

Remarks

The function is reserved for more complex plugins and can be omitted.

4.2 Faces

Matching of multi-face templates is performed a little bit different than this is done for other modalities. Each face from the query template is matched with all the faces from the database template. Then the scores of individual comparisons are fused to one faces score (in other modalities, the best score is returned).

When composite templates with other modalities are used, faces can be used to perform selection before matching other modalities. It is recommended to use NMP_FACES_MATCHING_THRESHOLD_NEW [193] (NMatcher.FacesMatchingThresholdNew [1754]) parameter for setting threshold for this selection. This new parameter uses unified integer score values [13], and the older NMP_FACES_MATCHING_THRESHOLD [192] parameter uses obsolete floating point scores [45].

When faces are the only used modality, the NMP_MATCHING_THRESHOLD [195] (NMatcher.MatchingThreshold [1756]) parameters with integer score values [13] should be used.

4.2.1 Obsolete scores

This section describes obsolete floating point scores used by older face matchers: NlsMatcher [143], NLMatcher [128]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

and NMP_FACES_MATCHING_THRESHOLD [192] (NMatcher.FacesMatchingThreshold) parameter of NMatcher [171]. Threshold should be determined from table below:

FAR (false acceptance rate)	Matching threshold (score)
100 %	0
10 %	0.58
1 %	0.61
0.1 %	0.64
0.01 %	0.67
0.001 %	0.70
0.0001 %	0.73
0.00001 %	0.76
0.000001 %	0.79

Note

These scores are obsolete, instead use NMP_MATCHING_THRESHOLD [195] (NMatcher.MatchingThreshold [1756]) and NMP_FACES_MATCHING_THRESHOLD_NEW [193] (NMatcher.FacesMatchingThresholdNew [1754]) parameters with integer score values [13].

4.2.2 Face Feature Points

Neurotechnology face recognition algorithm can detect all frontal face features. The image below shows how these features are detected:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

This option can be enabled in face samples applications by selecting *Detect all facial features when extracting from file*.

These API parameters controls detection of all facial features:

- NLE_FEATURE_POINT_COUNT [120]
- NLEP_DETECT_ALL_FEATURE_POINTS [121] (ParameterDetectAllFeaturePoints [1709] for .NET and PARAMETER_DETECT_ALL_FEATURE_POINTS for Java)

If detection of all facial features is not required user can set NLEP_DETECT_BASE_FEATURE_POINTS [121] (ParameterDetectBaseFeaturePoints [1709] for .NEt or PARAMETER_DETECT_BASE_FEATURE_POINTS for Java). In this case only base face feature points are detected.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

5 Tutorials

This section contains short tutorials description.

Biometrics

Tutorial	Description
CreateMultiFaceTemplate	Creates NTemplate [219] that contains multiple faces templates (multiple NLRecord [134]).
CreateTokenFaceImage	Demonstrates creation of token face image.
DetectFacialFeatures	Demonstrates detection of face and facial features in the image.
EnrollFaceFromCamera	Demonstrates face feature extraction from camera.
EnrollFaceFromImage	Demonstrates how to write code for face enrollment to database from single image.
EnrollFaceFromStream	Demonstrates how to write code for face enrollment to database from stream (image sequence).
IdentifyFace	Demonstrates how to write code for face identification against database.
VerifyFace	Demonstrates how to match (verify) two faces templates.
MatchMultipleFaces	Demonstrates how to write code for face identification against image containing multiple faces.

Devices

Tutorial	Description
ImageCapture	Demonstrates capturing images from cameras.

Surveillance

Tutorial	Description
CountSubjects	This tutorial uses NSurveillance [1439] library to count the number of subjects passed through the field of view of the camera. Tutorial runs for a fixed number of time and prints intermediate information. On exit prints final statistics.
WatchList	This tutorial uses NSurveillance [1439] library to create a watch list and log all the subjects who passed through the field of view of the camera and are present in the watch list. Tutorial runs for a fixed number of minutes and prints to console a list of best matches for each identified subject.

All tutorials are written in C#, VB.NET and C programming languages. Almost all C tutorials are intended for Linux OS too.

Files

All tutorials are located in the VeriLook Surveillance 2.3 SDK `/Tutorials` folder.

Notes

If you have User Account Control (UAC) enabled on Windows Vista or Windows 7 you will not be able to compile Tutorials when installation folder is `Program Files` or `Program Files (x86)`. Please, select different installation location in this case.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

6 Samples

This chapter provides samples information.

Notes

If you have User Account Control (UAC) enabled on Windows Vista or Windows 7 you will not be able to compile Samples when installation folder is `Program Files` or `Program Files (x86)`. Please, select different installation location in this case.

6.1 C++ (WxWidgets)

C++ sample application is used to track subjects using live video streams (cameras or video files). Using this application subjects can be added to internal database and after that tracked.

This application application is started by launching `SurveillanceSampleWX.exe` file from `Bin\Win32_x86` directory. After application was started, *Choose Device* window is shown:

All installed cameras (web or IP cameras) are displayed in this window. You should select video device and resolution for this device in this window. If you want to use video files, select option *Video File Virtual Camera*.

After you have chosen your video capture device, you will see the main application window. The main application window has two-pane layout, where the left pane is used for displaying video stream and the right pane is used for displaying tracked subjects. Menu commands are used to control application.

The main VeriLook Surveillance 2.3 SDK window after 7 subjects were tracked:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Tracked subjects are displayed in the right pane. If the subject was enrolled to database previously, his Id is shown in brackets. All list items (found subjects) in the right panel can be expanded. When list items are expanded, the time of subject's appearance and disappearance is shown for each subject.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

New subjects are enrolled to database by selecting list item in the right panel and pressing button *Add template from selected node to watch list*. When adding templates to watch list, subject Id for located face should be entered.

Alternatively, new subjects can be enrolled from an image file or directory by clicking "File -> Enroll from Image(s)" or "File -> Enroll from Directory".

When the application detects an enrolled face, it will write template ID under the tracking rectangle.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Application settings

Application settings can be accessed by choosing menu item `Tools->Settings`. This allows to change Detection, Identification, Extraction and other application settings:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Detection

- *Min Inter-Ocular Distance* - minimum distance between eyes in a face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 40 (See also NSP_DETECTION_MIN_IOD [1463] and DetectionMinIod [1898]).
- *Max Inter-Ocular Distance* - maximum distance between eyes in a face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 4000 (See also NSP_DETECTION_MAX_IOD [1463] and DetectionMaxIod [1897]).
- *Face Confidence Threshold* - specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100]. The default value is 51 (See also NSP_DETECTION_FACE_CONFIDENCE_THRESHOLD [1463] and DetectionFaceConfidenceThreshold [1897]).
- *Max Roll Angle Deviation* - defines maximum roll angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180] (See also NSP_DETECTION_MAX_ROLL_ANGLE_DEVIATION [1459] and DetectionMaxRollAngleDeviation [1897]).
- *Face Roll Angle Base* - defines the base roll angle of the image.
- *Max Yaw Angle Deviation* - defines maximum roll angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180] (See also NSP_DETECTION_MAX_YAW_ANGLE_DEVIATION [1459] and DetectionMaxYawAngleDeviation [1898]).
- *Face Detection Scale* - specifies number of scales in one octave when constructing scale space for face detection.
- *Detect More Faces [45]* - specifies whether to use different face detection procedure, which has a higher recall rate.

Extraction

- *Face Quality Threshold* - specifies the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.
- *Max Records Per Template* - defines maximum number of records which can be saved in one template. The value of this parameter can be in range [1..16]. The default value is 5.
- *Enrollment Template Size* - size of face image templates. Can be used Large, Medium or Small template. It is recommended to use large template size. This parameter is used when generating a template from tracked subject.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- *Identification Template Size* - default size of face image templates when performing identification. Can be used Large, Medium or Small template. It is recommended to use medium template size.
- *Detect Gender* - If checked, the application will detect gender.

Identification

- *Matching Threshold* - specifies matching threshold. Value should be more than 0.
- *Max Best Match Count* - -Maximum length of the list which is returned by each identification. The default value is 10.

Misc

- *Max Thread Count* - specifies the number of concurrent surveillance threads. Recommended value is "number of logical processors on CPU" plus one. Must be in range [3..8]. The default value is 5.
- *Max Tree Node Count* - maximum number of items in tree node. Defines how many subjects should be displayed in the right panel. Default value is 10.
- *Display Time (in ms)* - specifies the display time.

Database Synchronization

- *Synchronize Subjects With Other Instances* - specifies whether the subjects should be synchronized with other instances.
- *Listen Port* - the listening port.
- *Instances* - list of instances.
- *Address* - database address.
- *Port* - the database port.

Notes

When camera returns insufficient number of frames per second to sample application, green tracking rectangle is not drawn. In this case display time parameter can be used to set time for displaying tracking rectangle.

6.1.1 wxWidgets Compilation

wxWidgets are useful for creating cross-platform GUI applications. They are used for recent C++ samples and algorithm demos in Neurotechnology products. wxWidgets library can be downloaded from http://www.wxwidgets.org/. Before using wxWidgets you should compile it.

wxWidgets compilation using command line tools

wxWidgets libraries also can be compiled using command line tool. Run these commands to compile 32 and 64 bit libraries:

32 bit Debug:

nmake /A /f makefile.vc UNICODE=1 USE_GDIPLUS=1 RUNTIME_LIBS=static CPPFLAGS=/MTd BUILD=debug

32bit Release:

nmake /A /f makefile.vc UNICODE=1 USE_GDIPLUS=1 RUNTIME_LIBS=static CPPFLAGS=/MT BUILD=release

64bit Debug:

nmake /A /f makefile.vc UNICODE=1 USE_GDIPLUS=1 TARGET_CPU=amd64 RUNTIME_LIBS=static CPPFLAGS=/MTd BUILD=debug

64bit Release:

nmake /A /f makefile.vc UNICODE=1 USE_GDIPLUS=1 TARGET_CPU=amd64 RUNTIME_LIBS=static CPPFLAGS=/MT BUILD=release

After these libraries were compiled Visual Studio include and library paths have to be setup. See information below.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

wxWidgets compilation using Visual Studio

To compile wxWidgets as a static library do the following steps (Microsoft Visual Studio is required):

1. Open solution file `C:\wxWidgets-2.8.10\build\msw\wx.dsw` (in case wxWidgets are located in C disk).
2. Select all projects and change C/C++/Code Generation/Runtime library to Multi-threaded.
3. Create 64 bit configuration:
   1. In Visual Studio's *Configuration manager* create **x64** *Active Solution platform* (Copy settings from Win32)
   2. To prevent platform collisions for each project in 64 bit **Unicode Release** and **Unicode Debug** configurations change following settings: in tabs "C\C++: Precompiled Headers" and "C\C++: Output Files" change "Precompiled header file", "ASM List Location", "Object File Name", "Program Database File Name" and "Librarian: Output File" to contain "\Win64_x64\" sub folder (e.g. vc_mswu\xxx.pch change to vc_mswu\Win64_x64\xxx.pch; ..\..\lib\vc_lib\wxmsw28u_adv.lib -> ..\..\lib\vc_lib\Win64_x64\wxmsw28u_adv.lib).
4. Open file `include\wx\msw\setup.h` and set `wxUSE_GRAPHICS_CONTEXT` value to 1, i.e. `#define wxUSE_GRAPHICS_CONTEXT 1`
5. Build Unicode Release and Unicode Debug configurations for both Win32 and x64 platforms.

(Compile wxWidgets and your applications using the same Visual Studio that was used for sample compilation (Visual Studio 2005 or later) otherwise it will lead to compilation errors)

Finally, Visual Studio include and library paths have to be setup. Go to Tools->Options->Projects and Solutions->VC++ Directories and include these directories and library file from these directories:

- C:\wxWidgets-2.8.10\include
- C:\wxWidgets-2.8.10\include\msvc

Lib (Win32):

- C:\wxWidgets-2.8.10\lib\vc_lib

Lib (x64):

- C:\wxWidgets-2.8.10\lib\vc_lib\Win64_x64\

6.2 Devices

Devices sample applications are used to demonstrate how to use NDevices [1178] library functionality. Source code (C# and VB.NET programming languages) for Device samples is saved in `Samples\Devices` directory.

6.2.1 NdmSample - Adding support of custom device to NDeviceManager

NdmSample is example of how to write NDeviceManger plugin. It demonstrates how to extend NDeviceManger with support of custom devices, such that they can be used from Neurotechnology API the same way as any other supported device.

Trying it out

Source code for this sample is saved in `Samples\Devices\C\NdmSample` directory and can be compiled with Visual Studio project located in the same directory. The compiled plugin library will be saved into `NdmSample.dll` in `Bin\Win64_x64\` or `Bin\Win32_x86` directories (depending on selected build configuration).

NdmSample supports interfaces for microphone, video camera, fingerprint and iris scanners. Provided devices are virtual,

Copyright © 2009-2014 Neurotechnology. All rights reserved.

they are not tied to any specific hardware, instead when device is read (captured) the data from the sound, video or image file is returned. To use sample plugin the following files should be present in the compiled plugin directory:

- `NdmSampleMicrophone.wav` – Sample microphone;
- `NdmSampleCamera.avi` – Sample camera;
- `NdmSampleFingerScanner.png` – Sample fingerprint scanner;
- `NdmSampleIrisScannerLeft.png` and `NdmSampleIrisScannerRight.png` – Sample iris scanner.

When mentioned files are created, the sample plugin can be used as device from any Neurotechnology application. For the test it is recommended to use `DevicesSampleCS.exe`, as this is simple GUI application that allows to enumerate devices and perform capturing.

Customizing

The source code of the example can be used as reference for implementing support for custom device.

Please see the device manager interface documentation [21] and the comments in the source file (NdmSample.c). One can also find the list of needed interface functions in `Includes\NdmInterface.h`.

6.3 .NET

C# sample application is used to track subjects using live video streams (cameras or video files) or image files. Using this application subjects can be added to internal database and after that tracked.

This sample program demonstrates *VeriLook Surveillance 2.3 SDK* functionality:

- Detects all the faces.
- Tracks all detected faces.
- Performs real-time identification of the tracked faces.
- Logs all surveillance events in the right panel.
- Allows to add faces which disappear from camera's field of view to watch list in real time.
- Allows to modify all NSurveillance [1439] parameters.
- Allows to review the faces in watch list.
- Allows to enroll subjects from files.

This application application is started by launching `SurveillanceSampleCS.exe` file from `Bin\Win32_x86` or `Bin\Win64_x64` directories. After application was started, *Choose Device* window is shown:

All installed cameras (web or IP cameras) are displayed in this window. You should select video device and resolution for this device in this window.

After you have chosen your video capture device, you will see the main application window. The main application window

Copyright © 2009-2014 Neurotechnology. All rights reserved.

has two-pane layout, where the left pane is used for displaying video stream and the right pane is used for displaying tracked subjects. Menu commands are used to control application.

The main window after 3 subjects were tracked:

Tracked subjects (only the last 10) are displayed in the right pane. If the subject was enrolled to database previously, his Id is shown in brackets. All list items (found subjects) in the right panel can be expanded. When list items are expanded, the time of subject's appearance and disappearance is shown for each subject.

New subjects are enrolled to database by selecting list item in the right panel and pressing button *Add template from selected node to watch list*. When adding templates to watch list, subject Id for located face should be entered.

Application settings

Sample settings can be accessed by choosing menu item Tools->Settings. This allows to change Detection, Identification, Extraction and other application settings:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Detection

- *Min Inter-Ocular Distance* - minimum distance between eyes in a face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 40 (See also NSP_DETECTION_MIN_IOD [1463] and DetectionMinIod [1898]).
- *Max Inter-Ocular Distance* - maximum distance between eyes in a face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 4000 (See also NSP_DETECTION_MAX_IOD [1463] and DetectionMaxIod [1897]).
- *Face Confidence Threshold* - specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100]. The default value is 51 (See also NSP_DETECTION_FACE_CONFIDENCE_THRESHOLD [1463] and DetectionFaceConfidenceThreshold [1897]).
- *Max Roll Angle Deviation* - defines maximum roll angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180] (See also NSP_DETECTION_MAX_ROLL_ANGLE_DEVIATION [1459] and DetectionMaxRollAngleDeviation [1897]).

Extraction

- *Face Quality Threshold* - specifies the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.
- *Max Records Per Template* - defines maximum number of records which can be saved in one template. The value of this parameter can be in range [1..16]. The default value is 5.
- *Enrollment Template Size* - size of face image templates. Can be used Large, Medium or Small template. It is recommended to use large template size. This parameter is used when generating a template from tracked subject.
- *Identification Template Size* - default size of face image templates when performing identification. Can be used Large, Medium or Small template. It is recommended to use medium template size.

Identification

- *Matching Threshold* - specifies matching threshold. Value should be more than 0.
- *Max Best Match Count* - -Maximum length of the list which is returned by each identification. The default value is 10.

Misc

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- *Max Thread Count* - specifies the number of concurrent surveillance threads. Minimal thread count should be N*3 + 2 where N is running cameras count The default value is 5 (when one camera is running).
- *Flip Image* - mirrors horizontally image received from video camera or video file.
- *Max Tree Node Count* - maximum number of items in tree node. Defines how many subjects should be displayed in the right panel. Default value is 10.
- *Display time* - how long tracked face rectangle is displayed after face disappeared. This parameter is useful when camera returns insufficient number of frames per second to sample application but green tracking rectangle should be drawn.

Database synchronization

- *Synchronize subjects with other instances* - synchronizes tracked subjects with other Surveillance clients. Note: Surveillance clients should be started on different computers.
- *Instances* - other VeriLook Surveillance clients defined by *address* and *port*.

Files

Binaries of sample are located:

- `\bin\Win32_x86\SurveillanceSampleCS.exe - C# sample`
- `\bin\Win32_x86\SurveillanceSampleVB.exe - VB.NET sample`

Source are located:

- `\Samples\Surveillance\CS\SurveillanceSampleCS`
- `\Samples\Surveillance\VB\SurveillanceSampleVB`

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7 API Reference

VeriLook Surveillance 2.3 SDK consists of:

- NBiometrics [60]
- NDevices [1124]
- NMedia [1196]
- NSurveillance [1439]

main libraries and libraries wrappers for .Net:

- Neurotec.Biometrics [1654]
- Neurotec.Devices [1792]
- Neurotec.Media
- Neurotec.Surveillance [1883]

NCore [257] and NLicensing (Neurotec, Neurotec.Licensing [1851] in .NET) are auxiliary libraries that provides infrastructure.

Modules

Name	Description
.NET Reference [1468]	In this section *VeriLook Surveillance 2.3 SDK* .NET programming reference for API is provided. The required libraries for namespaces are listed in the table bellow:

7.1 C Reference

In this section *VeriLook Surveillance 2.3 SDK* C programming reference for API is provided.

7.1.1 NBiometrics Library

Provides the main functionality for matching, extracting or comparing biometrical data.

Remarks

Requirements (Windows)

- **Import Library**: NBiometrics.dll.lib
- **DLLs**: *NBiometrics.dll*

Requirements (Linux)

- **Shared objects**: *libNBiometrics.so*

Modules

Name	Description
NBiometricTypes Unit [61]	Defines biometric types used by Neurotechnology components.
NExtractors Unit [103]	Provides functions for retrieving NExtractors Library information.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLAttributes Unit [103]	Provides functions to retrieve face attributes (metadata .which is not saved in face template).
NLExtractor Unit [104]	NLExtractor is a part of NBiometrics library intended for human face finding in pictures and extracting their features into Neurotechnology Faces [45] Template (NLTemplate [153]). NLExtractor can detect human gender, expression, glasses, dark glasses, blink eyes and mouth open.
NLMatcher Unit [128]	Provides functionality to identify or verify face feature templates (NLRecord [134]).
NLMatchingDetails Unit [134]	Provides functionality to retrieve face templates matching details.
NLRecord Unit [134]	Provides functionality for packing, unpacking and editing Neurotechnology Face Records (NLRecords).
NlsMatcher Unit [143]	Provides functionality for comparing Neurotechnology Face Templates (NLTemplates), encapsulated in Neurotechnology Face Matcher (NlsMatcher) object.
NLTemplate Unit [153]	Provides functionality for packing, unpacking and editing Neurotechnology Faces [45] Templates (NLTemplates).
NMatcher Unit [171]	Provides functionality for comparing all Neurotechnology Templates (NTemplates), encapsulated in Neurotechnology Matcher (NMatcher) object.
NMatchers Unit [200]	Provides functionality for retrieving information about NMatchers library.
NMatchingDetails Unit [200]	Provides functionality to retrieve Neurotechnology templates matching details (such as score, records count, etc.).
NTemplate Unit [219]	Provides functionality for packing, unpacking and editing Neurotechnology Templates (NTemplates).
NTemplates Unit [256]	Provides functionality for retrieving information about NTemplates library.

7.1.1.1 NBiometricTypes Unit

Defines biometric types used by Neurotechnology components.

Files

Name	Description
NBiometricTypes.h [100]	Header file for NBiometricType [89] module.

Functions

	Name	Description
	NBiometricAngleFromRadians [65]	Converts the specified angle from radians to degrees.
	NBiometricAngleToRadians [65]	Converts the specified angle from degrees to radians.
	NBiometricAngleToString [65]	Gets a string representation of the specified angle.
	NBiometricQualityIsValid [66]	Checks whether the specified quality is valid.
	NBiometricQualityToString [66]	Gets a string representation of the specified quality.
	NBiometricStatusIsFinal [66]	Checks if the value passed as a parameter is denotes completed biometric operation rather then intermediate status of operation in progress.
	NBiometricStatusIsValid [67]	Checks if the value passed as a parameter is defined in NBiometricStatus [86].
	NBiometricSubtypeIsValid [67]	Checks if the value passed as a parameter is a valid NBiometricSubtype [88] value.
	NBiometricTypeIsValid [67]	Checks if the value passed as a parameter is a valid NBiometricType [89] value.
	NEPositionGetAvailableParts [67]	Determines available iris positions based on specified missing iris positions and base position.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NEPositionIsKnown [68]	Checks whether the specified position is known.
	NEPositionIsLeft [68]	Checks whether the specified position is left.
	NEPositionIsOneOf [69]	Checks if specified NEPosition [91] value matches one in an array.
	NEPositionIsRight [69]	Checks whether the specified position is right.
	NEPositionIsRightAndLeft [69]	Checks whether the specified position is both left and right.
	NEPositionIsSingleEye [70]	Checks whether the specified position represents a single eye.
	NEPositionIsTheEye [70]	Checks whether the specified position represents a separate eye.
	NEPositionIsTwoEyes [70]	Checks whether the specified position represents two eyes.
	NEPositionIsValid [70]	Checks if the value passed as a parameter is a valid NEPosition [91] value.
	NEPositionIsValidTheEye [71]	Checks if the specified value is a valid NEPosition [91] value.
	NFCoreToString [71]	Gets a string representation of the core.
	NFDeltaToString [71]	Gets a string representation of the delta.
	NFDoubleCoreToString [72]	Gets a string representation of the double core.
	NFImpressionTypeIsCompatibleWith [72]	Checks if two specified values are compatible.
	NFImpressionTypeIsContact [73]	Checks whether the specified NFImpressionType [91] value is one of the following: nfitLiveScanOpticalContactPlain, nfitLiveScanOpticalContactRolled, nfitLiveScanNonOpticalContactPlain, nfitLiveScanNonOpticalContactRolled
	NFImpressionTypeIsContactless [73]	Checks if the specified NFImpressionType [91] value is one of the following: nfitLiveScanContactless, nfitLiveScanOpticalContactlessPlain, nfitLiveScanOpticalContactlessRolled, nfitLiveScanNonOpticalContactlessPlain, nfitLiveScanNonOpticalContactlessRolled.
	NFImpressionTypeIsFinger [73]	Checks whether the specified value is fingerprint.
	NFImpressionTypeIsGeneric [74]	Checks whether the specified impression type is generic.
	NFImpressionTypeIsLatent [74]	Checks if the specified value is latent fingerprint or palmprint.
	NFImpressionTypeIsLiveScan [74]	Checks if the specified value is a live scan.
	NFImpressionTypeIsNonliveScan [74]	Checks if the specified value is a non-live scan.
	NFImpressionTypeIsNonOptical [75]	Checks if the specified NFImpressionType [91] value is one of the following: nfitLiveScanNonOpticalContactPlain, nfitLiveScanNonOpticalContactRolled, nfitLiveScanNonOpticalContactlessPlain, nfitLiveScanNonOpticalContactlessRolled.
	NFImpressionTypeIsOneOf [75]	Checks if specified NFImpressionType [91] value matches one in an array.
	NFImpressionTypeIsOptical [76]	Checks if the specified NFImpressionType [91] value is one of the following: nfitLiveScanOpticalContactPlain, nfitLiveScanOpticalContactRolled, nfitLiveScanOpticalContactlessPlain, nfitLiveScanOpticalContactlessRolled.
	NFImpressionTypeIsPalm [76]	Checks if the specified value is a palmprint.
	NFImpressionTypeIsPlain [76]	Checks if the specified value is a plain scan or optical contact/contactless.
	NFImpressionTypeIsRolled [76]	Checks if the specified value is one of rolled biometric types.
	NFImpressionTypeIsSwipe [77]	Checks if the specified NFImpressionType [91] value is one of the following: nfitLiveScanVerticalSwipe.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NFImpressionTypeIsValid [77]	Checks if the value passed as a parameter is a valid NFImpressionType [91] value.
	NFImpressionTypeIsValidFinger [77]	Checks if the specified value is a valid fingerprint.
	NFImpressionTypeIsValidPalm [78]	Checks if the specified value is a valid palmprint.
	NFImpressionTypeToFinger [78]	Converts an impression type object into finger impression type object.
	NFMinutiaNeighborToString [78]	Gets a string representation of the minutia neighbor.
	NFMinutiaToString [79]	Gets a string representation of the minutia.
	NFPositionGetAvailableParts [79]	Determines available finger positions based on specified missing finger positions and base position.
	NFPositionIsCompatibleWith [80]	Checks if two specified values are compatible.
	NFPositionIsCompatibleWithImpressionType [80]	Checks whether the specified position is compatible with the specified impression type.
	NFPositionIsFinger [80]	Checks if the specified value is fingerprint.
	NFPositionIsFourFingers [81]	Checks if the specified value is a type of 4 fingerprints data.
	NFPositionIsKnown [81]	Checks if the record contains biometric data which type is one of the known fingerprint types (thumb, palmprint, etc.).
	NFPositionIsLeft [81]	Checks if the specified position is left fingerprint.
	NFPositionIsLeftAndRight [81]	Checks if the specified position is left and right fingerprint.
	NFPositionIsOneOf [82]	Checks if specified NFPosition [92] value matches one in an array.
	NFPositionIsPalm [82]	Checks if the specified value is a palmprint.
	NFPositionIsRight [83]	Checks if the specified position is right fingerprint.
	NFPositionIsSingleFinger [83]	Checks if the specified value is a single fingerprint.
	NFPositionIsTheFinger [83]	Checks if the specified value is the fingerprint.
	NFPositionIsThePalm [83]	Checks whether the specified position represents a separate palm.
	NFPositionIsThreeFingers [84]	Checks if the specified value is a type of 3 fingerprints data.
	NFPositionIsTwoFingers [84]	Checks if the specified value is a type of 2 fingerprints data.
	NFPositionIsValid [84]	Checks if the value passed as a parameter is a valid NFPosition [92] value.
	NFPositionIsValidFinger [85]	Checks if the specified value is a valid fingerprint.
	NFPositionIsValidPalm [85]	Checks if the specified value is a valid palmprint.
	NFPositionIsValidTheFinger [85]	Checks if the specified value is a valid fingerprint.
	NLFeaturePointToString [86]	Gets a string representation of the feature point.

Macros

Name	Description
N_BIOMETRIC_QUALITY_FAILED [97]	Failed quality.
N_BIOMETRIC_QUALITY_MAX [97]	Maximal quality.
N_BIOMETRIC_QUALITY_MIN [97]	Minimal quality.
N_BIOMETRIC_QUALITY_UNKNOWN [98]	Unknown quality.
N_PHRASE_ID_UNKNOWN [98]	Unknown phrase ID.
NBiometricAngleToString [98]	Picks a Unicode (W) or ANSCII (A) version of the NBiometricAngleToString function.
NBiometricQualityToString [98]	Picks a Unicode (W) or ANSCII (A) version of the NBiometricQualityToString function.
NFCoreToString [98]	Picks a Unicode (W) or ANSCII (A) version of the NFCoreToString function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFDeltaToString [99]	Picks a Unicode (W) or ANSCII (A) version of the NFDeltaToString function.
NFDoubleCoreToString [99]	Picks a Unicode (W) or ANSCII (A) version of the NFDoubleCoreToString function.
NFMinutiaNeighborToString [99]	Picks a Unicode (W) or ANSCII (A) version of the NFMinutiaNeighborToString function.
NFMinutiaToString [99]	Picks a Unicode (W) or ANSCII (A) version of the NFMinutiaToString function.
NLFeaturePointToString [99]	Picks a Unicode (W) or ANSCII (A) version of the NLFeaturePointToString function.

Structs, Records, Enums

	Name	Description
	NBiometricStatus_ [86]	Enumerates biometric status values. These values are returned by NBiometricDevice [1142].
	NBiometricSubtype_ [88]	Enumerates values which define the type of biometric data in the record. The values in this enumeration more specifically define the type of biometric data (NBiometricType [89]) in the record.
	NBiometricType_ [89]	Enumerates types of biometric technology used in a record.
	NEPosition_ [91]	Enumerates eye position values.
	NFImpressionType_ [91]	Specifies the impression type.
	NFPosition_ [92]	Specifies the finger position.
	NGender_ [95]	Specifies the gender.
	NLExpression_ [95]	Specifies the facial expression.
	NLFeaturePoint_ [86]	Enumerates detected facial feature point values.
	NLProperties_ [96]	Specified the facial properties.
	NBiometricStatus [86]	Enumerates biometric status values. These values are returned by NBiometricDevice [1142].
	NBiometricSubtype [88]	Enumerates values which define the type of biometric data in the record. The values in this enumeration more specifically define the type of biometric data (NBiometricType [89]) in the record.
	NBiometricType [89]	Enumerates types of biometric technology used in a record.
	NEPosition [91]	Enumerates eye position values.
	NFImpressionType [91]	Specifies the impression type.
	NFPosition [92]	Specifies the finger position.
	NGender [95]	Specifies the gender.
	NLExpression [95]	Specifies the facial expression.
	NLProperties [96]	Specified the facial properties.

Types

Name	Description
NLFeaturePoint [97]	Represents a facial feature point in a NLRecord [134].

7.1.1.1.1 Functions

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.1.1 NBiometricAngleFromRadians Function

Converts the specified angle from radians to degrees.

C++

```
NInt N_API NBiometricAngleFromRadians(NDouble value);
```

Parameters

Parameters	Description
NDouble value	[in] Radian value.

Returns

Degree value.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.2 NBiometricAngleToRadians Function

Converts the specified angle from degrees to radians.

C++

```
NDouble N_API NBiometricAngleToRadians(NInt value);
```

Parameters

Parameters	Description
NInt value	[in] Degree value.

Returns

Radian value.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.3 NBiometricAngleToString Function

Gets a string representation of the specified angle.

C++

```
NResult N_API NBiometricAngleToString(NInt value, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
NInt value	[in] Angle in degrees.
const NChar * szFormat	[in] Reserved.
HNString * phValue	[out] Pointer to a handle to the string representing the angle.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricTypes Unit [61]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.1.4 NBiometricQualityIsValid Function

Checks whether the specified quality is valid.

C++

```
NBool N_API NBiometricQualityIsValid(NByte value);
```

Parameters

Parameters	Description
NByte value	[in] The biometric quality to be checked.

Returns

NBoolean [1004] value - NTrue [1064] or NFalse [1041].

Module

NBiometricTypes Unit [61]

7.1.1.1.1.5 NBiometricQualityToString Function

Gets a string representation of the specified quality.

C++

```
NResult N_API NBiometricQualityToString(NByte value, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
NByte value	Quality value.
const NChar * szFormat	[in] Reserved.
HNString * phValue	[out] Pointer to a handle to the string representing the angle.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.6 NBiometricStatusIsFinal Function

Checks if the value passed as a parameter is denotes completed biometric operation rather then intermediate status of operation in progress.

C++

```
NBool N_API NBiometricStatusIsFinal(NBiometricStatus value);
```

Parameters

Parameters	Description
NBiometricStatus value	[in] One of NBiometricStatus [86] values to check.

Returns

NBool [1004] type value.

Module

NBiometricTypes Unit [61]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.1.7 NBiometricStatusIsValid Function

Checks if the value passed as a parameter is defined in NBiometricStatus [86].

C++

```
NBool N_API NBiometricStatusIsValid(NBiometricStatus value);
```

Parameters

Parameters	Description
NBiometricStatus value	[in] One of NBiometricStatus [86] values to check.

Returns

NBool [1004] type value - boolean value value which indicates status validation.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.8 NBiometricSubtypeIsValid Function

Checks if the value passed as a parameter is a valid NBiometricSubtype [88] value.

C++

```
NBool N_API NBiometricSubtypeIsValid(NBiometricSubtype value);
```

Parameters

Parameters	Description
NBiometricSubtype value	[in] One of NBiometricSubtype [88] values to check.

Returns

NBool [1004] type value - boolean value value which indicates subtype validation.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.9 NBiometricTypeIsValid Function

Checks if the value passed as a parameter is a valid NBiometricType [89] value.

C++

```
NBool N_API NBiometricTypeIsValid(NBiometricType value);
```

Parameters

Parameters	Description
NBiometricType value	[in] One of NBiometricType [89] values to check.

Returns

NBool [1004] type value - boolean value value which indicates biometric type validation.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.10 NEPositionGetAvailableParts Function

Determines available iris positions based on specified missing iris positions and base position.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NEPositionGetAvailableParts(NEPosition value, const NEPosition *
arMissingPositions, NInt missingPositionCount, NEPosition * arResults, NInt resultsLength);
```

Parameters

Parameters	Description
NEPosition value	[in] NEPosition [91] object, representing the general position.
const NEPosition * arMissingPositions	[in] Pointer to an array of NEPosition [91] objects, representing missing position parts.
NInt missingPositionCount	[in] Number of missing positions.
NEPosition * arResults	[out] Pointer to an array of NEPosition [91] objects, representing available position parts.
NInt resultsLength	[in] Length of memory buffer where item value is stored.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.11 NEPositionIsKnown Function

Checks whether the specified position is known.

C++

```
NBool N_API NEPositionIsKnown(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	[in] A position to be checked.

Returns

NBoolean [1004] value - NTrue [1064] or NFalse [1041].

Module

NBiometricTypes Unit [61]

7.1.1.1.1.12 NEPositionIsLeft Function

Checks whether the specified position is left.

C++

```
NBool N_API NEPositionIsLeft(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	[in] The position to be checked.

Returns

NBoolean [1004] value - NTrue [1064] or NFalse [1041].

Module

NBiometricTypes Unit [61]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.1.13 NEPositionIsOneOf Function

Checks if specified NEPosition [91] value matches one in an array.

C++

```
NResult N_API NEPositionIsOneOf(NEPosition value, NEPosition * arSupportedPositions, NInt supportedPositionCount, NBool * pResult);
```

Parameters

Parameters	Description
NEPosition value	[in] One of the NEPosition [91] values to check.
NEPosition * arSupportedPositions	[in] NEPosition [91] values array.
NInt supportedPositionCount	[in] Length of `arSupportedPositions` array.
NBool * pResult	[out] Check result (NTrue [1064] or NFalse [1041]).

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.14 NEPositionIsRight Function

Checks whether the specified position is right.

C++

```
NBool N_API NEPositionIsRight(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	[in] The position to be checked.

Returns

NBoolean [1004] value - NTrue [1064] or NFalse [1041].

Module

NBiometricTypes Unit [61]

7.1.1.1.1.15 NEPositionIsRightAndLeft Function

Checks whether the specified position is both left and right.

C++

```
NBool N_API NEPositionIsRightAndLeft(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	[in] The position to be checked.

Returns

NBoolean [1004] value - NTrue [1064] or NFalse [1041].

Module

NBiometricTypes Unit [61]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.1.16 NEPositionIsSingleEye Function

Checks whether the specified position represents a single eye.

C++

```
NBool N_API NEPositionIsSingleEye(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	[in] The position to be checked.

Returns

NBoolean [1004] value - NTrue [1064] or NFalse [1041].

Module

NBiometricTypes Unit [61]

7.1.1.1.1.17 NEPositionIsTheEye Function

Checks whether the specified position represents a separate eye.

C++

```
NBool N_API NEPositionIsTheEye(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	[in] The position to be checked.

Returns

NBoolean [1004] value - NTrue [1064] or NFalse [1041].

Module

NBiometricTypes Unit [61]

7.1.1.1.1.18 NEPositionIsTwoEyes Function

Checks whether the specified position represents two eyes.

C++

```
NBool N_API NEPositionIsTwoEyes(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	[in] The position to be checked.

Returns

NBoolean [1004] value - NTrue [1064] or NFalse [1041].

Module

NBiometricTypes Unit [61]

7.1.1.1.1.19 NEPositionIsValid Function

Checks if the value passed as a parameter is a valid NEPosition [91] value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NBool N_API NEPositionIsValid(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	[in] One of NEPosition [91] values to check.

Returns

NBool [1004] type value - boolean value value which indicates biometric type validation.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.20 NEPositionIsValidTheEye Function

Checks if the specified value is a valid NEPosition [91] value.

C++

```
NBool N_API NEPositionIsValidTheEye(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	[in] NEPosition [91] value to check.

Returns

NBool [1004] type value - boolean value value which indicates biometric type validation.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.21 NFCoreToString Function

Gets a string representation of the core.

C++

```
NResult N_API NFCoreToString(const NFCore * pValue, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
const NFCore * pValue	[in] Pointer to the core.
const NChar * szFormat	[in] Reserved.
HNString * phValue	[out] Pointer to a handle to the string representing the core.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.22 NFDeltaToString Function

Gets a string representation of the delta.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NFDeltaToString(const NFDelta * pValue, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
const NFDelta * pValue	[in] Pointer to the delta.
const NChar * szFormat	[in] Reserved.
HNString * phValue	[out] Pointer to a handle to the string representing the delta.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.23 NFDoubleCoreToString Function

Gets a string representation of the double core.

C++

```
NResult N_API NFDoubleCoreToString(const NFDoubleCore * pValue, const NChar * szFormat,
HNString * phValue);
```

Parameters

Parameters	Description
const NFDoubleCore * pValue	[in] Pointer to the double core.
const NChar * szFormat	[in] Reserved.
HNString * phValue	[out] Pointer to a handle to the string representing the double core.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.24 NFImpressionTypeIsCompatibleWith Function

Checks if two specified values are compatible.

C++

```
NBool N_API NFImpressionTypeIsCompatibleWith(NFImpressionType value, NFImpressionType
otherValue);
```

Parameters

Parameters	Description
NFImpressionType value	[in] Value to check if it is compatible with other value.
NFImpressionType otherValue	[in] Value which is used to check compatibility with *value*.

Returns

NBool [1004] type value. NTrue [1064] - if types are compatible. NFalse [1041] - if types are incompatible.

Module

NBiometricTypes Unit [61]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.1.25 NFImpressionTypeIsContact Function

Checks whether the specified NFImpressionType [91] value is one of the following: nfitLiveScanOpticalContactPlain, nfitLiveScanOpticalContactRolled, nfitLiveScanNonOpticalContactPlain, nfitLiveScanNonOpticalContactRolled

C++

```
NBool N_API NFImpressionTypeIsContact(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] value to check.

Returns

NBool [1004] type value. NTrue [1064] - if type is contact. NFalse [1041] -otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.26 NFImpressionTypeIsContactless Function

Checks if the specified NFImpressionType [91] value is one of the following: nfitLiveScanContactless, nfitLiveScanOpticalContactlessPlain, nfitLiveScanOpticalContactlessRolled, nfitLiveScanNonOpticalContactlessPlain, nfitLiveScanNonOpticalContactlessRolled.

C++

```
NBool N_API NFImpressionTypeIsContactless(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] value to check.

Returns

NBool [1004] type value. NTrue [1064] - if type is contactless. NFalse [1041] -otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.27 NFImpressionTypeIsFinger Function

Checks whether the specified value is fingerprint.

C++

```
NBool N_API NFImpressionTypeIsFinger(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if type is a record contains fingerprint data. NFalse [1041] -otherwise.

Module

NBiometricTypes Unit [61]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.1.28 NFImpressionTypeIsGeneric Function

Checks whether the specified impression type is generic.

C++

```
NBool N_API NFImpressionTypeIsGeneric(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] The impression type to be checked.

Returns

NBoolean [1004] value - NTrue [1064] or NFalse [1041].

Module

NBiometricTypes Unit [61]

7.1.1.1.1.29 NFImpressionTypeIsLatent Function

Checks if the specified value is latent fingerprint or palmprint.

C++

```
NBool N_API NFImpressionTypeIsLatent(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if type is a record contains latent fingerprint or palmprint data. NFalse [1041] -otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.30 NFImpressionTypeIsLiveScan Function

Checks if the specified value is a live scan.

C++

```
NBool N_API NFImpressionTypeIsLiveScan(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if type is a record contains a live scan. NFalse [1041] -otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.31 NFImpressionTypeIsNonliveScan Function

Checks if the specified value is a non-live scan.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NBool N_API NFImpressionTypeIsNonliveScan(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if type is a record contains a non-live scan. NFalse [1041] - otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.32 NFImpressionTypeIsNonOptical Function

Checks if the specified NFImpressionType [91] value is one of the following: nfitLiveScanNonOpticalContactPlain, nfitLiveScanNonOpticalContactRolled, nfitLiveScanNonOpticalContactlessPlain, nfitLiveScanNonOpticalContactlessRolled.

C++

```
NBool N_API NFImpressionTypeIsNonOptical(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] value to check.

Returns

NBool [1004] type value. NTrue [1064] - if type is non-optical. NFalse [1041] - otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.33 NFImpressionTypeIsOneOf Function

Checks if specified NFImpressionType [91] value matches one in an array.

C++

```
NResult N_API NFImpressionTypeIsOneOf(NFImpressionType value, NFImpressionType *
arSupportedImpressionTypes, NInt supportedImressionTypeCount, NBool * pResult);
```

Parameters

Parameters	Description
NFImpressionType value	[in] One of the NFImpressionType [91] values to check.
NFImpressionType * arSupportedImpressionTypes	[in] NFImpressionType [91] values array.
NInt supportedImressionTypeCount	[in] Length of `arSupportedImpressionTypes` array.
NBool * pResult	[out] Check result (NTrue [1064] or NFalse [1041]).

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NBiometricTypes Unit [61]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.1.34 NFImpressionTypeIsOptical Function

Checks if the specified NFImpressionType [91] value is one of the following: nfitLiveScanOpticalContactPlain, nfitLiveScanOpticalContactRolled, nfitLiveScanOpticalContactlessPlain, nfitLiveScanOpticalContactlessRolled.

C++

```
NBool N_API NFImpressionTypeIsOptical(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] value to check.

Returns

NBool [1004] type value. NTrue [1064] - if type is optical. NFalse [1041] - otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.35 NFImpressionTypeIsPalm Function

Checks if the specified value is a palmprint.

C++

```
NBool N_API NFImpressionTypeIsPalm(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if record contains palmprint data. NFalse [1041] -otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.36 NFImpressionTypeIsPlain Function

Checks if the specified value is a plain scan or optical contact/contactless.

C++

```
NBool N_API NFImpressionTypeIsPlain(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] value to check its type.

Returns

NBool [1004] type value.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.37 NFImpressionTypeIsRolled Function

Checks if the specified value is one of rolled biometric types.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NBool N_API NFImpressionTypeIsRolled(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] value to check its type.

Returns

NBool [1004] type value.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.38 NFImpressionTypeIsSwipe Function

Checks if the specified NFImpressionType [91] value is one of the following: nfitLiveScanVerticalSwipe.

C++

```
NBool N_API NFImpressionTypeIsSwipe(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] value to check.

Returns

NBool [1004] type value.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.39 NFImpressionTypeIsValid Function

Checks if the value passed as a parameter is a valid NFImpressionType [91] value.

C++

```
NBool N_API NFImpressionTypeIsValid(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] One of NFImpressionType [91] values to check.

Returns

NBool [1004] type value - boolean value value which indicates NFImpressionType [91] validation.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.40 NFImpressionTypeIsValidFinger Function

Checks if the specified value is a valid fingerprint.

C++

```
NBool N_API NFImpressionTypeIsValidFinger(NFImpressionType value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NFImpressionType value	[in] One of NFImpressionType [91] values to check.

Returns

NBool [1004] type value.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.41 NFImpressionTypeIsValidPalm Function

Checks if the specified value is a valid palmprint.

C++

```
NBool N_API NFImpressionTypeIsValidPalm(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] One of NFImpressionType [91] values to check.

Returns

NBool [1004] type value.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.42 NFImpressionTypeToFinger Function

Converts an impression type object into finger impression type object.

C++

```
NFImpressionType N_API NFImpressionTypeToFinger(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	[in] NFImpressionType [91] to be converted.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.43 NFMinutiaNeighborToString Function

Gets a string representation of the minutia neighbor.

C++

```
NResult N_API NFMinutiaNeighborToString(const NFMinutiaNeighbor * pValue, const NChar *
szFormat, HNString * phValue);
```

Parameters

Parameters	Description
const NFMinutiaNeighbor * pValue	[in] Pointer to the minutia neighbor.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const NChar * szFormat	[in] Reserved.
HNString * phValue	[out] Pointer to a handle to the string representing the minutia neighbor.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.44 NFMinutiaToString Function

Gets a string representation of the minutia.

C++

```
NResult N_API NFMinutiaToString(const NFMinutia * pValue, const NChar * szFormat, HNString
* phValue);
```

Parameters

Parameters	Description
const NFMinutia * pValue	[in] Pointer to the minutia.
const NChar * szFormat	[in] Reserved.
HNString * phValue	[out] Pointer to a handle to the string representing the minutia.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.45 NFPositionGetAvailableParts Function

Determines available finger positions based on specified missing finger positions and base position.

C++

```
NResult N_API NFPositionGetAvailableParts(NFPosition value, const NFPosition *
arMissingPositions, NInt missingPositionCount, NFPosition * arResults, NInt resultsLength);
```

Parameters

Parameters	Description
NFPosition value	[in] NFPosition [92] object, representing the general position.
const NFPosition * arMissingPositions	[in] Pointer to an array of NFPosition [92] objects, representing missing position parts.
NInt missingPositionCount	[in] Number of missing positions.
NFPosition * arResults	[out] Pointer to an array of NFPosition [92] objects, representing available position parts.
NInt resultsLength	[in] Length of memory buffer where item value is stored.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricTypes Unit [61]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.1.46 NFPositionIsCompatibleWith Function

Checks if two specified values are compatible.

C++

```
NBool N_API NFPositionIsCompatibleWith(NFPosition value, NFPosition otherValue);
```

Parameters

Parameters	Description
NFPosition value	[in] Value to check if it is compatible with other value.
NFPosition otherValue	[in] Value which is used to check compatibility with *value*.

Returns

NBool [1004] type value. NTrue [1064] - if types are compatible. NFalse [1041] - if types are incompatible.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.47 NFPositionIsCompatibleWithImpressionType Function

Checks whether the specified position is compatible with the specified impression type.

C++

```
NBool N_API NFPositionIsCompatibleWithImpressionType(NFPosition value, NFImpressionType
imp);
```

Parameters

Parameters	Description
NFPosition value	[in] The position to be checked.
NFImpressionType imp	[in] Impression type.

Returns

NBoolean [1004] value - NTrue [1064] or NFalse [1041].

Module

NBiometricTypes Unit [61]

7.1.1.1.1.48 NFPositionIsFinger Function

Checks if the specified value is fingerprint.

C++

```
NBool N_API NFPositionIsFinger(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] NFPosition [92] value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if type is a record contains fingerprint data. NFalse [1041] -otherwise.

Module

NBiometricTypes Unit [61]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.1.49 NFPositionIsFourFingers Function

Checks if the specified value is a type of 4 fingerprints data.

C++

```
NBool N_API NFPositionIsFourFingers(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] Value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if record contains four fingerprints data. NFalse [1041] - otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.50 NFPositionIsKnown Function

Checks if the record contains biometric data which type is one of the known fingerprint types (thumb, palmprint, etc.).

C++

```
NBool N_API NFPositionIsKnown(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] Value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if record contains biometric data of known fingerprint types.. NFalse [1041] - otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.51 NFPositionIsLeft Function

Checks if the specified position is left fingerprint.

C++

```
NBool N_API NFPositionIsLeft(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] Value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if record contains biometric data of left fingerprint.. NFalse [1041] - otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.52 NFPositionIsLeftAndRight Function

Checks if the specified position is left and right fingerprint.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NBool N_API NFPositionIsLeftAndRight(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] Fingerprint position to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if specified value is biometric data of left and right fingerprint type. NFalse [1041] - otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.53 NFPositionIsOneOf Function

Checks if specified NFPosition [92] value matches one in an array.

C++

```
NResult N_API NFPositionIsOneOf(NFPosition value, NFPosition * arSupportedPositions, NInt
supportedPositionCount, NBool * pResult);
```

Parameters

Parameters	Description
NFPosition value	[in] One of the NFPosition [92] values to check.
NFPosition * arSupportedPositions	[in] NFPosition [92] values array.
NInt supportedPositionCount	[in] Length of `arSupportedPositions` array.
NBool * pResult	[out] Check result (NTrue [1064] or NFalse [1041]).

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.54 NFPositionIsPalm Function

Checks if the specified value is a palmprint.

C++

```
NBool N_API NFPositionIsPalm(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] NFPosition [92] value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if specified value is a palmprint. NFalse [1041] -otherwise.

Module

NBiometricTypes Unit [61]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.1.55 NFPositionIsRight Function

Checks if the specified position is right fingerprint.

C++

```
NBool N_API NFPositionIsRight(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] Value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if specified position is right\ fingerprint. NFalse [1041] - otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.56 NFPositionIsSingleFinger Function

Checks if the specified value is a single fingerprint.

C++

```
NBool N_API NFPositionIsSingleFinger(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] NFPosition [92] value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if specified position is single fingerprint. NFalse [1041] - otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.57 NFPositionIsTheFinger Function

Checks if the specified value is the fingerprint.

C++

```
NBool N_API NFPositionIsTheFinger(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] NFPosition [92] value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if specified position the fingerprint. NFalse [1041] - otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.58 NFPositionIsThePalm Function

Checks whether the specified position represents a separate palm.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NBool N_API NFPositionIsThePalm(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] Position to be checked.

Returns

NBoolean [1004] value - NTrue [1064] or NFalse [1041].

Module

NBiometricTypes Unit [61]

7.1.1.1.1.59 NFPositionIsThreeFingers Function

Checks if the specified value is a type of 3 fingerprints data.

C++

```
NBool N_API NFPositionIsThreeFingers(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] Value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if record contains three fingerprints data. NFalse [1041] - otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.60 NFPositionIsTwoFingers Function

Checks if the specified value is a type of 2 fingerprints data.

C++

```
NBool N_API NFPositionIsTwoFingers(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] Value to check its type.

Returns

NBool [1004] type value. NTrue [1064] - if record contains two fingerprints data. NFalse [1041] - otherwise.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.61 NFPositionIsValid Function

Checks if the value passed as a parameter is a valid NFPosition [92] value.

C++

```
NBool N_API NFPositionIsValid(NFPosition value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NFPosition value	[in] One of NFPosition [92] values to check.

Returns

NBool [1004] type value - boolean value value which indicates biometric type validation.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.62 NFPositionIsValidFinger Function

Checks if the specified value is a valid fingerprint.

C++

```
NBool N_API NFPositionIsValidFinger(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] One of NFPosition [92] values to check.

Returns

NBool [1004] type value.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.63 NFPositionIsValidPalm Function

Checks if the specified value is a valid palmprint.

C++

```
NBool N_API NFPositionIsValidPalm(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] One of NFPosition [92] values to check.

Returns

NBool [1004] type value.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.64 NFPositionIsValidTheFinger Function

Checks if the specified value is a valid fingerprint.

C++

```
NBool N_API NFPositionIsValidTheFinger(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	[in] One of NFPosition [92] values to check.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NBool [1004] type value.

Module

NBiometricTypes Unit [61]

7.1.1.1.1.65 NLFeaturePointToString Function

Gets a string representation of the feature point.

C++

```
NResult N_API NLFeaturePointToString(const NLFeaturePoint * pValue, const NChar * szFormat,
HNString * phValue);
```

Parameters

Parameters	Description
const NLFeaturePoint * pValue	[in] Pointer to the feature point.
const NChar * szFormat	[in] Reserved.
HNString * phValue	[out] Pointer to a handle to the string representing the feature point.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricTypes Unit [61]

7.1.1.1.2 Structs, Records, Enums

7.1.1.1.2.1 NLFeaturePoint_ Structure

Enumerates detected facial feature point values.

C++

```
struct NLFeaturePoint_ {
  NUShort Code;
  NUShort X;
  NUShort Y;
  NByte Confidence;
};
```

Members

Members	Description
NUShort Code;	Facial feature point id.
NUShort X;	X coordinate of the point.
NUShort Y;	Y coordinate of the point.
NByte Confidence;	Detection confidence (0 – not available, [1..100] – detected).

Module

NBiometricTypes Unit [61]

7.1.1.1.2.2 NBiometricStatus Enumeration

Enumerates biometric status values. These values are returned by NBiometricDevice [1142].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
typedef enum NBiometricStatus_ {
  nbsNone = 0,
  nbsOk = 1,
  nbsCanceled = 2,
  nbsTimeout = 3,
  nbsCleaningNeeded = 10,
  nbsObjectsNotRemoved = 20,
  nbsObjectMissing = 21,
  nbsObjectNotFound = 22,
  nbsTooFewObjects = 23,
  nbsTooManyObjects = 24,
  nbsBadObjectSequence = 25,
  nbsSpoofDetected = 30,
  nbsBadObject = 40,
  nbsTooSoft = 51,
  nbsTooHard = 52,
  nbsBadPosition = 60,
  nbsTooNorth = 61,
  nbsTooEast = 62,
  nbsTooSouth = 63,
  nbsTooWest = 64,
  nbsTooClose = 65,
  nbsTooFar = 66,
  nbsBadSpeed = 70,
  nbsTooSlow = 71,
  nbsTooFast = 72,
  nbsBadSize = 80,
  nbsTooShort = 81,
  nbsTooLong = 82,
  nbsTooNarrow = 83,
  nbsTooWide = 84,
  nbsTooSkewed = 91,
  nbsWrongDirection = 92
} NBiometricStatus;
```

Members

Members	Description
nbsNone = 0	Indicates that device is performing a concrete operation and can not return status value. E.g., fingerprint is swiped on a fingerprint scanner (operation is not finished yet and NBiometricDevice [1142] can not return any other value than nbsNone).
nbsOk = 1	Indicates succeeded operation.
nbsCanceled = 2	Indicates canceled operation.
nbsTimeout = 3	Indicates that operation timed-out.
nbsCleaningNeeded = 10	Indicates that biometric device should be cleaned.
nbsObjectsNotRemoved = 20	Indicates that object(s) was not removed after NBiometricDevice [1142] finished operation.
nbsObjectMissing = 21	Indicates that object is missing.
nbsObjectNotFound = 22	Indicates that object(s) was not found.
nbsTooFewObjects = 23	Indicates that too few objects was provided for device to succeed its operation. E.g., not enough fingerprints were scanned.
nbsTooManyObjects = 24	Indicates that too many objects were provided for NBiometricDevice [1142] to perform operation.
nbsBadObjectSequence = 25	Indicates sequence of bad object(s).
nbsSpoofDetected = 30	Indicates that spoof object was detected.
nbsBadObject = 40	Indicates bad object.
nbsTooSoft = 51	Indicates that fingerprint was placed too soft on a device.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

nbsTooHard = 52	Indicates that fingerprint was placed too hard on a device.
nbsBadPosition = 60	Indicates bad fingerprint position.
nbsTooNorth = 61	Indicated that object was too north from device.
nbsTooEast = 62	Indicated that object was too east from device.
nbsTooSouth = 63	Indicated that object was too south from device.
nbsTooWest = 64	Indicated that object was too west from device.
nbsTooClose = 65	Indicated that object was too close to device.
nbsTooFar = 66	Indicated that object was too far from device.
nbsBadSpeed = 70	Indicates too slow/fast moving object during operation. E.g., fingerprint was swiped too slow (too fast) when fingerprint scanning operation was performed on a scanner.
nbsTooSlow = 71	Indicates that object was moving too slow. E.g., fingerprint was swiped through scanner too slow.
nbsTooFast = 72	Indicates that object was moving too fast. E.g., fingerprint was swiped through scanner too fast.
nbsBadSize = 80	Indicates object's bad size.
nbsTooShort = 81	Indicates that object is too short for NBiometricDevice [1142] to perform operation.
nbsTooLong = 82	Indicates that object is too long for NBiometricDevice [1142] to perform operation.
nbsTooNarrow = 83	Indicates that object is too narrow for NBiometricDevice [1142] to perform operation.
nbsTooWide = 84	Indicates that object is too wide for NBiometricDevice [1142] to perform operation.
nbsTooSkewed = 91	Indicates that object is too skewed.
nbsWrongDirection = 92	Indicates that object was moving to wrong direction.

Module

NBiometricTypes Unit [61]

7.1.1.1.2.3 NBiometricSubtype Enumeration

Enumerates values which define the type of biometric data in the record. The values in this enumeration more specifically define the type of biometric data (NBiometricType [89]) in the record.

C++

```
typedef enum NBiometricSubtype_ {
  nbstNone = 0x000000,
  nbstLeft = 0x000001,
  nbstRight = 0x000002,
  nbstLeftThumb = 0x000004,
  nbstLeftPointerFinger = 0x000008,
  nbstLeftMiddleFinger = 0x000010,
  nbstLeftRingFinger = 0x000020,
  nbstLeftLittleFinger = 0x000040,
  nbstRightThumb = 0x000080,
  nbstRightPointerFinger = 0x000100,
  nbstRightMiddleFinger = 0x000200,
  nbstRightRingFinger = 0x000400,
  nbstRightLittleFinger = 0x000800,
  nbstLeftPalm = 0x001000,
  nbstLeftBackOfHand = 0x002000,
  nbstLeftWrist = 0x004000,
  nbstRightPalm = 0x008000,
  nbstRightBackOfHand = 0x010000,
  nbstRightWrist = 0x020000
} NBiometricSubtype;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Members

Members	Description
nbstNone = 0x000000	Indicates that none of the NBiometricSubtype values were set for the record.
nbstLeft = 0x000001	Indicates that record contains left biometric object. E.g., if `nbtIris` value from `NBiometricType` [89] enumeration and `nbstLeft` values from `NBiometricSubtype` enumeration were set, this means that record contains left iris data.
nbstRight = 0x000002	Indicates that record contains right biometric object
nbstLeftThumb = 0x000004	Indicates that record contains left thumb data.
nbstLeftPointerFinger = 0x000008	Indicates that record contains left pointer finger data.
nbstLeftMiddleFinger = 0x000010	Identify the instance of biometric type related to left middle finger.
nbstLeftRingFinger = 0x000020	Indicates that record contains left ring finger data.
nbstLeftLittleFinger = 0x000040	Identify the instance of biometric type related to left little finger.
nbstRightThumb = 0x000080	Indicates that record contains right thumb data.
nbstRightPointerFinger = 0x000100	Indicates that record contains right pointer finger data.
nbstRightMiddleFinger = 0x000200	Identify the instance of biometric type related to right middle finger.
nbstRightRingFinger = 0x000400	Indicates that record contains right ring finger data.
nbstRightLittleFinger = 0x000800	Identify the instance of biometric type related to right little finger.
nbstLeftPalm = 0x001000	Indicates that record contains left palm data.
nbstLeftBackOfHand = 0x002000	Indicates that record contains left back of the palm data.
nbstLeftWrist = 0x004000	Indicates that record contains left wrist data.
nbstRightPalm = 0x008000	Indicates that record contains right palm data.
nbstRightBackOfHand = 0x010000	Indicates that record contains right back of the palm data.
nbstRightWrist = 0x020000	Indicates that record contains right wrist data.

Module

NBiometricTypes Unit [61]

7.1.1.1.2.4 NBiometricType Enumeration

Enumerates types of biometric technology used in a record.

C++

```
typedef enum NBiometricType_ {
  nbtNone = 0x000000,
  nbtMultipleBiometrics = 0x000001,
  nbtFace = 0x000002,
  nbtFacialFeatures = nbtFace,
  nbtVoice = 0x000004,
  nbtFinger = 0x000008,
  nbtFingerprint = nbtFinger,
  nbtIris = 0x000010,
  nbtRetina = 0x000020,
  nbtHandGeometry = 0x000040,
  nbtSignatureOrSign = 0x000080,
  nbtSignatureDynamics = nbtSignatureOrSign,
  nbtKeystroke = 0x000100,
  nbtKeystrokeDynamics = nbtKeystroke,
  nbtLipMovement = 0x000200,
  nbtThermalFace = 0x000400,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
    nbtThermalHand = 0x000800,
    nbtThermalFaceImage = nbtThermalFace,
    nbtThermalHandImage = nbtThermalHand,
    nbtGait = 0x001000,
    nbtScent = 0x002000,
    nbtBodyOdor = nbtScent,
    nbtDna = 0x004000,
    nbtEar = 0x008000,
    nbtEarShape = nbtEar,
    nbtFingerGeometry = 0x010000,
    nbtPalm = 0x020000,
    nbtPalmPrint = nbtPalm,
    nbtVein = 0x040000,
    nbtVeinPattern = nbtVein,
    nbtFoot = 0x080000,
    nbtFootPrint = nbtFoot,
    nbtPalmGeometry = 0x100000
  } NBiometricType;
```

Members

Members	Description
nbtNone = 0x000000	Record does not contain data of any biometric type.
nbtMultipleBiometrics = 0x000001	Indicates that record contain data of multiple biometric types.
nbtFace = 0x000002	Indicates face data.
nbtFacialFeatures = nbtFace	Indicates that record contain facial features data.
nbtVoice = 0x000004	Indicates voice data.
nbtFinger = 0x000008	Indicates finger data.
nbtFingerprint = nbtFinger	Indicates fingerprint data.
nbtIris = 0x000010	Indicates iris data.
nbtRetina = 0x000020	Indicates retina data.
nbtHandGeometry = 0x000040	Indicates hand geometry.
nbtSignatureOrSign = 0x000080	Indicates signature or sign.
nbtSignatureDynamics = nbtSignatureOrSign	Indicates signature dynamics.
nbtKeystroke = 0x000100	Indicates keystroke.
nbtKeystrokeDynamics = nbtKeystroke	Indicates keystroke dynamics.
nbtLipMovement = 0x000200	Indicates lip movement.
nbtThermalFace = 0x000400	Indicates thermal face.
nbtThermalHand = 0x000800	Indicates thermal hand.
nbtThermalFaceImage = nbtThermalFace	Indicates thermal face image.
nbtThermalHandImage = nbtThermalHand	Indicates thermal hand image.
nbtGait = 0x001000	Indicates gait.
nbtScent = 0x002000	Indicates scent.
nbtBodyOdor = nbtScent	Indicates body odor.
nbtDna = 0x004000	Indicates DNA.
nbtEar = 0x008000	Indicates ear.
nbtEarShape = nbtEar	Indicates that record contains ear shape object.
nbtFingerGeometry = 0x010000	Indicates finger geometry.
nbtPalm = 0x020000	Indicates palm data.
nbtPalmPrint = nbtPalm	Indicates palmprint data.
nbtVein = 0x040000	Indicates vein data.
nbtVeinPattern = nbtVein	Indicates vein pattern.
nbtFoot = 0x080000	Indicates foot data.
nbtFootPrint = nbtFoot	Indicates footprint data.
nbtPalmGeometry = 0x100000	Indicates palm geometry.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NBiometricTypes Unit [61]

7.1.1.1.2.5 NEPosition Enumeration

Enumerates eye position values.

C++

```
typedef enum NEPosition_ {
  nepUnknown = 0,
  nepRight = 1,
  nepLeft = 2,
  nepBoth = 3
} NEPosition;
```

Members

Members	Description
nepUnknown = 0	Unknown eye.
nepRight = 1	Right eye.
nepLeft = 2	Left eye.
nepBoth = 3	Both eyes.

Module

NBiometricTypes Unit [61]

7.1.1.1.2.6 NFImpressionType Enumeration

Specifies the impression type.

C++

```
typedef enum NFImpressionType_ {
  nfitLiveScanPlain = 0,
  nfitLiveScanRolled = 1,
  nfitNonliveScanPlain = 2,
  nfitNonliveScanRolled = 3,
  nfitLatentImpression = 4,
  nfitLatentTracing = 5,
  nfitLatentPhoto = 6,
  nfitLatentLift = 7,
  nfitLiveScanVerticalSwipe = 8,
  nfitSwipe = nfitLiveScanVerticalSwipe,
  nfitLiveScanContactless = 9,
  nfitLiveScanPalm = 10,
  nfitNonliveScanPalm = 11,
  nfitLatentPalmImpression = 12,
  nfitLatentPalmTracing = 13,
  nfitLatentPalmPhoto = 14,
  nfitLatentPalmLift = 15,
  nfitLiveScanOpticalContactPlain = 20,
  nfitLiveScanOpticalContactRolled = 21,
  nfitLiveScanNonOpticalContactPlain = 22,
  nfitLiveScanNonOpticalContactRolled = 23,
  nfitLiveScanOpticalContactlessPlain = 24,
  nfitLiveScanOpticalContactlessRolled = 25,
  nfitLiveScanNonOpticalContactlessPlain = 26,
  nfitLiveScanNonOpticalContactlessRolled = 27,
  nfitOther = 28,
  nfitUnknown = 29
} NFImpressionType;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Members

Members	Description
nfitLiveScanPlain = 0	Live-scanned plain fingerprint.
nfitLiveScanRolled = 1	Live-scanned rolled fingerprint.
nfitNonliveScanPlain = 2	Non-live-scanned (from paper) plain fingerprint.
nfitNonliveScanRolled = 3	Non-live-scanned (from paper) rolled fingerprint.
nfitLatentImpression = 4	Latent impression fingerprint.
nfitLatentTracing = 5	Latent tracing fingerprint.
nfitLatentPhoto = 6	Latent photo fingerprint.
nfitLatentLift = 7	Latent lift fingerprint.
nfitLiveScanVerticalSwipe = 8	Fingerprint live-scanned by a vertical swipe across the scanner.
nfitSwipe = nfitLiveScanVerticalSwipe	Live-scanned fingerprint by sliding the finger across a "swipe" sensor.
nfitLiveScanContactless = 9	Live-scanned fingerprint using contactless device.
nfitLiveScanPalm = 10	Live scanned palm.
nfitNonliveScanPalm = 11	Non-live scanned palm.
nfitLatentPalmImpression = 12	Latent palm impression.
nfitLatentPalmTracing = 13	Latent palm tracing.
nfitLatentPalmPhoto = 14	Latent palm photo.
nfitLatentPalmLift = 15	Latent palm lift.
nfitLiveScanOpticalContactPlain = 20	Live-scanned plain fingerprint using an optical contact device.
nfitLiveScanOpticalContactRolled = 21	Live-scanned rolled fingerprint using an optical contact device.
nfitLiveScanNonOpticalContactPlain = 22	Live-scanned plain fingerprint using a non-optical contact device.
nfitLiveScanNonOpticalContactRolled = 23	Live-scanned rolled fingerprint using a non-optical contact device.
nfitLiveScanOpticalContactlessPlain = 24	Live-scanned plain fingerprint using an optical contactless device.
nfitLiveScanOpticalContactlessRolled = 25	Live-scanned rolled fingerprint using an optical contactless device.
nfitLiveScanNonOpticalContactlessPlain = 26	Live-scanned plain fingerprint using a non-optical contactless device.
nfitLiveScanNonOpticalContactlessRolled = 27	Live-scanned rolled fingerprint using a non-optical contactless device.
nfitOther = 28	Other impression type.
nfitUnknown = 29	Unknown impression type.

Module

NBiometricTypes Unit [61]

7.1.1.1.2.7 NFPosition Enumeration

Specifies the finger position.

C++

```
typedef enum NFPosition_ {
  nfpUnknown = 0,
  nfpRightThumb = 1,
  nfpRightIndexFinger = 2,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
    nfpRightIndex = nfpRightIndexFinger,
    nfpRightMiddleFinger = 3,
    nfpRightMiddle = nfpRightMiddleFinger,
    nfpRightRingFinger = 4,
    nfpRightRing = nfpRightRingFinger,
    nfpRightLittleFinger = 5,
    nfpRightLittle = nfpRightLittleFinger,
    nfpLeftThumb = 6,
    nfpLeftIndexFinger = 7,
    nfpLeftIndex = nfpLeftIndexFinger,
    nfpLeftMiddleFinger = 8,
    nfpLeftMiddle = nfpLeftMiddleFinger,
    nfpLeftRingFinger = 9,
    nfpLeftRing = nfpLeftRingFinger,
    nfpLeftLittleFinger = 10,
    nfpLeftLittle = nfpLeftLittleFinger,
    nfpPlainRightThumb = 11,
    nfpPlainLeftThumb = 12,
    nfpPlainRightFourFingers = 13,
    nfpPlainLeftFourFingers = 14,
    nfpPlainThumbs = 15,
    nfpUnknownPalm = 20,
    nfpRightFullPalm = 21,
    nfpRightWritersPalm = 22,
    nfpLeftFullPalm = 23,
    nfpLeftWritersPalm = 24,
    nfpRightLowerPalm = 25,
    nfpRightUpperPalm = 26,
    nfpLeftLowerPalm = 27,
    nfpLeftUpperPalm = 28,
    nfpRightOther = 29,
    nfpLeftOther = 30,
    nfpRightInterdigital = 31,
    nfpRightThenar = 32,
    nfpRightHypothenar = 33,
    nfpLeftInterdigital = 34,
    nfpLeftThenar = 35,
    nfpLeftHypothenar = 36,
    nfpRightIndexMiddleFingers = 40,
    nfpRightMiddleRingFingers = 41,
    nfpRightRingLittleFingers = 42,
    nfpLeftIndexMiddleFingers = 43,
    nfpLeftMiddleRingFingers = 44,
    nfpLeftRingLittleFingers = 45,
    nfpRightIndexLeftIndexFingers = 46,
    nfpRightIndexMiddleRingFingers = 47,
    nfpRightMiddleRingLittleFingers = 48,
    nfpLeftIndexMiddleRingFingers = 49,
    nfpLeftMiddleRingLittleFingers = 50,
    nfpUnknownTwoFingers = -2,
    nfpUnknownThreeFingers = -3,
    nfpUnknownFourFingers = -4
  } NFPosition;
```

Members

Members	Description
nfpUnknown = 0	Unknown finger.
nfpRightThumb = 1	Thumb of the right hand.
nfpRightIndexFinger = 2	Index finger of the right hand.
nfpRightIndex = nfpRightIndexFinger	Index finger of the right hand.
nfpRightMiddleFinger = 3	Middle finger of the right hand.
nfpRightMiddle = nfpRightMiddleFinger	Middle finger of the right hand.
nfpRightRingFinger = 4	Ring finger of the right hand.
nfpRightRing = nfpRightRingFinger	Ring finger of the right hand.
nfpRightLittleFinger = 5	Little finger of the right hand.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

nfpRightLittle = nfpRightLittleFinger	Little finger of the right hand.
nfpLeftThumb = 6	Thumb of the left hand.
nfpLeftIndexFinger = 7	Index finger of the left hand.
nfpLeftIndex = nfpLeftIndexFinger	Index finger of the left hand.
nfpLeftMiddleFinger = 8	Middle finger of the left hand.
nfpLeftMiddle = nfpLeftMiddleFinger	Middle finger of the left hand.
nfpLeftRingFinger = 9	Little finger of the left hand.
nfpLeftRing = nfpLeftRingFinger	Ring finger of the left hand.
nfpLeftLittleFinger = 10	Little finger of the left hand.
nfpLeftLittle = nfpLeftLittleFinger	Little finger of the left hand.
nfpPlainRightThumb = 11	Specifies plain right thumb.
nfpPlainLeftThumb = 12	Specifies leftplain thumb.
nfpPlainRightFourFingers = 13	Specifies plain right four fingers.
nfpPlainLeftFourFingers = 14	Specifies left four fingers.
nfpPlainThumbs = 15	Specifies plain thumbs.
nfpUnknownPalm = 20	Specifies unknown palm.
nfpRightFullPalm = 21	Specifies right full palm.
nfpRightWritersPalm = 22	Specifies right writers palm.
nfpLeftFullPalm = 23	Specifies full left palm.
nfpLeftWritersPalm = 24	Specifies left writers palm.
nfpRightLowerPalm = 25	Specifies right lower palm.
nfpRightUpperPalm = 26	Specifies right upper palm.
nfpLeftLowerPalm = 27	Specifies lower palm.
nfpLeftUpperPalm = 28	Specifies left upper palm.
nfpRightOther = 29	Unknown finger of the right arm .
nfpLeftOther = 30	Unknown finger of the left arm .
nfpRightInterdigital = 31	Specifies right interdigital.
nfpRightThenar = 32	Specifies right thenar.
nfpRightHypothenar = 33	Specifies right hypothenar.
nfpLeftInterdigital = 34	Specifies left interdigital.
nfpLeftThenar = 35	Specifies left thenar.
nfpLeftHypothenar = 36	Specifies left hypothenar.
nfpRightIndexMiddleFingers = 40	Index and middle fingers of the right hand.
nfpRightMiddleRingFingers = 41	Middle and ring fingers of the right hand.
nfpRightRingLittleFingers = 42	Ring and little fingers of the right hand.
nfpLeftIndexMiddleFingers = 43	Index and middle fingers of the left hand.
nfpLeftMiddleRingFingers = 44	Middle and ring fingers of the left hand.
nfpLeftRingLittleFingers = 45	Ring and little fingers of the left hand.
nfpRightIndexLeftIndexFingers = 46	Index fingers of the right and left hands.
nfpRightIndexMiddleRingFingers = 47	Index, middle and ring fingers of the right hand.
nfpRightMiddleRingLittleFingers = 48	Middle, ring and little fingers of the right hand.
nfpLeftIndexMiddleRingFingers = 49	Index, middle and ring fingers of the left hand.
nfpLeftMiddleRingLittleFingers = 50	Middle, ring and little fingers of the left hand.
nfpUnknownTwoFingers = -2	Unknown two fingers.
nfpUnknownThreeFingers = -3	Unknown three fingers.
nfpUnknownFourFingers = -4	Unknown four fingers.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NBiometricTypes Unit [61]

7.1.1.1.2.8 NGender Enumeration

Specifies the gender.

C++

```
typedef enum NGender_ {
  ngUnspecified = 0,
  ngMale = 1,
  ngFemale = 2,
  ngUnknown = 255
} NGender;
```

Members

Members	Description
ngUnspecified = 0	Unspecified gender.
ngMale = 1	Male.
ngFemale = 2	Female.
ngUnknown = 255	Unknown gender.

Module

NBiometricTypes Unit [61]

7.1.1.1.2.9 NLExpression Enumeration

Specifies the facial expression.

C++

```
typedef enum NLExpression_ {
  nleUnspecified = 0,
  nleNeutral = 1,
  nleSmile = 2,
  nleSmileOpenedJaw = 3,
  nleRaisedBrows = 4,
  nleEyesAway = 5,
  nleSquinting = 6,
  nleFrowning = 7,
  nleUnknown = 0xFFFF
} NLExpression;
```

Members

Members	Description
nleUnspecified = 0	Unspecified expression.
nleNeutral = 1	Neutral expression.
nleSmile = 2	Smiling.
nleSmileOpenedJaw = 3	Smiling with jaw open.
nleRaisedBrows = 4	Eyebrows raised.
nleEyesAway = 5	Eyes looking away.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

nleSquinting = 6	Squinting.
nleFrowning = 7	Frowning.
nleUnknown = 0xFFFF	Unknown expression.

Module

NBiometricTypes Unit [61]

7.1.1.1.2.10 NLProperties Enumeration

Specified the facial properties.

C++

```
typedef enum NLProperties_ {
  nlpNotSpecified = 0,
  nlpSpecified = 0x000001,
  nlpGlasses = 0x000002,
  nlpMoustache = 0x000004,
  nlpBeard = 0x000008,
  nlpTeethVisible = 0x000010,
  nlpBlink = 0x000020,
  nlpMouthOpen = 0x000040,
  nlpLeftEyePatch = 0x000080,
  nlpRightEyePatch = 0x000100,
  nlpBothEyePatch = 0x000200,
  nlpDarkGlasses = 0x000400,
  nlpDistortingCondition = 0x000800,
  nlpHat = 0x01000000,
  nlpScarf = 0x02000000,
  nlpNoEar = 0x04000000
} NLProperties;
```

Members

Members	Description
nlpNotSpecified = 0	Face properties not specified.
nlpSpecified = 0x000001	Face properties specified.
nlpGlasses = 0x000002	Wearing glasses.
nlpMoustache = 0x000004	Moustache.
nlpBeard = 0x000008	Beard.
nlpTeethVisible = 0x000010	Teeth are visible.
nlpBlink = 0x000020	Eye blink.
nlpMouthOpen = 0x000040	Mouth is open.
nlpLeftEyePatch = 0x000080	Patch on left eye.
nlpRightEyePatch = 0x000100	Patch on right eye.
nlpBothEyePatch = 0x000200	Patch on both eyes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

nlpDarkGlasses = 0x000400	Wearing dark glasses.
nlpDistortingCondition = 0x000800	Distorting conditions.
nlpHat = 0x01000000	Wearing a hat.
nlpScarf = 0x02000000	Wearing a scarf.
nlpNoEar = 0x04000000	One ear missing.

Module

NBiometricTypes Unit [61]

7.1.1.1.3 Types

7.1.1.1.3.1 NLFeaturePoint Type

Represents a facial feature point in a NLRecord [134].

C++

```
typedef struct NLFeaturePoint_ NLFeaturePoint;
```

Module

NBiometricTypes Unit [61]

7.1.1.1.4 Macros

7.1.1.1.4.1 N_BIOMETRIC_QUALITY_FAILED Macro

Failed quality.

C++

```
#define N_BIOMETRIC_QUALITY_FAILED 255
```

Module

NBiometricTypes Unit [61]

7.1.1.1.4.2 N_BIOMETRIC_QUALITY_MAX Macro

Maximal quality.

C++

```
#define N_BIOMETRIC_QUALITY_MAX 100
```

Module

NBiometricTypes Unit [61]

7.1.1.1.4.3 N_BIOMETRIC_QUALITY_MIN Macro

Minimal quality.

C++

```
#define N_BIOMETRIC_QUALITY_MIN 0
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NBiometricTypes Unit [61]

7.1.1.1.4.4 N_BIOMETRIC_QUALITY_UNKNOWN Macro

Unknown quality.

C++

```
#define N_BIOMETRIC_QUALITY_UNKNOWN 254
```

Module

NBiometricTypes Unit [61]

7.1.1.1.4.5 N_PHRASE_ID_UNKNOWN Macro

Unknown phrase ID.

C++

```
#define N_PHRASE_ID_UNKNOWN 0
```

Module

NBiometricTypes Unit [61]

7.1.1.1.4.6 NBiometricAngleToString Macro

Picks a Unicode (W) or ANSCII (A) version of the NBiometricAngleToString function.

C++

```
#define NBiometricAngleToString N_FUNC_AW(NBiometricAngleToString)
```

Module

NBiometricTypes Unit [61]

7.1.1.1.4.7 NBiometricQualityToString Macro

Picks a Unicode (W) or ANSCII (A) version of the NBiometricQualityToString function.

C++

```
#define NBiometricQualityToString N_FUNC_AW(NBiometricQualityToString)
```

Module

NBiometricTypes Unit [61]

7.1.1.1.4.8 NFCoreToString Macro

Picks a Unicode (W) or ANSCII (A) version of the NFCoreToString function.

C++

```
#define NFCoreToString N_FUNC_AW(NFCoreToString)
```

Module

NBiometricTypes Unit [61]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.4.9 NFDeltaToString Macro

Picks a Unicode (W) or ANSCII (A) version of the NFDeltaToString function.

C++

```
#define NFDeltaToString N_FUNC_AW(NFDeltaToString)
```

Module

NBiometricTypes Unit [61]

7.1.1.1.4.10 NFDoubleCoreToString Macro

Picks a Unicode (W) or ANSCII (A) version of the NFDoubleCoreToString function.

C++

```
#define NFDoubleCoreToString N_FUNC_AW(NFDoubleCoreToString)
```

Module

NBiometricTypes Unit [61]

7.1.1.1.4.11 NFMinutiaNeighborToString Macro

Picks a Unicode (W) or ANSCII (A) version of the NFMinutiaNeighborToString function.

C++

```
#define NFMinutiaNeighborToString N_FUNC_AW(NFMinutiaNeighborToString)
```

Module

NBiometricTypes Unit [61]

7.1.1.1.4.12 NFMinutiaToString Macro

Picks a Unicode (W) or ANSCII (A) version of the NFMinutiaToString function.

C++

```
#define NFMinutiaToString N_FUNC_AW(NFMinutiaToString)
```

Module

NBiometricTypes Unit [61]

7.1.1.1.4.13 NLFeaturePointToString Macro

Picks a Unicode (W) or ANSCII (A) version of the NLFeaturePointToString function.

C++

```
#define NLFeaturePointToString N_FUNC_AW(NLFeaturePointToString)
```

Module

NBiometricTypes Unit [61]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.1.5 Files

7.1.1.1.5.1 NBiometricTypes.h

Header file for NBiometricType [89] module.

Enumerations

	Name	Description
	NBiometricStatus_ [86]	Enumerates biometric status values. These values are returned by NBiometricDevice [1142].
	NBiometricSubtype_ [88]	Enumerates values which define the type of biometric data in the record. The values in this enumeration more specifically define the type of biometric data (NBiometricType [89]) in the record.
	NBiometricType_ [89]	Enumerates types of biometric technology used in a record.
	NEPosition_ [91]	Enumerates eye position values.
	NFImpressionType_ [91]	Specifies the impression type.
	NFPosition_ [92]	Specifies the finger position.
	NGender_ [95]	Specifies the gender.
	NLExpression_ [95]	Specifies the facial expression.
	NLProperties_ [96]	Specified the facial properties.
	NBiometricStatus [86]	Enumerates biometric status values. These values are returned by NBiometricDevice [1142].
	NBiometricSubtype [88]	Enumerates values which define the type of biometric data in the record. The values in this enumeration more specifically define the type of biometric data (NBiometricType [89]) in the record.
	NBiometricType [89]	Enumerates types of biometric technology used in a record.
	NEPosition [91]	Enumerates eye position values.
	NFImpressionType [91]	Specifies the impression type.
	NFPosition [92]	Specifies the finger position.
	NGender [95]	Specifies the gender.
	NLExpression [95]	Specifies the facial expression.
	NLProperties [96]	Specified the facial properties.

Functions

	Name	Description
	NBiometricAngleFromRadians [65]	Converts the specified angle from radians to degrees.
	NBiometricAngleToRadians [65]	Converts the specified angle from degrees to radians.
	NBiometricAngleToString [65]	Gets a string representation of the specified angle.
	NBiometricQualityIsValid [66]	Checks whether the specified quality is valid.
	NBiometricQualityToString [66]	Gets a string representation of the specified quality.
	NBiometricStatusIsFinal [66]	Checks if the value passed as a parameter is denotes completed biometric operation rather then intermediate status of operation in progress.
	NBiometricStatusIsValid [67]	Checks if the value passed as a parameter is defined in NBiometricStatus [86].
	NBiometricSubtypeIsValid [67]	Checks if the value passed as a parameter is a valid NBiometricSubtype [88] value.
	NBiometricTypeIsValid [67]	Checks if the value passed as a parameter is a valid NBiometricType [89] value.
	NEPositionGetAvailableParts [67]	Determines available iris positions based on specified missing iris positions and base position.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NEPositionIsKnown [68]	Checks whether the specified position is known.
	NEPositionIsLeft [68]	Checks whether the specified position is left.
	NEPositionIsOneOf [69]	Checks if specified NEPosition [91] value matches one in an array.
	NEPositionIsRight [69]	Checks whether the specified position is right.
	NEPositionIsRightAndLeft [69]	Checks whether the specified position is both left and right.
	NEPositionIsSingleEye [70]	Checks whether the specified position represents a single eye.
	NEPositionIsTheEye [70]	Checks whether the specified position represents a separate eye.
	NEPositionIsTwoEyes [70]	Checks whether the specified position represents two eyes.
	NEPositionIsValid [70]	Checks if the value passed as a parameter is a valid NEPosition [91] value.
	NEPositionIsValidTheEye [71]	Checks if the specified value is a valid NEPosition [91] value.
	NFCoreToString [71]	Gets a string representation of the core.
	NFDeltaToString [71]	Gets a string representation of the delta.
	NFDoubleCoreToString [72]	Gets a string representation of the double core.
	NFImpressionTypeIsCompatibleWith [72]	Checks if two specified values are compatible.
	NFImpressionTypeIsContact [73]	Checks whether the specified NFImpressionType [91] value is one of the following: nfitLiveScanOpticalContactPlain, nfitLiveScanOpticalContactRolled, nfitLiveScanNonOpticalContactPlain, nfitLiveScanNonOpticalContactRolled
	NFImpressionTypeIsContactless [73]	Checks if the specified NFImpressionType [91] value is one of the following: nfitLiveScanContactless, nfitLiveScanOpticalContactlessPlain, nfitLiveScanOpticalContactlessRolled, nfitLiveScanNonOpticalContactlessPlain, nfitLiveScanNonOpticalContactlessRolled.
	NFImpressionTypeIsFinger [73]	Checks whether the specified value is fingerprint.
	NFImpressionTypeIsGeneric [74]	Checks whether the specified impression type is generic.
	NFImpressionTypeIsLatent [74]	Checks if the specified value is latent fingerprint or palmprint.
	NFImpressionTypeIsLiveScan [74]	Checks if the specified value is a live scan.
	NFImpressionTypeIsNonliveScan [74]	Checks if the specified value is a non-live scan.
	NFImpressionTypeIsNonOptical [75]	Checks if the specified NFImpressionType [91] value is one of the following: nfitLiveScanNonOpticalContactPlain, nfitLiveScanNonOpticalContactRolled, nfitLiveScanNonOpticalContactlessPlain, nfitLiveScanNonOpticalContactlessRolled.
	NFImpressionTypeIsOneOf [75]	Checks if specified NFImpressionType [91] value matches one in an array.
	NFImpressionTypeIsOptical [76]	Checks if the specified NFImpressionType [91] value is one of the following: nfitLiveScanOpticalContactPlain, nfitLiveScanOpticalContactRolled, nfitLiveScanOpticalContactlessPlain, nfitLiveScanOpticalContactlessRolled.
	NFImpressionTypeIsPalm [76]	Checks if the specified value is a palmprint.
	NFImpressionTypeIsPlain [76]	Checks if the specified value is a plain scan or optical contact/contactless.
	NFImpressionTypeIsRolled [76]	Checks if the specified value is one of rolled biometric types.
	NFImpressionTypeIsSwipe [77]	Checks if the specified NFImpressionType [91] value is one of the following: nfitLiveScanVerticalSwipe.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NFImpressionTypeIsValid [77]	Checks if the value passed as a parameter is a valid NFImpressionType [91] value.
	NFImpressionTypeIsValidFinger [77]	Checks if the specified value is a valid fingerprint.
	NFImpressionTypeIsValidPalm [78]	Checks if the specified value is a valid palmprint.
	NFImpressionTypeToFinger [78]	Converts an impression type object into finger impression type object.
	NFMinutiaNeighborToString [78]	Gets a string representation of the minutia neighbor.
	NFMinutiaToString [79]	Gets a string representation of the minutia.
	NFPositionGetAvailableParts [79]	Determines available finger positions based on specified missing finger positions and base position.
	NFPositionIsCompatibleWith [80]	Checks if two specified values are compatible.
	NFPositionIsCompatibleWithImpressionType [80]	Checks whether the specified position is compatible with the specified impression type.
	NFPositionIsFinger [80]	Checks if the specified value is fingerprint.
	NFPositionIsFourFingers [81]	Checks if the specified value is a type of 4 fingerprints data.
	NFPositionIsKnown [81]	Checks if the record contains biometric data which type is one of the known fingerprint types (thumb, palmprint, etc.).
	NFPositionIsLeft [81]	Checks if the specified position is left fingerprint.
	NFPositionIsLeftAndRight [81]	Checks if the specified position is left and right fingerprint.
	NFPositionIsOneOf [82]	Checks if specified NFPosition [92] value matches one in an array.
	NFPositionIsPalm [82]	Checks if the specified value is a palmprint.
	NFPositionIsRight [83]	Checks if the specified position is right fingerprint.
	NFPositionIsSingleFinger [83]	Checks if the specified value is a single fingerprint.
	NFPositionIsTheFinger [83]	Checks if the specified value is the fingerprint.
	NFPositionIsThePalm [83]	Checks whether the specified position represents a separate palm.
	NFPositionIsThreeFingers [84]	Checks if the specified value is a type of 3 fingerprints data.
	NFPositionIsTwoFingers [84]	Checks if the specified value is a type of 2 fingerprints data.
	NFPositionIsValid [84]	Checks if the value passed as a parameter is a valid NFPosition [92] value.
	NFPositionIsValidFinger [85]	Checks if the specified value is a valid fingerprint.
	NFPositionIsValidPalm [85]	Checks if the specified value is a valid palmprint.
	NFPositionIsValidTheFinger [85]	Checks if the specified value is a valid fingerprint.
	NLFeaturePointToString [86]	Gets a string representation of the feature point.

Macros

Name	Description
N_BIOMETRIC_QUALITY_FAILED [97]	Failed quality.
N_BIOMETRIC_QUALITY_MAX [97]	Maximal quality.
N_BIOMETRIC_QUALITY_MIN [97]	Minimal quality.
N_BIOMETRIC_QUALITY_UNKNOWN [98]	Unknown quality.
N_PHRASE_ID_UNKNOWN [98]	Unknown phrase ID.
NBiometricAngleToString [98]	Picks a Unicode (W) or ANSCII (A) version of the NBiometricAngleToString function.
NBiometricQualityToString [98]	Picks a Unicode (W) or ANSCII (A) version of the NBiometricQualityToString function.
NFCoreToString [98]	Picks a Unicode (W) or ANSCII (A) version of the NFCoreToString function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFDeltaToString [99]	Picks a Unicode (W) or ANSCII (A) version of the NFDeltaToString function.
NFDoubleCoreToString [99]	Picks a Unicode (W) or ANSCII (A) version of the NFDoubleCoreToString function.
NFMinutiaNeighborToString [99]	Picks a Unicode (W) or ANSCII (A) version of the NFMinutiaNeighborToString function.
NFMinutiaToString [99]	Picks a Unicode (W) or ANSCII (A) version of the NFMinutiaToString function.
NLFeaturePointToString [99]	Picks a Unicode (W) or ANSCII (A) version of the NLFeaturePointToString function.

Module

NBiometricTypes Unit [61]

Structures

	Name	Description
	NLFeaturePoint_ [86]	Enumerates detected facial feature point values.

Types

Name	Description
NLFeaturePoint [97]	Represents a facial feature point in a NLRecord [134].

7.1.1.2 NExtractors Unit

Provides functions for retrieving NExtractors Library information.

Files

Name	Description
NExtractors.h [103]	Header file for the NExtractor. Provides functions for retrieving information about NExtractors library.

7.1.1.2.1 Files

7.1.1.2.1.1 NExtractors.h

Header file for the NExtractor. Provides functions for retrieving information about NExtractors library.

Module

NExtractors Unit [103]

7.1.1.3 NLAttributes Unit

Provides functions to retrieve face attributes (metadata .which is not saved in face template).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Files

Name	Description
NLAttributes.h [104]	Header file for the NLAttributes Unit.

Functions

	Name	Description
	NLAttributesCreate [104]	Creates NLAttributes.

7.1.1.3.1 Functions

7.1.1.3.1.1 NLAttributesCreate Function

Creates NLAttributes [103].

C++

```
NResult N_API NLAttributesCreate(HNLAttributes * phAttributes);
```

Parameters

Parameters	Description
HNLAttributes * phAttributes	[out] Handle to NLAttributes [103] which contains face attributes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NLAttributes Unit [103]

7.1.1.3.2 Files

7.1.1.3.2.1 NLAttributes.h

Header file for the NLAttributes [103] Unit.

Functions

	Name	Description
	NLAttributesCreate [104]	Creates NLAttributes [103].

Module

NLAttributes Unit [103]

7.1.1.4 NLExtractor Unit

NLExtractor is a part of NBiometrics library intended for human face finding in pictures and extracting their features into Neurotechnology Faces [45] Template (NLTemplate [153]).

NLExtractor can detect human gender, expression, glasses, dark glasses, blink eyes and mouth open.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Files

Name	Description
NLExtractor.h [125]	Header file for the NLExtractor Unit. NLExtractor is a part of NBiometrics library intended for human face finding in pictures and extracting their features into Neurotechnology Faces [45] Template(NLTemplate [153]).
NLExtractorParams.h [127]	C header file which contains NLExtractor parameters.

Functions

	Name	Description
	NleCompressEx [107]	Compresses the features extracted or generalized by the extraction routines to be stored or used in matching.
	NleDetectFace [108]	Finds one face region that best fits extraction requirements from the given image.
	NleDetectFaces [108]	Finds all face regions in the given image.
	NleDetectFacialFeatures [109]	Finds locations of facial feature points.
	NleDetectionDetailsClear [109]	Clears the detection details.
	NleExtract [110]	Fully automatically extracts feature template from the face that best fits extraction requirements from the given image.
	NleGeneralizeEx [110]	Generates generalized template from uncompressed set of templates.
	NleExtractEndEx [111]	Ends extracting face features from stream.
	NleExtractNextEx [111]	Extracts one features template from image sequence.
	NleCreate [112]	Creates a NLExtractor.
	NleExtractNextEx2 [112]	Extracts one features template from specified image.
	NleExtractStart [113]	Initializes features extraction routines that use image sequences to extract features.
	NleExtractStartEx [113]	Initializes features extraction routines to extract features.
	HNLExtractor [114]	Handle to NLExtractor object.
	NleExtractUsingDetails [114]	Extracts feature template from the face with given detection details from an image.

Macros

Name	Description
NLE_FEATURE_POINT_COUNT [120]	Defines the number of detected face feature points.
NleExtractEnd [121]	Ends extracting face features from stream.
NLEP_BLINK_CONFIDENCE_THRESHOLD [121]	Closed eyes absence/presence confidence threshold.
NLEP_DARK_GLASSES_CONFIDENCE_THRESHOLD [121]	Dark glasses on a face absence/presence confidence threshold.
NLEP_DETECT_ALL_FEATURE_POINTS [121]	Identifier specifying whether all facial feature points should be detected. If the value is NFalse [1041], only centers of eyes or base facial features are detected, otherwise all supported facial feature points are detected. The default value is NFalse [1041].
NLEP_DETECT_BASE_FEATURE_POINTS [121]	Identifier specifying whether base facial feature points should be detected during facial feature detection. Currently, base facial feature points are: center of right eye, center of left eye, nose tip and center of mouth. The default value is false.
NLEP_FACE_CONFIDENCE_THRESHOLD [122]	Identifier of type N_TYPE_DOUBLE [576]. Specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100].
NLEP_DETECT_BLINK [122]	Identifier used to detect closed eyes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLEP_DETECT_DARK_GLASSES [122]	Identifier used to detect faces with dark glasses.
NLEP_DETECT_EXPRESSION [122]	Identifier used to detect face expressions.
NLEP_DETECT_GENDER [122]	Identifier indicating whether to use gender detection procedure.
NLEP_FACE_QUALITY_THRESHOLD [122]	Identifier of type N_TYPE_BYTE [576]. Specifies the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.
NLEP_DETECT_GLASSES [122]	Identifier used to detect faces with glasses.
NLEP_DETECT_MORE_FACES [123]	Identifier used to choose different face detection procedures which result in high recall.
NLEP_FAVOR_LARGEST_FACE [123]	Identifier of type N_TYPE_BOOL [576]. If set to true the largest face found in an image will be used for facial feature extraction instead of the one with the highest face confidence threshold score.
NLEP_DETECT_MOUTH_OPEN [123]	Identifier used to detect faces with mouth open.
NLEP_EXPRESSION_CONFIDENCE_THRESHOLD [123]	Face expressions absence/presence confidence threshold.
NLEP_FACE_DETECTION_SCALE_COUNT [123]	Identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate. Values are in range [1..16].
NLEP_LIVENESS_THRESHOLD [123]	Identifier of type N_TYPE_DOUBLE [576]. This threshold checks if faces are extracted from a live image stream. The higher the value of this parameter, the more strictly the stream is checked if the face in it is real (for example if it's not forged by showing a photo of a person in front a camera). The value of this parameter can be in range [0..100]. The default value is 50.
NLEP_FACE_ROLL_ANGLE_BASE [123]	Identifier specifying a base angle for faces roll angle in the image to search. Available values are these: -180; -90; 0; 90; 180. The default value is 0. Identifier is used to set extractor base angle to support mobile devices with orientation sensor.
NLEP_GENDER_CONFIDENCE_THRESHOLD [124]	This threshold checks the gender of the captured face. The higher the value of this parameter, the more strictly the gender of the face is checked. The value of this parameter can be in range [0..100].
NLEP_MAX_IOD [124]	Identifier of type N_TYPE_INT [577] specifying maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000].
NLEP_GLASSES_CONFIDENCE_THRESHOLD [124]	Glasses on a face absence/presence confidence threshold.
NLEP_MAX_RECORDS_PER_TEMPLATE [124]	Identifier of type N_TYPE_INT [577]. Sets the maximum number of records an extraction function can return in one NLTemplate [153]. The value of this parameter can be in range [1..16]. The default value is 4.
NLEP_MAX_ROLL_ANGLE_DEVIATION [124]	Defines maximum roll angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180]. Parameter type - NShort [1008].
NLEP_MAX_STREAM_DURATION_IN_FRAMES [124]	Parameter identifier specifying maximum duration in frames to be used for template extraction from stream.
NLEP_MIN_IOD [124]	Identifier of type N_TYPE_INT [577] specifying minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLEP_MAX_YAW_ANGLE_DEVIATION [125]	Identifier specifying the maximum yaw angle deviation from frontal face in degrees. The value of this parameter can be in range [0..90]. The default value is 15. Currently supports deviations up to 45 degrees.
NLEP_MOUTH_OPEN_CONFIDENCE_THRESHOLD [125]	Face with mouth open absence/presence confidence threshold.
NLEP_USE_LIVENESS_CHECK [125]	Identifier of type N_TYPE_BOOL [576]. If set to true liveness check is performed while doing facial feature extraction from an image stream.
NLEP_TEMPLATE_SIZE [125]	Defines template size of Face template (NLTEmplate). Parameter type - NUShort [1010].

Structs, Records, Enums

	Name	Description
	NleRotation_ [115]	Enumerates face rotation values.
	NleTemplateSize_ [115]	Enumerates available template size values.
	NleRotation [115]	Enumerates face rotation values.
	NleTemplateSize [115]	Enumerates available template size values.
	NleDetectionDetails_ [115]	Structure with information about face detection results in a face detection routine.
	NleEyes_ [117]	Structure defining information of an eye pair.
	NleExtractionStatus_ [117]	Face features extraction status.
	NleFace_ [118]	Structure describing face region information
	NleEyes [117]	Structure defining information of an eye pair.
	NleExtractionStatus [117]	Face features extraction status.
	NleFace [118]	Structure describing face region information

Types

Name	Description
NleDetectionDetails [118]	Structure with information about face detection results in a face detection routine.

7.1.1.4.1 Functions

7.1.1.4.1.1 NleCompressEx Function

Compresses the features extracted or generalized by the extraction routines to be stored or used in matching.

C++

```
NResult N_API NleCompressEx(HNLTemplate hSrcTemplate, NleTemplateSize dstTemplateSize,
HNLTemplate * phDstTemplate);
```

Parameters

Parameters	Description
HNLTemplate hSrcTemplate	[in] Handle to the NLTemplate [153] object to be compressed.
NleTemplateSize dstTemplateSize	[in] Size of compressed template. Can be used small, medium or large template. For more information see NLEP_TEMPLATE_SIZE [125] parameter.
HNLTemplate * phDstTemplate	[out] Pointer to HNLTemplate [165] that receives handle to created NLTemplate [153] object.

Returns

If the function succeeds, the return value is N_OK [433] or NSucceeded [434]. Otherwise, error code is returned.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NLExtractor Unit [104]

7.1.1.4.1.2 NleDetectFace Function

Finds one face region that best fits extraction requirements from the given image.

C++

```
NResult N_API NleDetectFace(HNLExtractor hExtractor, HNGrayscaleImage hImage, NBool *
pDetected, NleFace * pFace);
```

Parameters

Parameters	Description
HNLExtractor hExtractor	[in] Handle of the NLExtractor [104] object.
HNGrayscaleImage hImage	[in] Handle of the source image.
NBool * pDetected	[out] Pointer to NBool [1004] variable that is set to true if face is found in the image else it's false
NleFace * pFace	[out] Pointer to the NleFace [118] structure of found face.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hExtractor, hImage*, *pDetected* or *pFace* is NULL [1075].

Module

NLExtractor Unit [104]

7.1.1.4.1.3 NleDetectFaces Function

Finds all face regions in the given image.

C++

```
NResult N_API NleDetectFaces(HNLExtractor hExtractor, HNGrayscaleImage hImage, NInt *
pFaceCount, NleFace * * parFaces);
```

Parameters

Parameters	Description
HNLExtractor hExtractor	[in] Handle to the NLExtractor [104] object.
HNGrayscaleImage hImage	[in] Handle to the source image.
NInt * pFaceCount	[out] Pointer to the NInt [1006] variable that is set to the number of found faces in the image.
NleFace * * parFaces	[out] Pointer to an array of NleFace [118] structures of found faces.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hExtractor*, *hImage*, *pFaceCount* or *parFaces* is NULL [1075].

Remarks

The buffer for the *parFaces* variable is allocated inside this function automatically, but you must free this variable yourself when it becomes unnecessary for you. It must be freed using NCore [257] library function NFree [506].

Module

NLExtractor Unit [104]

7.1.1.4.1.4 NleDetectFacialFeatures Function

Finds locations of facial feature points.

C++

```
NResult N_API NleDetectFacialFeatures(HNLExtractor hExtractor, HNGrayscaleImage hImage,
const NleFace * pFace, struct NleDetectionDetails_ * pDetectionDetails);
```

Parameters

Parameters	Description
HNLExtractor hExtractor	[in] Handle to the NLExtractor [104] object.
HNGrayscaleImage hImage	[in] Handle to the source image.
const NleFace * pFace	[out] Pointer to the NleFace [118] structure of a face found in the image hImage.
struct NleDetectionDetails_ * pDetectionDetails	[out] Pointer to the NleDetectionDetails [118] structure of face detection results in the image.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hExtractor*, *hImage*, *pFace* or *pDetectionDetails* is NULL [1075].

Module

NLExtractor Unit [104]

7.1.1.4.1.5 NleDetectionDetailsClear Function

Clears the detection details.

C++

```
NResult N_API NleDetectionDetailsClear(struct NleDetectionDetails_ * pDetectionDetails);
```

Parameters

Parameters	Description
struct NleDetectionDetails_ * pDetectionDetails	[out] Pointer to the cleared NleDetectionDetails [118] structure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NLExtractor Unit [104]

7.1.1.4.1.6 NleExtract Function

Fully automatically extracts feature template from the face that best fits extraction requirements from the given image.

C++

```
NResult N_API NleExtract(HNLExtractor hExtractor, HNGrayscaleImage hImage, struct
NleDetectionDetails_ * pDetectionDetails, NleExtractionStatus * pStatus, HNLTemplate *
phTemplate);
```

Parameters

Parameters	Description
HNLExtractor hExtractor	[in] Handle to the NLExtractor [104] object.
HNGrayscaleImage hImage	[in] Handle to the source image.
struct NleDetectionDetails_ * pDetectionDetails	[out] Pointer to the NleDetectionDetails [118] structure of face detection results in the image.
NleExtractionStatus * pStatus	[out] Pointer to NleExtractionStatus [117] that receives extraction status.
HNLTemplate * phTemplate	[out] Pointer to HNLTemplate [165] that receives handle to created NLTemplate [153] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hExtractor*, *hImage*, *pDetectionDetails*, *pStatus* or *phTemplate* is NULL [1075].

Module

NLExtractor Unit [104]

7.1.1.4.1.7 NleGeneralizeEx Function

Generates generalized template from uncompressed set of templates.

C++

```
NResult N_API NleGeneralizeEx(HNLExtractor hExtractor, NInt templateCount, const
HNLTemplate * arhTemplates, NleExtractionStatus * pStatus, NInt * pBaseTemplateIndex,
HNLTemplate * phTemplate);
```

Parameters

Parameters	Description
HNLExtractor hExtractor	[in] Handle to the NLExtractor [104] object.
NInt templateCount	[in] Count of templates to be used in generalization.
const HNLTemplate * arhTemplates	[out] Pointer to memory buffer that contains NLTemplate [153] templates.
NleExtractionStatus * pStatus	[out] Template extraction status.
NInt * pBaseTemplateIndex	[out] Index of main generalization template.
HNLTemplate * phTemplate	[out] Pointer to HNLTemplatethat receives handle to created NLTemplate [153] object.

7

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NLExtractor Unit [104]

7.1.1.4.1.8 NleExtractEndEx Function

Ends extracting face features from stream.

C++

```
NResult N_API NleExtractEndEx(HNLExtractor hExtractor, NInt * pBaseFrameIndex, struct
NleDetectionDetails_ * pDetectionDetails, NleExtractionStatus * pStatus, HNLTemplate *
phTemplate);
```

Parameters

Parameters	Description
HNLExtractor hExtractor	[in] Handle to the NLExtractor [104] object.
NInt * pBaseFrameIndex	[out] Index of a frame which was used to create the template.
struct NleDetectionDetails_ * pDetectionDetails	[out] Pointer to NleDetectionDetails [118] that receives detection details.
NleExtractionStatus * pStatus	[out] Pointer to NleExtractionStatus [117] that receives extraction status.
HNLTemplate * phTemplate	[out] Pointer to HNLTemplate [165] that receives handle to created NLTemplate [153] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NLExtractor Unit [104]

7.1.1.4.1.9 NleExtractNextEx Function

Extracts one features template from image sequence.

C++

```
NResult N_API NleExtractNextEx(HNLExtractor hExtractor, HNGrayscaleImage hImage, struct
NleDetectionDetails_ * pDetectionDetails, NInt * pBaseFrameIndex, HNLTemplate * phTemplate,
NleExtractionStatus * pStatus);
```

Parameters

Parameters	Description
HNLExtractor hExtractor	[in] Handle to the NLExtractor [104] object.
HNGrayscaleImage hImage	[in] Handle to the source image.
struct NleDetectionDetails_ * pDetectionDetails	[out] Pointer to the NleDetectionDetails [118] structure of face detection results in the image.
NInt * pBaseFrameIndex	[out] ?Index of base frame.
HNLTemplate * phTemplate	[out] Pointer to HNLTemplatethat receives handle to created NLTemplate [153] object.
NleExtractionStatus * pStatus	[out] Pointer to NleExtractionStatus [117] that receives extraction status.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Notes

This function is obsolete. Use NleExtractNextEx2 [112] instead.

Module

NLExtractor Unit [104]

7.1.1.4.1.10 NleCreate Function

Creates a NLExtractor [104].

C++

```
NResult N_API NleCreate(HNLExtractor * phExtractor);
```

Parameters

Parameters	Description
HNLExtractor * phExtractor	[out] Pointer to HNLExtractor [114] that receives handle to created NLExtractor [104] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*phExtractor* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

Created object must be deleted using NObjectFree [542] function.

Module

NLExtractor Unit [104]

7.1.1.4.1.11 NleExtractNextEx2 Function

Extracts one features template from specified image.

C++

```
NResult N_API NleExtractNextEx2(HNLExtractor hExtractor, HNGrayscaleImage hImage, struct
NleDetectionDetails_ * pDetectionDetails, NleExtractionStatus * pStatus);
```

Parameters

Parameters	Description
HNLExtractor hExtractor	[in] Handle to the NLExtractor [104] object.
HNGrayscaleImage hImage	[in] Handle to the source image.
struct NleDetectionDetails_ * pDetectionDetails	[out] Pointer to the NleDetectionDetails [118] structure of face detection results in the image.
NleExtractionStatus * pStatus	[out] Pointer to NleExtractionStatus [117] that receives extraction status.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NLExtractor Unit [104]

7.1.1.4.1.12 NleExtractStart Function

Initializes features extraction routines that use image sequences to extract features.

C++

```
NResult N_API NleExtractStart(HNLExtractor hExtractor, NInt durationInFrames);
```

Parameters

Parameters	Description
HNLExtractor hExtractor	[in] Handle to the NLExtractor [104] object.
NInt durationInFrames	[in] Maximum number of frames which will be used for one feature template extraction.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hExtractor* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	*durationInFrames* is smaller than 1.

Notes

This function is obsolete. Use NleExtractStartEx [113] instead.

Module

NLExtractor Unit [104]

7.1.1.4.1.13 NleExtractStartEx Function

Initializes features extraction routines to extract features.

C++

```
NResult N_API NleExtractStartEx(HNLExtractor hExtractor);
```

Parameters

Parameters	Description
HNLExtractor hExtractor	[in] Handle to the NLExtractor [104] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hExtractor* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	*durationInFrames* is smaller than 1.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NLExtractor Unit [104]

7.1.1.4.1.14 HNLExtractor

Handle to NLExtractor [104] object.

Remarks

The following parameters can be passed to function with corresponding handle:

- NLEP_MIN_IOD [124]
- NLEP_MAX_IOD [124]
- NLEP_FACE_CONFIDENCE_THRESHOLD [122]
- NLEP_FACE_QUALITY_THRESHOLD [122]
- NLEP_GENERALIZATION_THRESHOLD

Module

NLExtractor Unit [104]

7.1.1.4.1.15 NleExtractUsingDetails Function

Extracts feature template from the face with given detection details from an image.

C++

```
NResult N_API NleExtractUsingDetails(HNLExtractor hExtractor, HNGrayscaleImage hImage,
struct NleDetectionDetails_ * pDetectionDetails, NleExtractionStatus * pStatus, HNLTemplate
* phTemplate);
```

Parameters

Parameters	Description
HNLExtractor hExtractor	[in] Handle to the NLExtractor [104] object.
HNGrayscaleImage hImage	[in] Handle to the source image.
struct NleDetectionDetails_ * pDetectionDetails	[out] Pointer to the NleDetectionDetails [118] structure of face detection results in the image.
NleExtractionStatus * pStatus	[out] Pointer to NleExtractionStatus [117] that receives extraction status.
HNLTemplate * phTemplate	[out] Pointer to HNLTemplate [165] that receives handle to created NLTemplate [153] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hExtractor*, *hImage*, *pDetectionDetails*, *pStatus* or *phTemplate* is NULL [1075].

Module

NLExtractor Unit [104]

7.1.1.4.2 Structs, Records, Enums

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.4.2.1 NleRotation Structure

Enumerates face rotation values.

C++

```
typedef struct NleRotation_ {
  NShort Yaw;
  NShort Pitch;
  NShort Roll;
} NleRotation;
```

Members

Members	Description
NShort Yaw;	Rotation about vertical (y) axis.
NShort Pitch;	Rotation about the horizontal side-to-side (x) horizontal axis.
NShort Roll;	Rotation about the horizontal back-to-front (z) axis.

Module

NLExtractor Unit [104]

7.1.1.4.2.2 NleTemplateSize Enumeration

Enumerates available template size values.

C++

```
typedef enum NleTemplateSize_ {
  nletsSmall = 0,
  nletsCompact = 1,
  nletsMedium = 64,
  nletsLarge = 128
} NleTemplateSize;
```

Members

Members	Description
nletsSmall = 0	The small faces template size. This template should be used when the highest matching speed is required.
nletsCompact = 1	The compact NLRecord [134] size (~2.2KB, used for smartcards).
nletsMedium = 64	Medium faces template size. This is a default size for identification task.
nletsLarge = 128	Large template size. This template size should be used when the highest matching accuracy is needed.

Module

NLExtractor Unit [104]

7.1.1.4.2.3 NleDetectionDetails_ Structure

Structure with information about face detection results in a face detection routine.

C++

```
struct NleDetectionDetails_ {
  NBool FaceAvailable;
  NInt Padding1;
  NleFace Face;
  NInt Padding2;
  NGender Gender;
  NLExpression Expression;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
    NLProperties Properties;
    NByte GenderConfidence;
    NByte ExpressionConfidence;
    NByte GlassesConfidence;
    NByte DarkGlassesConfidence;
    NByte BlinkConfidence;
    NByte MouthOpenConfidence;
    NByte ReservedConfidence[6];
    struct NLFeaturePoint_ RightEyeCenter;
    struct NLFeaturePoint_ LeftEyeCenter;
    struct NLFeaturePoint_ NoseTip;
    struct NLFeaturePoint_ MouthCenter;
    struct NLFeaturePoint_ Reserved[4];
    struct NLFeaturePoint_ Points[NLE_FEATURE_POINT_COUNT];
  };
```

Members

Members	Description
NBool FaceAvailable;	Variable showing if a face region was found in an image.
NInt Padding1;	Variable representing padding.
NleFace Face;	Structure with face region information if face was found.
NInt Padding2;	Variable representing padding.
NGender Gender;	Structure with gender information.
NLExpression Expression;	Structure with face expression information.
NLProperties Properties;	Flags with face properties information. These values are available: Glasses (NLEP_DETECT_GLASSES [122]), DarkGlasses (NLEP_DETECT_DARK_GLASSES [122]), Blink (NLEP_DETECT_BLINK [122]) and MouthOpen (NLEP_DETECT_MOUTH_OPEN [123]).
NByte GenderConfidence;	Variable representing gender confidence.
NByte ExpressionConfidence;	Variable representing face expression confidence. Can get values [0; 100] which mean if expression features were found on a face or not. Also can get values 254 (face expressions were not calculated) and 255 (face expression were calculated but confidence was lower than threshold NLEP_EXPRESSION_CONFIDENCE_THRESHOLD [123]).
NByte GlassesConfidence;	Variable representing glasses confidence.
NByte DarkGlassesConfidence;	Variable representing dark glasses confidence. Can get values [0; 100] which mean if dark glasses features were found on a face or not. Also can get values 254 (dark glasses were not calculated) and 255 (dark glasses were calculated but confidence was lower than threshold NLEP_DARK_GLASSES_CONFIDENCE_THRESHOLD [121]).
NByte BlinkConfidence;	Variable representing blink (eyes closed) confidence. Can get values [0; 100] which mean if blink features were found on a face or not. Also can get values 254 (blink was not calculated) and 255 (blink was calculated but confidence was lower than threshold NLEP_BLINK_CONFIDENCE_THRESHOLD [121]).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NByte MouthOpenConfidence;	Variable representing mouth open confidence. Can get values [0; 100] which mean if mouth open features were found on a face or not. Also can get values 254 (mouth open features were not calculated) and 255 (mouth open features were calculated but confidence was lower than threshold).
NByte ReservedConfidence[6];	Variable representing reserved confidence.
struct NLFeaturePoint_ RightEyeCenter;	The position of the right eye center.
struct NLFeaturePoint_ LeftEyeCenter;	The position of the left eye center.
struct NLFeaturePoint_ NoseTip;	The position of the nose tip.
struct NLFeaturePoint_ MouthCenter;	The position of the mouth center.
struct NLFeaturePoint_ Reserved[4];	Structure with reserved information.
struct NLFeaturePoint_ Points[NLE_FEATURE_POINT_COUNT];	Structure with facial feature point information.

Module

NLExtractor Unit [104]

7.1.1.4.2.4 NleEyes Structure

Structure defining information of an eye pair.

C++

```
typedef struct NleEyes_ {
  struct NPoint_ First;
  NDouble FirstConfidence;
  struct NPoint_ Second;
  NDouble SecondConfidence;
} NleEyes;
```

Members

Members	Description
struct NPoint_ First;	First eye coordinates.
NDouble FirstConfidence;	How confidently the first eye was found.
struct NPoint_ Second;	Second eye coordinates.
NDouble SecondConfidence;	How confidently the second eye was found.

Module

NLExtractor Unit [104]

7.1.1.4.2.5 NleExtractionStatus Enumeration

Face features extraction status.

C++

```
typedef enum NleExtractionStatus_ {
  nleesNone = 0,
  nleesTemplateCreated = 1,
  nleesFaceNotDetected = 2,
  nleesEyesNotDetected = 3,
  nleesFaceTooCloseToImageBorder = 4,
  nleesQualityCheckGrayscaleDensityFailed = 100,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
  nleesQualityCheckExposureFailed = 101,
  nleesQualityCheckSharpnessFailed = 102,
  nleesLivenessCheckFailed = 200,
  nleesGeneralizationFailed = 300
} NleExtractionStatus;
```

Members

Members	Description
nleesNone = 0	No extraction status usage.
nleesTemplateCreated = 1	Template created.
nleesFaceNotDetected = 2	Face not detected.
nleesEyesNotDetected = 3	Eyes not detected.
nleesFaceTooCloseToImageBorder = 4	Face too close to image border.
nleesQualityCheckGrayscaleDensityFailed = 100	Quality check grayscale density failed.
nleesQualityCheckExposureFailed = 101	Quality check exposure failed.
nleesQualityCheckSharpnessFailed = 102	Quality check sharpness failed.
nleesLivenessCheckFailed = 200	Liveness check failed.
nleesGeneralizationFailed = 300	Indicate that function failed to generalize extracted features.

Module

NLExtractor Unit [104]

7.1.1.4.2.6 NleFace Structure

Structure describing face region information

C++

```
typedef struct NleFace_ {
  struct NRect_ Rectangle;
  NleRotation Rotation;
  NDouble Confidence;
} NleFace;
```

Members

Members	Description
struct NRect_ Rectangle;	Structure with face rectangle information.
NleRotation Rotation;	Face rotation structure.
NDouble Confidence;	How confidently the face region was found.

Module

NLExtractor Unit [104]

7.1.1.4.3 Types

7.1.1.4.3.1 NleDetectionDetails Type

Structure with information about face detection results in a face detection routine.

C++

```
typedef struct NleDetectionDetails_ NleDetectionDetails;
```

Module

NLExtractor Unit [104]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.4.4 Macros

To get a parameter value, use NObjectGetParameterEx function. To set parameter value - NObjectSetParameterEx function.

The following values can be used for *parameterId*:

- NLEP_MIN_IOD [124]
- NLEP_MAX_IOD [124]
- NLEP_FACE_CONFIDENCE_THRESHOLD [122]
- NLEP_FACE_QUALITY_THRESHOLD [122]
- NLEP_GENERALIZATION_THRESHOLD.

Macros

Name	Description
NLE_FEATURE_POINT_COUNT [120]	Defines the number of detected face feature points.
NleExtractEnd [121]	Ends extracting face features from stream.
NLEP_BLINK_CONFIDENCE_THRESHOLD [121]	Closed eyes absence/presence confidence threshold.
NLEP_DARK_GLASSES_CONFIDENCE_THRESHOLD [121]	Dark glasses on a face absence/presence confidence threshold.
NLEP_DETECT_ALL_FEATURE_POINTS [121]	Identifier specifying whether all facial feature points should be detected. If the value is NFalse [1041], only centers of eyes or base facial features are detected, otherwise all supported facial feature points are detected. The default value is NFalse [1041].
NLEP_DETECT_BASE_FEATURE_POINTS [121]	Identifier specifying whether base facial feature points should be detected during facial feature detection. Currently, base facial feature points are: center of right eye, center of left eye, nose tip and center of mouth. The default value is false.
NLEP_FACE_CONFIDENCE_THRESHOLD [122]	Identifier of type N_TYPE_DOUBLE [576]. Specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100].
NLEP_DETECT_BLINK [122]	Identifier used to detect closed eyes.
NLEP_DETECT_DARK_GLASSES [122]	Identifier used to detect faces with dark glasses.
NLEP_DETECT_EXPRESSION [122]	Identifier used to detect face expressions.
NLEP_DETECT_GENDER [122]	Identifier indicating whether to use gender detection procedure.
NLEP_FACE_QUALITY_THRESHOLD [122]	Identifier of type N_TYPE_BYTE [576]. Specifies the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.
NLEP_DETECT_GLASSES [122]	Identifier used to detect faces with glasses.
NLEP_DETECT_MORE_FACES [123]	Identifier used to choose different face detection procedures which result in high recall.
NLEP_FAVOR_LARGEST_FACE [123]	Identifier of type N_TYPE_BOOL [576]. If set to true the largest face found in an image will be used for facial feature extraction instead of the one with the highest face confidence threshold score.
NLEP_DETECT_MOUTH_OPEN [123]	Identifier used to detect faces with mouth open.
NLEP_EXPRESSION_CONFIDENCE_THRESHOLD [123]	Face expressions absence/presence confidence threshold.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLEP_FACE_DETECTION_SCALE_COUNT [123]	Identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate. Values are in range [1..16].
NLEP_LIVENESS_THRESHOLD [123]	Identifier of type N_TYPE_DOUBLE [576]. This threshold checks if faces are extracted from a live image stream. The higher the value of this parameter, the more strictly the stream is checked if the face in it is real (for example if it's not forged by showing a photo of a person in front a camera). The value of this parameter can be in range [0..100]. The default value is 50.
NLEP_FACE_ROLL_ANGLE_BASE [123]	Identifier specifying a base angle for faces roll angle in the image to search. Available values are these: -180; -90; 0; 90; 180. The default value is 0. Identifier is used to set extractor base angle to support mobile devices with orientation sensor.
NLEP_GENDER_CONFIDENCE_THRESHOLD [124]	This threshold checks the gender of the captured face. The higher the value of this parameter, the more strictly the gender of the face is checked. The value of this parameter can be in range [0..100].
NLEP_MAX_IOD [124]	Identifier of type N_TYPE_INT [577] specifying maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000].
NLEP_GLASSES_CONFIDENCE_THRESHOLD [124]	Glasses on a face absence/presence confidence threshold.
NLEP_MAX_RECORDS_PER_TEMPLATE [124]	Identifier of type N_TYPE_INT [577]. Sets the maximum number of records an extraction function can return in one NLTemplate [153]. The value of this parameter can be in range [1..16]. The default value is 4.
NLEP_MAX_ROLL_ANGLE_DEVIATION [124]	Defines maximum roll angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180]. Parameter type - NShort [1008].
NLEP_MAX_STREAM_DURATION_IN_FRAMES [124]	Parameter identifier specifying maximum duration in frames to be used for template extraction from stream.
NLEP_MIN_IOD [124]	Identifier of type N_TYPE_INT [577] specifying minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000].
NLEP_MAX_YAW_ANGLE_DEVIATION [125]	Identifier specifying the maximum yaw angle deviation from frontal face in degrees. The value of this parameter can be in range [0..90]. The default value is 15. Currently supports deviations up to 45 degrees.
NLEP_MOUTH_OPEN_CONFIDENCE_THRESHOLD [125]	Face with mouth open absence/presence confidence threshold.
NLEP_USE_LIVENESS_CHECK [125]	Identifier of type N_TYPE_BOOL [576]. If set to true liveness check is performed while doing facial feature extraction from an image stream.
NLEP_TEMPLATE_SIZE [125]	Defines template size of Face template (NLTEmplate). Parameter type - NUShort [1010].

Module

NLExtractor Unit [104]

7.1.1.4.4.1 NLE_FEATURE_POINT_COUNT Macro

Defines the number of detected face feature points.

C++

```
#define NLE_FEATURE_POINT_COUNT 68
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

See Also

NleDetectionDetails [118]

7.1.1.4.4.2 NleExtractEnd Macro

Ends extracting face features from stream.

C++

```
#define NleExtractEnd(hExtractor, pBaseFrameIndex, pStatus, phTemplate)
NleExtractEndEx(hExtractor, pBaseFrameIndex, NULL, pStatus, phTemplate)
```

Parameters

Parameters	Description
hExtractor	[in] Handle to the NLExtractor [104] object.
pBaseFrameIndex	[out] Index of a frame which was used to create the template.
pStatus	[out] Pointer to NleExtractionStatus [117] that receives extraction status.
phTemplate	[out] Pointer to HNLTemplate [165] that receives handle to created NLTemplate [153] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

7.1.1.4.4.3 NLEP_BLINK_CONFIDENCE_THRESHOLD Macro

Closed eyes absence/presence confidence threshold.

C++

```
#define NLEP_BLINK_CONFIDENCE_THRESHOLD 10460
```

7.1.1.4.4.4 NLEP_DARK_GLASSES_CONFIDENCE_THRESHOLD Macro

Dark glasses on a face absence/presence confidence threshold.

C++

```
#define NLEP_DARK_GLASSES_CONFIDENCE_THRESHOLD 10456
```

7.1.1.4.4.5 NLEP_DETECT_ALL_FEATURE_POINTS Macro

Identifier specifying whether all facial feature points should be detected. If the value is NFalse [1041], only centers of eyes or base facial features are detected, otherwise all supported facial feature points are detected. The default value is NFalse [1041].

C++

```
#define NLEP_DETECT_ALL_FEATURE_POINTS 10250
```

See Also

Face Feature Points [46]

7.1.1.4.4.6 NLEP_DETECT_BASE_FEATURE_POINTS Macro

Identifier specifying whether base facial feature points should be detected during facial feature detection. Currently, base facial feature points are: center of right eye, center of left eye, nose tip and center of mouth. The default value is false.

C++

```
#define NLEP_DETECT_BASE_FEATURE_POINTS 10255
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

See Also

NLEP_DETECT_ALL_FEATURE_POINTS [121] - for all face features detection.

7.1.1.4.4.7 NLEP_FACE_CONFIDENCE_THRESHOLD Macro

Identifier of type N_TYPE_DOUBLE [576]. Specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100].

C++

```
#define NLEP_FACE_CONFIDENCE_THRESHOLD 10103
```

7.1.1.4.4.8 NLEP_DETECT_BLINK Macro

Identifier used to detect closed eyes.

C++

```
#define NLEP_DETECT_BLINK 10459
```

7.1.1.4.4.9 NLEP_DETECT_DARK_GLASSES Macro

Identifier used to detect faces with dark glasses.

C++

```
#define NLEP_DETECT_DARK_GLASSES 10455
```

7.1.1.4.4.10 NLEP_DETECT_EXPRESSION Macro

Identifier used to detect face expressions.

C++

```
#define NLEP_DETECT_EXPRESSION 10457
```

7.1.1.4.4.11 NLEP_DETECT_GENDER Macro

Identifier indicating whether to use gender detection procedure.

C++

```
#define NLEP_DETECT_GENDER 10451
```

Remarks

By enabling this parameter (setting to NTrue [1064]) you can ask to estimate person's gender from the face during facial features template extraction. Gender will be estimated correct with higher confidence if the provided face image passes all face image quality checks and facial features template is created. Additionally, gender estimator works more accurate on near-frontal face images.

7.1.1.4.4.12 NLEP_FACE_QUALITY_THRESHOLD Macro

Identifier of type N_TYPE_BYTE [576]. Specifies the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.

C++

```
#define NLEP_FACE_QUALITY_THRESHOLD 10350
```

7.1.1.4.4.13 NLEP_DETECT_GLASSES Macro

Identifier used to detect faces with glasses.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NLEP_DETECT_GLASSES 10453
```

7.1.1.4.4.14 NLEP_DETECT_MORE_FACES Macro

Identifier used to choose different face detection procedures which result in high recall.

C++

```
#define NLEP_DETECT_MORE_FACES 10110
```

7.1.1.4.4.15 NLEP_FAVOR_LARGEST_FACE Macro

Identifier of type N_TYPE_BOOL [576]. If set to true the largest face found in an image will be used for facial feature extraction instead of the one with the highest face confidence threshold score.

C++

```
#define NLEP_FAVOR_LARGEST_FACE 10104
```

7.1.1.4.4.16 NLEP_DETECT_MOUTH_OPEN Macro

Identifier used to detect faces with mouth open.

C++

```
#define NLEP_DETECT_MOUTH_OPEN 10461
```

7.1.1.4.4.17 NLEP_EXPRESSION_CONFIDENCE_THRESHOLD Macro

Face expressions absence/presence confidence threshold.

C++

```
#define NLEP_EXPRESSION_CONFIDENCE_THRESHOLD 10458
```

7.1.1.4.4.18 NLEP_FACE_DETECTION_SCALE_COUNT Macro

Identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate. Values are in range [1..16].

C++

```
#define NLEP_FACE_DETECTION_SCALE_COUNT 10111
```

7.1.1.4.4.19 NLEP_LIVENESS_THRESHOLD Macro

Identifier of type N_TYPE_DOUBLE [576]. This threshold checks if faces are extracted from a live image stream. The higher the value of this parameter, the more strictly the stream is checked if the face in it is real (for example if it's not forged by showing a photo of a person in front a camera). The value of this parameter can be in range [0..100]. The default value is 50.

C++

```
#define NLEP_LIVENESS_THRESHOLD 10403
```

7.1.1.4.4.20 NLEP_FACE_ROLL_ANGLE_BASE Macro

Identifier specifying a base angle for faces roll angle in the image to search. Available values are these: -180; -90; 0; 90; 180. The default value is 0. Identifier is used to set extractor base angle to support mobile devices with orientation sensor.

C++

```
#define NLEP_FACE_ROLL_ANGLE_BASE 10117
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.4.4.21 NLEP_GENDER_CONFIDENCE_THRESHOLD Macro

This threshold checks the gender of the captured face. The higher the value of this parameter, the more strictly the gender of the face is checked. The value of this parameter can be in range [0..100].

C++

```
#define NLEP_GENDER_CONFIDENCE_THRESHOLD 10452
```

7.1.1.4.4.22 NLEP_MAX_IOD Macro

Identifier of type N_TYPE_INT [577] specifying maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000].

C++

```
#define NLEP_MAX_IOD 10102
```

Notes

If MinIOD and/or MaxIOD will be set very similar to the real distance between the eyes, it could cause to not so accurate results (of face detection).

When the face is set MinIOD and MaxIOD the face could be found up to ~1.5 times bigger/smaller value it is set.

7.1.1.4.4.23 NLEP_GLASSES_CONFIDENCE_THRESHOLD Macro

Glasses on a face absence/presence confidence threshold.

C++

```
#define NLEP_GLASSES_CONFIDENCE_THRESHOLD 10454
```

7.1.1.4.4.24 NLEP_MAX_RECORDS_PER_TEMPLATE Macro

Identifier of type N_TYPE_INT [577]. Sets the maximum number of records an extraction function can return in one NLTemplate [153]. The value of this parameter can be in range [1..16]. The default value is 4.

C++

```
#define NLEP_MAX_RECORDS_PER_TEMPLATE 10408
```

7.1.1.4.4.25 NLEP_MAX_ROLL_ANGLE_DEVIATION Macro

Defines maximum roll angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180]. Parameter type - NShort [1008].

C++

```
#define NLEP_MAX_ROLL_ANGLE_DEVIATION 10105
```

Remarks

Matching speed highly depends on a roll angle deviation from frontal face.

7.1.1.4.4.26 NLEP_MAX_STREAM_DURATION_IN_FRAMES Macro

Parameter identifier specifying maximum duration in frames to be used for template extraction from stream.

C++

```
#define NLEP_MAX_STREAM_DURATION_IN_FRAMES 10409
```

7.1.1.4.4.27 NLEP_MIN_IOD Macro

Identifier of type N_TYPE_INT [577] specifying minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NLEP_MIN_IOD 10101
```

Notes

If MinIOD and/or MaxIOD will be set very similar to the real distance between the eyes, it could cause to not so accurate results (of face detection).

When the face is set MinIOD and MaxIOD the face could be found up to ~1.5 times bigger/smaller value it is set.

7.1.1.4.4.28 NLEP_MAX_YAW_ANGLE_DEVIATION Macro

Identifier specifying the maximum yaw angle deviation from frontal face in degrees. The value of this parameter can be in range [0..90]. The default value is 15. Currently supports deviations up to 45 degrees.

C++

```
#define NLEP_MAX_YAW_ANGLE_DEVIATION 10109
```

7.1.1.4.4.29 NLEP_MOUTH_OPEN_CONFIDENCE_THRESHOLD Macro

Face with mouth open absence/presence confidence threshold.

C++

```
#define NLEP_MOUTH_OPEN_CONFIDENCE_THRESHOLD 10462
```

7.1.1.4.4.30 NLEP_USE_LIVENESS_CHECK Macro

Identifier of type N_TYPE_BOOL [576]. If set to true liveness check is performed while doing facial feature extraction from an image stream.

C++

```
#define NLEP_USE_LIVENESS_CHECK 10402
```

7.1.1.4.4.31 NLEP_TEMPLATE_SIZE Macro

Defines template size of Face template (NLTEmplate). Parameter type - NUShort [1010].

C++

```
#define NLEP_TEMPLATE_SIZE 10360
```

Remarks

Compact, small, medium or large template can be used. Matching speed and accuracy depends on chosen template size. The maximum speed is achieved when small template is used. But the most accurate matching results are when large template is used.

By default when enrolling large template is saved to database. But when performing identification, by default medium template size is used.

Compact, small, medium and large templates are fully compatible with older versions of VeriLook SDK.

See Also

For more information about template sizes see NleTemplateSize [115] enumeration.

7.1.1.4.5 Files

7.1.1.4.5.1 NLExtractor.h

Header file for the NLExtractor [104] Unit. NLExtractor [104] is a part of NBiometrics library intended for human face finding in pictures and extracting their features into Neurotechnology Faces [45] Template(NLTemplate [153]).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Enumerations

	Name	Description
	NleExtractionStatus_ [117]	Face features extraction status.
	NleExtractionStatus [117]	Face features extraction status.

Functions

	Name	Description
	NleCompressEx [107]	Compresses the features extracted or generalized by the extraction routines to be stored or used in matching.
	NleCreate [112]	Creates a NLExtractor [104].
	NleDetectFace [108]	Finds one face region that best fits extraction requirements from the given image.
	NleDetectFaces [108]	Finds all face regions in the given image.
	NleDetectFacialFeatures [109]	Finds locations of facial feature points.
	NleDetectionDetailsClear [109]	Clears the detection details.
	NleExtract [110]	Fully automatically extracts feature template from the face that best fits extraction requirements from the given image.
	NleExtractEndEx [111]	Ends extracting face features from stream.
	NleExtractNextEx [111]	Extracts one features template from image sequence.
	NleExtractNextEx2 [112]	Extracts one features template from specified image.
	NleExtractStart [113]	Initializes features extraction routines that use image sequences to extract features.
	NleExtractStartEx [113]	Initializes features extraction routines to extract features.
	NleExtractUsingDetails [114]	Extracts feature template from the face with given detection details from an image.
	NleGeneralizeEx [110]	Generates generalized template from uncompressed set of templates.

Macros

Name	Description
NLE_FEATURE_POINT_COUNT [120]	Defines the number of detected face feature points.
NleExtractEnd [121]	Ends extracting face features from stream.

Module

NLExtractor Unit [104]

Structures

	Name	Description
	NleDetectionDetails_ [115]	Structure with information about face detection results in a face detection routine.
	NleEyes_ [117]	Structure defining information of an eye pair.
	NleFace_ [118]	Structure describing face region information
	NleRotation_ [115]	Enumerates face rotation values.
	NleEyes [117]	Structure defining information of an eye pair.
	NleFace [118]	Structure describing face region information
	NleRotation [115]	Enumerates face rotation values.

Types

Name	Description
NleDetectionDetails [118]	Structure with information about face detection results in a face detection routine.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.4.5.2 NLExtractorParams.h

C header file which contains NLExtractor [104] parameters.

Enumerations

	Name	Description
	NleTemplateSize_ [115]	Enumerates available template size values.
	NleTemplateSize [115]	Enumerates available template size values.

Macros

Name	Description
NLEP_BLINK_CONFIDENCE_THRESHOLD [121]	Closed eyes absence/presence confidence threshold.
NLEP_DARK_GLASSES_CONFIDENCE_THRESHOLD [121]	Dark glasses on a face absence/presence confidence threshold.
NLEP_DETECT_ALL_FEATURE_POINTS [121]	Identifier specifying whether all facial feature points should be detected. If the value is NFalse [1041], only centers of eyes or base facial features are detected, otherwise all supported facial feature points are detected. The default value is NFalse [1041].
NLEP_DETECT_BASE_FEATURE_POINTS [121]	Identifier specifying whether base facial feature points should be detected during facial feature detection. Currently, base facial feature points are: center of right eye, center of left eye, nose tip and center of mouth. The default value is false.
NLEP_DETECT_BLINK [122]	Identifier used to detect closed eyes.
NLEP_DETECT_DARK_GLASSES [122]	Identifier used to detect faces with dark glasses.
NLEP_DETECT_EXPRESSION [122]	Identifier used to detect face expressions.
NLEP_DETECT_GENDER [122]	Identifier indicating whether to use gender detection procedure.
NLEP_DETECT_GLASSES [122]	Identifier used to detect faces with glasses.
NLEP_DETECT_MORE_FACES [123]	Identifier used to choose different face detection procedures which result in high recall.
NLEP_DETECT_MOUTH_OPEN [123]	Identifier used to detect faces with mouth open.
NLEP_EXPRESSION_CONFIDENCE_THRESHOLD [123]	Face expressions absence/presence confidence threshold.
NLEP_FACE_CONFIDENCE_THRESHOLD [122]	Identifier of type N_TYPE_DOUBLE [576]. Specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100].
NLEP_FACE_DETECTION_SCALE_COUNT [123]	Identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate. Values are in range [1..16].
NLEP_FACE_QUALITY_THRESHOLD [122]	Identifier of type N_TYPE_BYTE [576]. Specifies the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.
NLEP_FACE_ROLL_ANGLE_BASE [123]	Identifier specifying a base angle for faces roll angle in the image to search. Available values are these: -180; -90; 0; 90; 180. The default value is 0. Identifier is used to set extractor base angle to support mobile devices with orientation sensor.
NLEP_FAVOR_LARGEST_FACE [123]	Identifier of type N_TYPE_BOOL [576]. If set to true the largest face found in an image will be used for facial feature extraction instead of the one with the highest face confidence threshold score.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLEP_GENDER_CONFIDENCE_THRESHOLD [124]	This threshold checks the gender of the captured face. The higher the value of this parameter, the more strictly the gender of the face is checked. The value of this parameter can be in range [0..100].
NLEP_GLASSES_CONFIDENCE_THRESHOLD [124]	Glasses on a face absence/presence confidence threshold.
NLEP_LIVENESS_THRESHOLD [123]	Identifier of type N_TYPE_DOUBLE [576]. This threshold checks if faces are extracted from a live image stream. The higher the value of this parameter, the more strictly the stream is checked if the face in it is real (for example if it's not forged by showing a photo of a person in front a camera). The value of this parameter can be in range [0..100]. The default value is 50.
NLEP_MAX_IOD [124]	Identifier of type N_TYPE_INT [577] specifying maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000].
NLEP_MAX_RECORDS_PER_TEMPLATE [124]	Identifier of type N_TYPE_INT [577]. Sets the maximum number of records an extraction function can return in one NLTemplate [153]. The value of this parameter can be in range [1..16]. The default value is 4.
NLEP_MAX_ROLL_ANGLE_DEVIATION [124]	Defines maximum roll angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180]. Parameter type - NShort [1008].
NLEP_MAX_STREAM_DURATION_IN_FRAMES [124]	Parameter identifier specifying maximum duration in frames to be used for template extraction from stream.
NLEP_MAX_YAW_ANGLE_DEVIATION [125]	Identifier specifying the maximum yaw angle deviation from frontal face in degrees. The value of this parameter can be in range [0..90]. The default value is 15. Currently supports deviations up to 45 degrees.
NLEP_MIN_IOD [124]	Identifier of type N_TYPE_INT [577] specifying minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000].
NLEP_MOUTH_OPEN_CONFIDENCE_THRESHOLD [125]	Face with mouth open absence/presence confidence threshold.
NLEP_TEMPLATE_SIZE [125]	Defines template size of Face template (NLTEmplate). Parameter type - NUShort [1010].
NLEP_USE_LIVENESS_CHECK [125]	Identifier of type N_TYPE_BOOL [576]. If set to true liveness check is performed while doing facial feature extraction from an image stream.

Module

NLExtractor Unit [104]

7.1.1.5 NLMatcher Unit

Provides functionality to identify or verify face feature templates (NLRecord [134]).

Notes

This module is deprecated. Use NMatcher [171] instead.

Files

Name	Description
NLMatcher.h [133]	Header file for the NLMatcher. Provides functionality to identify or verify face feature templates(NLRecord [134]).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLMatcherParams.h [134]	Header file for the NLMatcher. Enumerates parameters used by the NMatcher [171].

Functions

	Name	Description
	NlmCreate [129]	Creates a NLMatcher.
	NlmIdentifyEnd [130]	Finalizes identification procedure.
	NlmIdentifyNext [130]	Verifies one given template with the one that is being identified.
	NlmIdentifyStart [131]	Initializes identification procedure.
	NlmVerify [131]	Verifies two face feature templates (NLRecords) and return their similarity score.
	HNLMatcher [132]	Handle to NLMatcher object.

Macros

Name	Description
NLMP_MATCHING_SPEED [133]	Identifier specifying matching speed parameter of type N_TYPE_INT [577]. Parameter value can be one of the NlmSpeed [132] enumeration members.
NLMP_MATCHING_THRESHOLD [133]	Template matching threshold (parameter type - NDouble [1005]).

Structs, Records, Enums

	Name	Description
	NlmSpeed_ [132]	Specifies matching speed used in NLMatcher.
	NlmSpeed [132]	Specifies matching speed used in NLMatcher.

7.1.1.5.1 Functions

7.1.1.5.1.1 NlmCreate Function

Creates a NLMatcher [128].

C++

```
NResult N_API NlmCreate(HNLMatcher * phMatcher);
```

Parameters

Parameters	Description
HNLMatcher * phMatcher	[out] Pointer to HNLMatcher [132] that receives handle to created NLMatcher [128] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*phMatcher* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

Created object must be deleted using NObjectFree [542] function.

Module

NLMatcher Unit [128]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.5.1.2 NlmIdentifyEnd Function

Finalizes identification procedure.

C++

```
NResult N_API NlmIdentifyEnd(HNLMatcher hMatcher);
```

Parameters

Parameters	Description
HNLMatcher hMatcher	[in] Handle to the NLMatcher [128] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher* is NULL [1075].
N_E_INVALID_OPERATION [430]	Identification is not started.

Module

NLMatcher Unit [128]

7.1.1.5.1.3 NlmIdentifyNext Function

Verifies one given template with the one that is being identified.

C++

```
NResult N_API NlmIdentifyNext(HNLMatcher hMatcher, const void * pTemplate, NSizeType
templateSize, NDouble * pScore);
```

Parameters

Parameters	Description
HNLMatcher hMatcher	[in] Handle to the NLMatcher [128] object.
const void * pTemplate	[in] Pointer to memory buffer containing face features template (NLRecord [134]). This template must be compressed.
NSizeType templateSize	[in] Size of given template.
NDouble * pScore	[out] Pointer to NDouble [1005] that receives similarity score.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher*, *pTemplate*, or *pScore* is NULL [1075] or *pTemplate* has wrong size.
N_E_INVALID_OPERATION [430]	Identification is not started.

Remarks

Value received via *pScore* is zero if similarity is less than matching threshold, i.e. two templates do not match (see NLMP_MATCHING_THRESHOLD [133] parameter and NlmSetParameter function), and is greater than or equal to

Copyright © 2009-2014 Neurotechnology. All rights reserved.

matching threshold otherwise.

Module

NLMatcher Unit [128]

7.1.1.5.1.4 NlmIdentifyStart Function

Initializes identification procedure.

C++

```
NResult N_API NlmIdentifyStart(HNLMatcher hMatcher, const void * pTemplate, NSizeType
templateSize);
```

Parameters

Parameters	Description
HNLMatcher hMatcher	[in] Handle to the NLMatcher [128] object.
const void * pTemplate	[in] Pointer to memory buffer containing face features template (NLRecord [134]). This template must be uncompressed.
NSizeType templateSize	[in] Size of given template.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher* or *pTemplate* is NULL [1075] or wrong size.
N_E_INVALID_OPERATION [430]	Identification is already started.

Remarks

NlmIdentify start accepts only uncompressed templates, which are matched against the compressed templates from the user database using NlmIdentifyNext [130].

Module

NLMatcher Unit [128]

7.1.1.5.1.5 NlmVerify Function

Verifies two face feature templates (NLRecords) and return their similarity score.

C++

```
NResult N_API NlmVerify(HNLMatcher hMatcher, const void * pTemplate1, NSizeType
template1Size, const void * pTemplate2, NSizeType template2Size, NDouble * pScore);
```

Parameters

Parameters	Description
HNLMatcher hMatcher	[in] Handle to the NLMatcher [128] object.
const void * pTemplate1	[out] Pointer to memory buffer containing first face features template (NLRecord [134]). This template must be uncompressed.
NSizeType template1Size	[in] Size of first face features template.
const void * pTemplate2	[out] Pointer to memory buffer containing second face features template (NLRecord [134]). This template must be compressed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSizeType template2Size	[in] Size of second face features template (NLRecord [134]).
NDouble * pScore	[out] Pointer to NDouble [1005] that receives similarity score.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher* or *pTemplate1* or *pTemplate2* is NULL [1075] or *pTemplate1* or *pTemplate2* is wrong size.
N_E_INVALID_OPERATION [430]	Identification is already started.

Remarks

Value received via *pScore* is zero if similarity is less than matching threshold, i.e. two face templates do not match (see NLMP_MATCHING_THRESHOLD [133] and NlmSetParameter function), and is greater than or equal to matching threshold otherwise.

Module

NLMatcher Unit [128]

7.1.1.5.1.6 HNLMatcher

Handle to NLMatcher [128] object.

Remarks

The following parameters can be passed to function with corresponding handle:

- NLMP_MATCHING_SPEED [133]
- NLMP_MATCHING_THRESHOLD [133].

Module

NLMatcher Unit [128]

7.1.1.5.2 Structs, Records, Enums

7.1.1.5.2.1 NlmSpeed Enumeration

Specifies matching speed used in NLMatcher [128].

C++

```
typedef enum NlmSpeed_ {
  nlmsLow = 0,
  nlmsHigh = 256
} NlmSpeed;
```

Members

Members	Description
nlmsLow = 0	Low matching speed (slower but more accurate).
nlmsHigh = 256	High matching speed (faster but less accurate).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NLMatcher Unit [128]

7.1.1.5.3 Macros

To get a parameter value, use NObjectGetParameterEx function. To set parameter value - NObjectSetParameterEx function.

The following values can be used for *parameterId* in these functions:

- NLMP_MATCHING_SPEED [133]
- NLMP_MATCHING_THRESHOLD [133].

Macros

Name	Description
NLMP_MATCHING_SPEED [133]	Identifier specifying matching speed parameter of type N_TYPE_INT [577]. Parameter value can be one of the NlmSpeed [132] enumeration members.
NLMP_MATCHING_THRESHOLD [133]	Template matching threshold (parameter type - NDouble [1005]).

Module

NLMatcher Unit [128]

7.1.1.5.3.1 NLMP_MATCHING_SPEED Macro

Identifier specifying matching speed parameter of type N_TYPE_INT [577]. Parameter value can be one of the NlmSpeed [132] enumeration members.

C++

```
#define NLMP_MATCHING_SPEED 10001
```

Notes

This parameter is obsolete. Use NMP_FACES_MATCHING_SPEED [192] instead.

7.1.1.5.3.2 NLMP_MATCHING_THRESHOLD Macro

Template matching threshold (parameter type - NDouble [1005]).

C++

```
#define NLMP_MATCHING_THRESHOLD 10000
```

Notes

This parameter is obsolete. Use NMP_FACES_MATCHING_THRESHOLD_NEW [193] instead.

7.1.1.5.4 Files

7.1.1.5.4.1 NLMatcher.h

Header file for the NLMatcher [128]. Provides functionality to identify or verify face feature templates(NLRecord [134]).

Functions

	Name	Description
	NlmCreate [129]	Creates a NLMatcher [128].
	NlmIdentifyEnd [130]	Finalizes identification procedure.
	NlmIdentifyNext [130]	Verifies one given template with the one that is being identified.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NlmIdentifyStart [131]	Initializes identification procedure.
	NlmVerify [131]	Verifies two face feature templates (NLRecords) and return their similarity score.

Module

NLMatcher Unit [128]

7.1.1.5.4.2 NLMatcherParams.h

Header file for the NLMatcher [128]. Enumerates parameters used by the NMatcher [171].

Enumerations

	Name	Description
	NlmSpeed_ [132]	Specifies matching speed used in NLMatcher [128].
	NlmSpeed [132]	Specifies matching speed used in NLMatcher [128].

Macros

Name	Description
NLMP_MATCHING_SPEED [133]	Identifier specifying matching speed parameter of type N_TYPE_INT [577]. Parameter value can be one of the NlmSpeed [132] enumeration members.
NLMP_MATCHING_THRESHOLD [133]	Template matching threshold (parameter type - NDouble [1005]).

Module

NLMatcher Unit [128]

7.1.1.6 NLMatchingDetails Unit

Provides functionality to retrieve face templates matching details.

Remarks

NLMatchingDetails inherit all functions from NXMatchingDetails and NMatchingDetailsBase.

Files

Name	Description
NLMatchingDetails.h [134]	Header file of NLMatchingDetails.

7.1.1.6.1 Files

7.1.1.6.1.1 NLMatchingDetails.h

Header file of NLMatchingDetails [134].

Module

NLMatchingDetails Unit [134]

7.1.1.7 NLRecord Unit

Provides functionality for packing, unpacking and editing Neurotechnology Face Records (NLRecords).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Files

Name	Description
NLRecord.h [142]	Header file for the NLRecord. Provides functionality for packing, unpacking and editing Neurotechnology Face Records (NLRecords).

Functions

	Name	Description
	NLRecordCheck [135]	Checks whether packed NLRecord is valid.
	NLRecordCreate [136]	Creates an empty NLRecord.
	NLRecordCreateFromMemory [136]	Unpacks NLRecord from the specified memory buffer.
	NLRecordGetCbeffProductType [137]	Retrieves the Cbeff product type of the NLRecord.
	NLRecordGetCbeffProductTypeMem [137]	Retrieves the Cbeff product type of the packed NLRecord.
	NLRecordGetQuality [138]	Retrieves the quality of the NLRecord.
	NLRecordGetQualityMem [138]	Retrieves the quality of the packed NLRecord.
	NLRecordSetCbeffProductType [139]	Sets the Cbeff product type.
	NLRecordGetSizeMem [139]	Retrieves the image size of the packed NLRecord.
	NLRecordSetQuality [140]	Sets the quality of the NLRecord.
	HNLRecord [140]	Handle to NERecord object.

Macros

Name	Description
NLRecordClone [140]	Creates a copy of the specified NLRecord.
NLRecordGetSize [141]	Calculates size of serialized NLRecord.
NLRecordSaveToMemory [141]	Serializes NLRecord into the specified memory buffer.

7.1.1.7.1 Functions

7.1.1.7.1.1 NLRecordCheck Function

Checks whether packed NLRecord [134] is valid.

C++

```
NResult N_API NLRecordCheck(const void * pBuffer, NSizeType bufferSize);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NLRecord [134].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NLRecord [134].

Returns

If the function succeeds, the return value is N_OK [433].

Module

NLRecord Unit [134]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.7.1.2 NLRecordCreate Function

Creates an empty NLRecord [134].

C++

```
NResult N_API NLRecordCreate(NUInt flags, HNLRecord * phRecord);
```

Parameters

Parameters	Description
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. This parameter is reserved, must be zero.
HNLRecord * phRecord	[out] Pointer to HNLRecord [140] that receives handle to created NLRecord [134] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	pHRecord is NULL [1075].
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

Created object must be deleted using NObjectFree [542] function.

Module

NLRecord Unit [134]

7.1.1.7.1.3 NLRecordCreateFromMemory Function

Unpacks NLRecord [134] from the specified memory buffer.

C++

```
NResult N_API NLRecordCreateFromMemory(const void * pBuffer, NSizeType bufferSize, NUInt
flags, NSizeType * pSize, HNLRecord * phRecord);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NLRecord [134].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NLRecord [134].
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
NSizeType * pSize	[out] Actual size of data read from pBuffer (bytes).
HNLRecord * phRecord	[out] Pointer to HNLRecord [140] that receives handle to newly created NLRecord [134] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT_NULL [427]	*pHRecord* or *buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer buffer points to is inconsistent with NLRecord [134] format.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

Created object must be deleted using NObjectFree [542] function.

Module

NLRecord Unit [134]

7.1.1.7.1.4 NLRecordGetCbeffProductType Function

Retrieves the Cbeff product type of the NLRecord [134].

C++

```
NResult N_API NLRecordGetCbeffProductType(HNLRecord hRecord, NUShort * pValue);
```

Parameters

Parameters	Description
HNLRecord hRecord	[in] Handle of NLRecord [134] to retrieve its Cbeff type.
NUShort * pValue	[out] Type value.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hRecord* or *pValue [870]* is NULL [1075].

Module

NLRecord Unit [134]

7.1.1.7.1.5 NLRecordGetCbeffProductTypeMem Function

Retrieves the Cbeff product type of the packed NLRecord [134].

C++

```
NResult N_API NLRecordGetCbeffProductTypeMem(const void * pBuffer, NSizeType bufferSize,
NUShort * pValue);
```

Parameters

Parameters	Description
const void * pBuffer	[out] Pointer to memory buffer that contains packed NLRecord [134].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NLRecord [134].
NUShort * pValue	[out] Pointer to NUShort [1010] that receives Cbeff product type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hRecord* is NULL [1075].

Module

NLRecord Unit [134]

7.1.1.7.1.6 NLRecordGetQuality Function

Retrieves the quality of the NLRecord [134].

C++

```
NResult N_API NLRecordGetQuality(HNLRecord hRecord, NByte * pValue);
```

Parameters

Parameters	Description
HNLRecord hRecord	[in] Handle to the NLRecord [134] object.
NByte * pValue	[out] Pointer to NByte [1005] that receives face quality.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hRecord* or *pValue [870]* is NULL [1075].

Module

NLRecord Unit [134]

7.1.1.7.1.7 NLRecordGetQualityMem Function

Retrieves the quality of the packed NLRecord [134].

C++

```
NResult N_API NLRecordGetQualityMem(const void * pBuffer, NSizeType bufferSize, NByte *
pValue);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NLRecord [134].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NLRecord [134].
NByte * pValue	[out] Pointer to NByte [1005] that receives face quality.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT_NULL [427]	*buffer* or *pValue [870]* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NLRecord [134] format.

Remarks

This function supports both NLRecord [134] version 1.0, 2.0 and 3.0 formats. Always returns 0 for version 1.0 format.

Module

NLRecord Unit [134]

7.1.1.7.1.8 NLRecordSetCbeffProductType Function

Sets the Cbeff product type.

C++

```
NResult N_API NLRecordSetCbeffProductType(HNLRecord hRecord, NUShort value);
```

Parameters

Parameters	Description
HNLRecord hRecord	[in] Handle to the NLRecord [134] object.
NUShort value	[in] Cbeff product type.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hRecord* is NULL [1075].

Module

NLRecord Unit [134]

7.1.1.7.1.9 NLRecordGetSizeMem Function

Retrieves the image size of the packed NLRecord [134].

C++

```
NResult N_API NLRecordGetSizeMem(const void * pBuffer, NSizeType bufferSize, NSizeType *
pValue);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NLRecord [134].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NLRecord [134].
NSizeType * pValue	[out] Pointer to NSizeType [1008] that receives size of face image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*buffer* or *pValue [870]* is NULL [1075].

Module

NLRecord Unit [134]

7.1.1.7.1.10 NLRecordSetQuality Function

Sets the quality of the NLRecord [134].

C++

```
NResult N_API NLRecordSetQuality(HNLRecord hRecord, NByte value);
```

Parameters

Parameters	Description
HNLRecord hRecord	[in] Handle to the NLRecord [134] object.
NByte value	[in] New face quality value.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hRecord* is NULL [1075].

Module

NLRecord Unit [134]

7.1.1.7.1.11 HNLRecord

Handle to NERecord object.

Module

NLRecord Unit [134]

7.1.1.7.2 Macros

7.1.1.7.2.1 NLRecordClone Macro

Creates a copy of the specified NLRecord [134].

C++

```
#define NLRecordClone(hRecord, phClonedRecord) NObjectClone(hRecord, (HNObject
*)phClonedRecord)
```

Parameters

Parameters	Description
hRecord	[in] Handle to the original NLRecord [134].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

phClonedRecord	[out] Pointer to a handle to the copied NLRecord [134].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NLRecord Unit [134]

7.1.1.7.2.2 NLRecordGetSize Macro

Calculates size of serialized NLRecord [134].

C++

```
#define NLRecordGetSize(hRecord, flags, pValue) NObjectGetSize(hRecord, flags, pValue)
```

Parameters

Parameters	Description
hRecord	[in] Handle to the NLRecord [134] object.
flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
pSize	[out] Pointer to NSizeType [1008] that receives size of packed NLRecord [134].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hRecord* or *pSize* is NULL [1075].

Remarks

For the list of flags that are supported see NLRecordSaveToMemory [141] function.

Module

NLRecord Unit [134]

7.1.1.7.2.3 NLRecordSaveToMemory Macro

Serializes NLRecord [134] into the specified memory buffer.

C++

```
#define NLRecordSaveToMemory(hRecord, pBuffer, bufferSize, flags, pSize)
NObjectSaveToMemoryDst(hRecord, pBuffer, bufferSize, flags, pSize)
```

Parameters

Parameters	Description
hRecord	[in] Handle to the NLRecord [134] object.
pBuffer	[out] Pointer to memory buffer to store packed NLRecord [134]. If NULL [1075] is passed as a parameter an error occurs. To get buffer size use NLRecordGetSize [141] function.
bufferSize	[in] Size of memory buffer to store packed NLRecord [134].
flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. Should be zero.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

pSize	[out] Pointer to NSizeType [1008] that receives size of packed NLRecord [134].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT [427]	*buffer* is not NULL [1075] and *bufferSize* is less than size required to store packed NLRecord [134].
N_E_ARGUMENT_NULL [427]	*hRecord* or *pSize* is NULL [1075]. - or - *buffer* is NULL [1075] and *bufferSize* is not zero.

Remarks

If *buffer* is NULL [1075] and *bufferSize* is zero the function only calculates the size of the buffer needed and has the same effect as NLRecordGetSize [141] function.

If *buffer* is not NULL [1075], *bufferSize* must not be less than value calculated with NLRecordGetSize [141] function.

Module

NLRecord Unit [134]

7.1.1.7.3 Files

7.1.1.7.3.1 NLRecord.h

Header file for the NLRecord [134]. Provides functionality for packing, unpacking and editing Neurotechnology Face Records (NLRecords).

Functions

	Name	Description
	NLRecordCheck [135]	Checks whether packed NLRecord [134] is valid.
	NLRecordCreate [136]	Creates an empty NLRecord [134].
	NLRecordCreateFromMemory [136]	Unpacks NLRecord [134] from the specified memory buffer.
	NLRecordGetCbeffProductType [137]	Retrieves the Cbeff product type of the NLRecord [134].
	NLRecordGetCbeffProductTypeMem [137]	Retrieves the Cbeff product type of the packed NLRecord [134].
	NLRecordGetQuality [138]	Retrieves the quality of the NLRecord [134].
	NLRecordGetQualityMem [138]	Retrieves the quality of the packed NLRecord [134].
	NLRecordGetSizeMem [139]	Retrieves the image size of the packed NLRecord [134].
	NLRecordSetCbeffProductType [139]	Sets the Cbeff product type.
	NLRecordSetQuality [140]	Sets the quality of the NLRecord [134].

Macros

Name	Description
NLRecordClone [140]	Creates a copy of the specified NLRecord [134].
NLRecordGetSize [141]	Calculates size of serialized NLRecord [134].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLRecordSaveToMemory [141]	Serializes NLRecord [134] into the specified memory buffer.

Module

NLRecord Unit [134]

7.1.1.8 NlsMatcher Unit

Provides functionality for comparing Neurotechnology Face Templates (NLTemplates), encapsulated in Neurotechnology Face Matcher (NlsMatcher) object.

Notes

This module is deprecated. Use NMatcher [171] instead.

Files

Name	Description
NlsMatcher.h [151]	Header file for the NlsMatcher.
NlsMatcherParams.h [152]	Header file for the NlsMatcher.
NlsmMatchDetails.h [152]	**Summary** Header file for the NlsMatcher. Provides functionality for comparing Neurotechnology Face Templates (NLTemplates), encapsulated in Neurotechnology Face Matcher (NlsMatcher) object.

Functions

	Name	Description
	NlsmCreate [144]	Creates a NfsMatcher.
	NlsmIdentifyEnd [144]	Ends identification using the specified NfsMatcher.
	NlsmIdentifyNext [145]	Compares the specified packed NFTemplate with the one identification was started with using the specified NfsMatcher.
	NlsmIdentifyStart [145]	Starts identification with the specified packed NFTemplate using the specified NfsMatcher.
	NlsmMatchDetailsDeserialize [146]	Deserializes NlsmMatchDetails [150].
	NlsmMatchDetailsDeserializeFromStream [147]	Deserializes NlsmMatchDetails [150] contained by the specified stream.
	NlsmMatchDetailsFree [147]	Releases memory resources used by the NlsmMatchDetails [150].
	NlsmMatchDetailsSerialize [147]	Serializes NlsmMatchDetails [150].
	NlsmMatchDetailsSerializeToStream [148]	Serializes NlsmMatchDetails [150] object to memory stream.
	NlsmVerify [148]	Compares two packed NFTemplate using the specified NfsMatcher.
	HNlsMatcher [149]	Handle to NlsMatcher object.

Macros

Name	Description
NLSM_PART_NLM [151]	Identifier specifying that retrieving or setting parameter for NLMatcher [128] contained in NlsMatcher.
NLSM_PART_NONE [151]	Identifier specifying that retrieving or setting parameter for NlsMatcher.
NLSMP_MATCHING_THRESHOLD [151]	Identifier specifying matching threshold (biggest allowed FAR) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Structs, Records, Enums

	Name	Description
	NlsmFaceMatchDetails_ [149]	Structure representing details of matching performed with NlsMatcher for one face in the first matched NLemplate.
	NlsmMatchDetails_ [150]	Structure containing the matching details performed using NlsMatcher.
	NlsmFaceMatchDetails [149]	Structure representing details of matching performed with NlsMatcher for one face in the first matched NLemplate.
	NlsmMatchDetails [150]	Structure containing the matching details performed using NlsMatcher.

7.1.1.8.1 Functions

7.1.1.8.1.1 NlsmCreate Function

Creates a NfsMatcher.

C++

```
NResult N_API NlsmCreate(HNlsMatcher * phMatcher);
```

Parameters

Parameters	Description
HNlsMatcher * phMatcher	[out] Pointer to HNlsMatcher [149] that receives handle to created NlsMatcher [143] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*phMatcher* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

Created object must be deleted using NObjectFree [542] function.

Module

NlsMatcher Unit [143]

7.1.1.8.1.2 NlsmIdentifyEnd Function

Ends identification using the specified NfsMatcher.

C++

```
NResult N_API NlsmIdentifyEnd(HNlsMatcher hMatcher);
```

Parameters

Parameters	Description
HNlsMatcher hMatcher	[in] Handle to the NlsMatcher [143] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher* is NULL [1075].
N_E_INVALID_OPERATION [430]	Identification is not started.

Module

NlsMatcher Unit [143]

7.1.1.8.1.3 NlsmIdentifyNext Function

Compares the specified packed NFTemplate with the one identification was started with using the specified NfsMatcher.

C++

```
NResult N_API NlsmIdentifyNext(HNlsMatcher hMatcher, const void * templ, NSizeType
templSize, NlsmMatchDetails * pMatchDetails, NDouble * pScore);
```

Parameters

Parameters	Description
HNlsMatcher hMatcher	[in] Handle to the NlsMatcher [143] object.
const void * templ	[in] Pointer to memory buffer containing packed NLTemplate [153].
NSizeType templSize	[in] Size of packed NLTemplate [153].
NlsmMatchDetails * pMatchDetails	[out] Pointer to NlsmMatchDetails [150] that is filled with details of the matching. Can be NULL [1075].
NDouble * pScore	[out] Pointer to NInt [1006] that receives similarity score.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher*, *templ*, or *pScore* is NULL [1075].
N_E_END_OF_STREAM [429]	*templSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *templ* points to is inconsistent with NLTemplate [153] format.
N_E_INVALID_OPERATION [430]	Identification is not started.

Remarks

Value received via *pScore* is zero if similarity is less than matching threshold, i.e. two NLTemplates do not match (see NLSMP_MATCHING_THRESHOLD [151] and NObjectSetParameter [564] function), and is greater than or equal to matching threshold otherwise.

If *pMatchDetails* is not NULL [1075] it should be a pointer obtained using NlsmIdentifyStart [145] function.

Module

NlsMatcher Unit [143]

7.1.1.8.1.4 NlsmIdentifyStart Function

Starts identification with the specified packed NFTemplate using the specified NfsMatcher.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NlsmIdentifyStart(HNlsMatcher hMatcher, const void * templ, NSizeType
templSize, NlsmMatchDetails * * ppMatchDetails);
```

Parameters

Parameters	Description
HNlsMatcher hMatcher	[in] Handle to the NlsMatcher [143] object.
const void * templ	[in] Pointer to memory buffer containing packed NLTemplate [153].
NSizeType templSize	[in] Size of packed NLTemplate [153].
NlsmMatchDetails * * ppMatchDetails	[out] Pointer to pointer to NlsmMatchDetails [150] that receives pointer to created match details. Can be NULL [1075].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher* or *templ* is NULL [1075].
N_E_END_OF_STREAM [429]	*templSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *templ* points to is inconsistent with NLTemplate [153] format.
N_E_INVALID_OPERATION [430]	Identification is already started.

Remarks

If *ppMatchDetails* is not NULL [1075], the received pointer should be sequent passed to NlsmIdentifyNext [145] function to obtain details of the matching.

Finally match details should be deleted using NlsmMatchDetailsFree [147] function.

Module

NlsMatcher Unit [143]

7.1.1.8.1.5 NlsmMatchDetailsDeserialize Function

Deserializes NlsmMatchDetails [150].

C++

```
NResult N_API NlsmMatchDetailsDeserialize(const void * pBuffer, NSizeType bufferSize,
NlsmMatchDetails * * ppMatchDetails);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to void * that receives pointer to serialized NlsmMatchDetails [150].
NSizeType bufferSize	[in] Size of allocated memory buffer
NlsmMatchDetails * * ppMatchDetails	[out] Pointer to pointer to NlsmMatchDetails [150] that returns pointer to created match details. Can be NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

If the function succeeds, the return value is N_OK [433].

Module

NlsMatcher Unit [143]

7.1.1.8.1.6 NlsmMatchDetailsDeserializeFromStream Function

Deserializes NlsmMatchDetails [150] contained by the specified stream.

C++

```
NResult N_API NlsmMatchDetailsDeserializeFromStream(HNStream hStream, NlsmMatchDetails * *
ppMatchDetails);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to memory stream that receives pointer to serialized NlsmMatchDetails [150].
NlsmMatchDetails * * ppMatchDetails	[out] Pointer to pointer to NlsmMatchDetails [150] that returns pointer to created match details. Can be NULL [1075].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hStream* or *ppMatchDetails* is NULL [1075].

Module

NlsMatcher Unit [143]

7.1.1.8.1.7 NlsmMatchDetailsFree Function

Releases memory resources used by the NlsmMatchDetails [150].

C++

```
void N_API NlsmMatchDetailsFree(NlsmMatchDetails * pMatchDetails);
```

Parameters

Parameters	Description
NlsmMatchDetails * pMatchDetails	[out] Pointer to NlsmMatchDetails [150] to delete.

Remarks

If *pMatchDetails* is NULL [1075], does nothing.

Module

NlsMatcher Unit [143]

7.1.1.8.1.8 NlsmMatchDetailsSerialize Function

Serializes NlsmMatchDetails [150].

C++

```
NResult N_API NlsmMatchDetailsSerialize(NlsmMatchDetails * pMatchDetails, void * *
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
ppBuffer, NSizeType * pBufferSize);
```

Parameters

Parameters	Description
NlsmMatchDetails * pMatchDetails	[out] Pointer to NlsmMatchDetails [150] to serialize.
void * * ppBuffer	[out] Pointer to void * that receives pointer to allocated memory buffer.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer.

Returns

If the function succeeds, the return value is N_OK [433].

Module

NlsMatcher Unit [143]

7.1.1.8.1.9 NlsmMatchDetailsSerializeToStream Function

Serializes NlsmMatchDetails [150] object to memory stream.

C++

```
NResult N_API NlsmMatchDetailsSerializeToStream(NlsmMatchDetails * pMatchDetails, HNStream
hStream);
```

Parameters

Parameters	Description
NlsmMatchDetails * pMatchDetails	[out] Pointer to NlsmMatchDetails [150] to serialize.
HNStream hStream	[in] Handle to memory stream.

Returns

If the function succeeds, the return value is N_OK [433].

Module

NlsMatcher Unit [143]

7.1.1.8.1.10 NlsmVerify Function

Compares two packed NFTemplate using the specified NfsMatcher.

C++

```
NResult N_API NlsmVerify(HNlsMatcher hMatcher, const void * template1, NSizeType
template1Size, const void * template2, NSizeType template2Size, NlsmMatchDetails * *
ppMatchDetails, NDouble * pScore);
```

Parameters

Parameters	Description
HNlsMatcher hMatcher	[in] Handle to the NlsMatcher [143] object.
const void * template1	[in] Pointer to memory buffer containing first packed NLTemplate [153].
NSizeType template1Size	[in] Size of first packed NLTemplate [153].
const void * template2	[in] Pointer to memory buffer containing second packed NLTemplate [153].
NSizeType template2Size	[in] Size of second packed NLTemplate [153].
NlsmMatchDetails * * ppMatchDetails	[out] Pointer to pointer to NlsmMatchDetails [150] that receives pointer to created match details. Can be NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDouble * pScore	[out] Pointer to NDouble [1005] that receives similarity score.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher*, *template1*, *template2*, or *pScore* is NULL [1075].
N_E_END_OF_STREAM [429]	*template1Size* or *template2Size* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *template1* or *template2* points to is inconsistent with NLTemplate [153] format.
N_E_INVALID_OPERATION [430]	Identification is started.

Remarks

Value received via *pScore* is zero if similarity is less than matching threshold, i.e. two NLTemplates do not match (see NLSMP_MATCHING_THRESHOLD [151] and NObjectSetParameter [564] function), and is greater than or equal to matching threshold otherwise.

If *ppMatchDetails* is not NULL [1075], the received pointer should be examined to obtain details of the matching.

Finally match details should be deleted using NlsmMatchDetailsFree [147] function.

Module

NlsMatcher Unit [143]

7.1.1.8.1.11 HNlsMatcher

Handle to NlsMatcher [143] object.

Remarks

The following parameters can be passed to function with corresponding handle:

- NLSM_PART_NLM [151]
- NLSM_PART_NONE [151].
- NLSMP_MATCHING_THRESHOLD [151].

Module

NlsMatcher Unit [143]

7.1.1.8.2 Structs, Records, Enums

7.1.1.8.2.1 NlsmFaceMatchDetails Structure

Structure representing details of matching performed with NlsMatcher [143] for one face in the first matched NLemplate.

C++

```
typedef struct NlsmFaceMatchDetails_ {
  NDouble Score;
  NInt MatchedIndex;
} NlsmFaceMatchDetails;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Members

Members	Description
NDouble Score;	Similarity score of the two matched faces.
NInt MatchedIndex;	Index of matched face in the second matcher NLTemplate [153].

Module

NlsMatcher Unit [143]

7.1.1.8.2.2 NlsmMatchDetails Structure

Structure containing the matching details performed using NlsMatcher [143].

C++

```
typedef struct NlsmMatchDetails_ {
  NDouble Score;
  NInt MatchedIndex;
  NInt FaceCount;
  NlsmFaceMatchDetails * Faces;
} NlsmMatchDetails;
```

Members

Members	Description
NDouble Score;	Similarity score of the two matched NLTemplates.
NInt FaceCount;	Number of faces in the first matched NLTemplate [153].
NlsmFaceMatchDetails * Faces;	Pointer to array of NlsmFaceMatchDetails [149] containing details of matching performed for each face in the first matched NLTemplate [153].

Module

NlsMatcher Unit [143]

7.1.1.8.3 Macros

To get a parameter value, use NObjectSetParameterWithPartEx [557] function. To set parameter value - NObjectGetParameterWithPartEx [546].

Parameter *partId* passed to these functions can be one of the following:

- NLSM_PART_NLM [151]
- NLSM_PART_NONE [151]

If *partId* is NLSM_PART_NLM [151], *parameterId* must be any of NLMatcher macros [133] except for NLMP_MATCHING_THRESHOLD [133].

If *partId* is NLSM_PART_NONE [151], the following values can be used for *parameterId*:

- NLSMP_MATCHING_THRESHOLD [151]

Macros

Name	Description
NLSM_PART_NLM [151]	Identifier specifying that retrieving or setting parameter for NLMatcher [128] contained in NlsMatcher [143].
NLSM_PART_NONE [151]	Identifier specifying that retrieving or setting parameter for NlsMatcher [143].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLSMP_MATCHING_THRESHOLD [151]	Identifier specifying matching threshold (biggest allowed FAR) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

Module

NlsMatcher Unit [143]

7.1.1.8.3.1 NLSM_PART_NLM Macro

Identifier specifying that retrieving or setting parameter for NLMatcher [128] contained in NlsMatcher [143].

C++

```
#define NLSM_PART_NLM 1
```

Notes

This macro is deprecated, use NM_PART_FACES [191] instead.

See Also

NObjectGetParameter [562]

7.1.1.8.3.2 NLSM_PART_NONE Macro

Identifier specifying that retrieving or setting parameter for NlsMatcher [143].

C++

```
#define NLSM_PART_NONE 0
```

Notes

This macro is deprecated, use NM_PART_NONE [192] instead.

See Also

NObjectGetParameter [562]

NObjectSetParameter [564]

7.1.1.8.3.3 NLSMP_MATCHING_THRESHOLD Macro

Identifier specifying matching threshold (biggest allowed FAR) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

C++

```
#define NLSMP_MATCHING_THRESHOLD 100
```

Notes

This macro is deprecated, use NMP_FACES_MATCHING_THRESHOLD [192] instead.

7.1.1.8.4 Files

7.1.1.8.4.1 NlsMatcher.h

Header file for the NlsMatcher [143].

Functions

	Name	Description
	NlsmCreate [144]	Creates a NfsMatcher.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NlsmIdentifyEnd [144]	Ends identification using the specified NfsMatcher.
	NlsmIdentifyNext [145]	Compares the specified packed NFTemplate with the one identification was started with using the specified NfsMatcher.
	NlsmIdentifyStart [145]	Starts identification with the specified packed NFTemplate using the specified NfsMatcher.
	NlsmVerify [148]	Compares two packed NFTemplate using the specified NfsMatcher.

Module

NlsMatcher Unit [143]

7.1.1.8.4.2 NlsMatcherParams.h

Header file for the NlsMatcher [143].

Macros

Name	Description
NLSM_PART_NLM [151]	Identifier specifying that retrieving or setting parameter for NLMatcher [128] contained in NlsMatcher [143].
NLSM_PART_NONE [151]	Identifier specifying that retrieving or setting parameter for NlsMatcher [143].
NLSMP_MATCHING_THRESHOLD [151]	Identifier specifying matching threshold (biggest allowed FAR) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

Module

NlsMatcher Unit [143]

7.1.1.8.4.3 NlsmMatchDetails.h

Summary

Header file for the NlsMatcher [143]. Provides functionality for comparing Neurotechnology Face Templates (NLTemplates), encapsulated in Neurotechnology Face Matcher (NlsMatcher [143]) object.

Functions

	Name	Description
	NlsmMatchDetailsDeserialize [146]	Deserializes NlsmMatchDetails [150].
	NlsmMatchDetailsDeserializeFromStream [147]	Deserializes NlsmMatchDetails [150] contained by the specified stream.
	NlsmMatchDetailsFree [147]	Releases memory resources used by the NlsmMatchDetails [150].
	NlsmMatchDetailsSerialize [147]	Serializes NlsmMatchDetails [150].
	NlsmMatchDetailsSerializeToStream [148]	Serializes NlsmMatchDetails [150] object to memory stream.

Module

NlsMatcher Unit [143]

Structures

	Name	Description
	NlsmFaceMatchDetails_ [149]	Structure representing details of matching performed with NlsMatcher [143] for one face in the first matched NLemplate.
	NlsmMatchDetails_ [150]	Structure containing the matching details performed using NlsMatcher [143].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NlsmFaceMatchDetails [149]	Structure representing details of matching performed with NlsMatcher [143] for one face in the first matched NLemplate.
	NlsmMatchDetails [150]	Structure containing the matching details performed using NlsMatcher [143].

7.1.1.9 NLTemplate Unit

Provides functionality for packing, unpacking and editing Neurotechnology Faces [45] Templates (NLTemplates).

Files

Name	Description
NLTemplate.h [170]	Header file for the NLTemplate. Provides functionality for packing, unpacking and editing Neurotechnology Faces [45] Templates (NLTemplates).

Functions

	Name	Description
	NLTemplateAddRecord [154]	Adds an empty face record to the NLTemplate.
	NLTemplateAddRecordFromMemoryEx [154]	Unpacks a face record from the specified memory buffer and adds it to the NLTemplate.
	NLTemplateAddRecordCopyEx [155]	Adds a copy of the face record to the NLTemplate.
	NLTemplateCalculateSize [156]	Calculates the size of a packed NLTemplate containing the specified number of face records of specified size.
	NLTemplateAddRecordEx [156]	Adds a face record to the NLTemplate.
	NLTemplateCheck [157]	Checks if format of the packed NLTemplate is correct.
	NLTemplateClearRecords [157]	Removes all face records from the NLTemplate.
	NLTemplateCreateFromMemory [158]	Unpacks a NLTemplate from the specified memory buffer.
	NLTemplateCreateEx [159]	Creates an empty NLTemplate.
	NLTemplateGetRecord [159]	Retrieves the face record at the specified index of the NLTemplate.
	NLTemplateGetRecordCapacity [160]	Retrieves the number of face records that the NLTemplate can contain..
	NLTemplateGetRecordCount [160]	Retrieves the number of face records in the NLTemplate.
	NLTemplateGetRecordCountMem [161]	Retrieves the number of face records in the packed NLTemplate.
	NLTemplatePack [161]	Packs packed face records as NLTemplate into the specified memory buffer.
	NLTemplateGetRecordEx [162]	Retrieves the face record at the specified index of the NLTemplate.
	NLTemplateRemoveRecord [163]	Removes the face record at the specified index of the NLTemplate.
	NLTemplateGetSizeMem [163]	Retrieves the image size of the packed NLTemplate.
	NLTemplateInsertRecord [164]	Inserts the specified NLRecord [134] to the specified NLTemplate at the specified index.
	NLTemplateSetRecordCapacity [164]	Sets the number of face records that the NLTemplate can contain.
	HNLTemplate [165]	Handle to the NLTemplate object.
	NLTemplateSetRecord [165]	Sets the value of the specified NLRecord [134] to the specified value.
	NLTemplateUnpack [165]	Unpacks packed face records from the packed NLTemplate.

Macros

Name	Description
NLT_MAX_RECORD_COUNT [166]	The maximum number of face records NLTemplate can contain.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLT_PROCESS_FIRST_RECORD_ONLY [166]	The flag indicating whether only the first face record should be unpacked or packed while unpacking or packing NLTemplate.
NLTemplateAddRecordCopy [166]	Adds a copy of the face record to the NLTemplate.
NLTemplateAddRecordFromMemory [167]	Unpacks a face record from the specified memory buffer and adds it to the NLTemplate.
NLTemplateClone [168]	Creates a copy of the specified NLTemplate.
NLTemplateCreate [168]	Creates a new empty NLTemplate.
NLTemplateGetSize [168]	Calculates packed size of the NLTemplate.
NLTemplateSaveToMemory [169]	Packs the NLTemplate into the specified memory buffer.

7.1.1.9.1 Functions

7.1.1.9.1.1 NLTemplateAddRecord Function

Adds an empty face record to the NLTemplate [153].

C++

```
NResult N_API NLTemplateAddRecord(HNLTemplate hTemplate, NUInt flags, HNLRecord * phRecord);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] Handle to the NLTemplate [153] object.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
HNLRecord * phRecord	[out] Pointer to HNLRecord [140] that receives handle to created NLRecord [134] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHRecord* is NULL [1075].
N_E_INVALID_OPERATION [430]	Number of face records in NLTemplate [153] (see NLTemplateGetRecordCount [160]) is equal to NLT_MAX_RECORD_COUNT [166].
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

For the list of flags that are supported see NLRecordCreate [136] function.

Module

NLTemplate Unit [153]

7.1.1.9.1.2 NLTemplateAddRecordFromMemoryEx Function

Unpacks a face record from the specified memory buffer and adds it to the NLTemplate [153].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NLTemplateAddRecordFromMemoryEx(HNLTemplate hTemplate, const void * pBuffer,
NSizeType bufferSize, NUInt flags, NSizeType * pSize, HNLRecord * phRecord);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] Handle to the NLTemplate [153] object.
const void * pBuffer	[in] Pointer to memory buffer that contains packed NLRecord [134].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NLRecord [134].
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
NSizeType * pSize	[out] Actual size of data read from pBuffer (bytes).
HNLRecord * phRecord	[out] Pointer to HNLRecord [140] that receives handle to created NLRecord [134] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHRecord*, or *buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NLRecord [134] format.
N_E_INVALID_OPERATION [430]	Number of face records in NLTemplate [153] (see NLTemplateGetRecordCount [160]) is equal to NLT_MAX_RECORD_COUNT [166].
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

For the list of flags that are supported see NLRecordCreateFromMemory [136] function.

Module

NLTemplate Unit [153]

7.1.1.9.1.3 NLTemplateAddRecordCopyEx Function

Adds a copy of the face record to the NLTemplate [153].

C++

```
NResult N_API NLTemplateAddRecordCopyEx(HNLTemplate hTemplate, HNLRecord hSrcRecord, NUInt
flags, HNLRecord * phRecord);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] A handle to the NLTemplate [153].
HNLRecord hSrcRecord	[in] A handle to the NLRecord [134] to be copied.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
HNLRecord * phRecord	[out] A pointer to a handle to the copy of NLRecord [134].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For the list of flags that are supported see NLRecordCreate [136] function.

Module

NLTemplate Unit [153]

7.1.1.9.1.4 NLTemplateCalculateSize Function

Calculates the size of a packed NLTemplate [153] containing the specified number of face records of specified size.

C++

```
NResult N_API NLTemplateCalculateSize(NInt recordCount, NSizeType * arRecordSizes,
NSizeType * pSize);
```

Parameters

Parameters	Description
NInt recordCount	[in] The number of face records.
NSizeType * arRecordSizes	[out] Pointer to array of NSizeTypethat contains packed sizes of each face record.
NSizeType * pSize	[out] Pointer to NSizeType [1008] that receives calculated size of packed NLTemplate [153].

Returns

If the function succeeds, the return value is N_OK [433]. If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT [427]	Any entry of the array *arRecordSizes* points to is less than minimal face record size.
N_E_ARGUMENT_NULL [427]	*arRecordSizes* or *pSize* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	*recordCount* is less than zero or greater than NLT_MAX_RECORD_COUNT [166].

Remarks

This is a low-level function and can be changed in future version of the library.

Use NLRecordGetMaxSize function to calculate packed size of an individual face record.

Module

NLTemplate Unit [153]

7.1.1.9.1.5 NLTemplateAddRecordEx Function

Adds a face record to the NLTemplate [153].

C++

```
NResult N_API NLTemplateAddRecordEx(HNLTemplate hTemplate, HNLRecord hValue, NInt * pIndex);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] Handle to the NLTemplate [153] object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNLRecord hValue	[in] A handle to the face record to be added.
NInt * pIndex	[out] A pointer to the index of the new record.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NLTemplate Unit [153]

7.1.1.9.1.6 NLTemplateCheck Function

Checks if format of the packed NLTemplate [153] is correct.

C++

```
NResult N_API NLTemplateCheck(const void * pBuffer, NSizeType bufferSize);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NLTemplate [153].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NLTemplate [153].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NLTemplate [153] format.

Module

NLTemplate Unit [153]

7.1.1.9.1.7 NLTemplateClearRecords Function

Removes all face records from the NLTemplate [153].

C++

```
NResult N_API NLTemplateClearRecords(HNLTemplate hTemplate);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] Handle to the NLTemplate [153] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* is NULL [1075].

Module

NLTemplate Unit [153]

7.1.1.9.1.8 NLTemplateCreateFromMemory Function

Unpacks a NLTemplate [153] from the specified memory buffer.

C++

```
NResult N_API NLTemplateCreateFromMemory(const void * pBuffer, NSizeType bufferSize, NUInt flags, NSizeType * pSize, HNLTemplate * phTemplate);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NLTemplate [153].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NLTemplate [153].
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
NSizeType * pSize	[out] Actual size of data read from pBuffer (bytes).
HNLTemplate * phTemplate	[out] Pointer to HNLTemplate [165] that receives handle to newly created NLTemplate [153] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*phTemplate* or *buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NLTemplate [153] format.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

The following flags are supported:

- NLT_PROCESS_FIRST_RECORD_ONLY [166].

Flags supported by NLRecordCreateFromMemory [136] function are applied to each face record contained in the NLTemplate [153].

Created object must be deleted using NObjectFree [542] function.

Module

NLTemplate Unit [153]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.9.1.9 NLTemplateCreateEx Function

Creates an empty NLTemplate [153].

C++

```
NResult N_API NLTemplateCreateEx(NUInt flags, HNLTemplate * phTemplate);
```

Parameters

Parameters	Description
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. This parameter is reserved, must be zero.
HNLTemplate * phTemplate	[out] Pointer to HNLTemplate [165] that receives handle to created NLTemplate [153] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*phTemplate* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

Created object must be deleted using NObjectFree [542] function.

Module

NLTemplate Unit [153]

7.1.1.9.1.10 NLTemplateGetRecord Function

Retrieves the face record at the specified index of the NLTemplate [153].

C++

```
NResult N_API NLTemplateGetRecord(HNLTemplate hTemplate, NInt index, HNLRecord * phValue);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] Handle to the NLTemplate [153] object.
NInt index	[in] Index of face record to retrieve.
HNLRecord * phValue	[out] Pointer to HNLRecord [140] that receives handle to NLRecord [134] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *phValue* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	*index* is less than zero or greater than or equal to face record count obtained using NLTemplateGetRecordCount [160] function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NLTemplate Unit [153]

7.1.1.9.1.11 NLTemplateGetRecordCapacity Function

Retrieves the number of face records that the NLTemplate [153] can contain..

C++

```
NResult N_API NLTemplateGetRecordCapacity(HNLTemplate hTemplate, NInt * pValue);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] Handle to the NLTemplate [153] object.
NInt * pValue	[out] Pointer to NInt [1006] that receives number of face records NLTemplate [153] can contain.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pValue [870]* is NULL [1075].

Remarks

Face record capacity is the number of face records that the NLTemplate [153] can store. Face record count (see NLTemplateGetRecordCount [160] function) is the number of face records that are actually in the NLTemplate [153].

Capacity is always greater than or equal to count. If count exceeds capacity while adding face records the capacity is automatically increased by reallocating the internal array before copying the old elements and adding the new elements.

Module

NLTemplate Unit [153]

7.1.1.9.1.12 NLTemplateGetRecordCount Function

Retrieves the number of face records in the NLTemplate [153].

C++

```
NResult N_API NLTemplateGetRecordCount(HNLTemplate hTemplate, NInt * pValue);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] Handle to NLTemplate [153] object.
NInt * pValue	[out] Pointer to NInt [1006] that receives number of face records.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pValue [870]* is NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

Face record capacity (see NLTemplateGetRecordCapacity [160] and NLTemplateSetRecordCapacity [164] functions) is the number of face records that the NLTemplate [153] can store. Face record count is the number of face records that are actually in the NLTemplate [153].

Capacity is always greater than or equal to count. If count exceeds capacity while adding face records the capacity is automatically increased by reallocating the internal array before copying the old elements and adding the new elements.

Module

NLTemplate Unit [153]

7.1.1.9.1.13 NLTemplateGetRecordCountMem Function

Retrieves the number of face records in the packed NLTemplate [153].

C++

```
NResult N_API NLTemplateGetRecordCountMem(const void * pBuffer, NSizeType bufferSize, NInt
* pValue);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NLTemplate [153].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NLTemplate [153].
NInt * pValue	[out] Pointer to NInt [1006] that receives number of face records.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*buffer* or *pValue [870]* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NLTemplate [153] format.

Module

NLTemplate Unit [153]

7.1.1.9.1.14 NLTemplatePack Function

Packs packed face records as NLTemplate [153] into the specified memory buffer.

C++

```
NResult N_API NLTemplatePack(NInt recordCount, const void * * arpRecords, NSizeType *
arRecordSizes, void * pBuffer, NSizeType bufferSize, NSizeType * pSize);
```

Parameters

Parameters	Description
NInt recordCount	[in] Number of records to be packed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const void * * arpRecords	[in] Pointer to array of void * that contains packed face records.
NSizeType * arRecordSizes	[out] Pointer to array of NSizeType [1008] that contains packed sizes of each face record.
void * pBuffer	[out] Pointer to memory buffer to store packed NLTemplate [153]. If NULL [1075] is passed as a parameter an error occurs. To get buffer size use NLTemplateCalculateSize [156] or NLTemplateGetSize [168] functions.
NSizeType bufferSize	[in] Size of memory buffer to store packed NTemplate [219].
NSizeType * pSize	[out] Pointer to NSizeType [1008] that receives size of packed NTemplate [219].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*arRecordSizes*, *recordCount*, *pBuffer* or *pSize* is NULL [1075].
N_E_ARGUMENT [427]	*bufferSize* is less than size required to store packed NLTemplate [153]. - or - Any entry of the array *arRecordSizes* points to is less than minimal face record size. - or - Any entry of the array *arpRecords* points to is NULL [1075]. - or - Any entry of the array *arRecordSizes* points to is not equal to size stored in memory buffer according entry of the array *arpRecords* points to, points to.
N_E_ARGUMENT_OUT_OF_RANGE [428]	*recordCount* is less than zero or greater than NLT_MAX_RECORD_COUNT [166].
N_E_FORMAT [429]	Data in memory buffer any entry of the array *arpRecords* points to is inconsistent with NLRecord [134] format.

Remarks

bufferSize must not be less than value calculated with NTemplateCalculateSize [230] function with the same parameter values.

Module

NLTemplate Unit [153]

7.1.1.9.1.15 NLTemplateGetRecordEx Function

Retrieves the face record at the specified index of the NLTemplate [153].

C++

```
NResult N_API NLTemplateGetRecordEx(HNLTemplate hTemplate, NInt index, HNLRecord * phValue);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] Handle to the NLTemplate [153] object.
NInt index	[in] Index of face record to retrieve.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNLRecord * phValue	[out] Pointer to HNLRecord [140] that receives handle to NLRecord [134] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *phValue* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	*index* is less than zero or greater than or equal to face record count obtained using NLTemplateGetRecordCount [160] function.

Module

NLTemplate Unit [153]

7.1.1.9.1.16 NLTemplateRemoveRecord Function

Removes the face record at the specified index of the NLTemplate [153].

C++

```
NResult N_API NLTemplateRemoveRecord(HNLTemplate hTemplate, NInt index);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] Handle to the NLTemplate [153] object.
NInt index	[in] Index of face record to remove.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	*index* is less than zero or greater or equal than records count.

Module

NLTemplate Unit [153]

7.1.1.9.1.17 NLTemplateGetSizeMem Function

Retrieves the image size of the packed NLTemplate [153].

C++

```
NResult N_API NLTemplateGetSizeMem(const void * pBuffer, NSizeType bufferSize, NSizeType *
pValue);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NLTemplate [153].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NFTemplate.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSizeType * pValue	[out] Pointer to NSizeType [1008] that receives size of face image template.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NLTemplate Unit [153]

7.1.1.9.1.18 NLTemplateInsertRecord Function

Inserts the specified NLRecord [134] to the specified NLTemplate [153] at the specified index.

C++

```
NResult N_API NLTemplateInsertRecord(HNLTemplate hTemplate, NInt index, HNLRecord hValue);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] A handle to the NLTemplate [153].
NInt index	[in] Index at which the record is to be inserted.
HNLRecord hValue	[in] A handle to the NLRecord [134] to be inserted.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NLTemplate Unit [153]

7.1.1.9.1.19 NLTemplateSetRecordCapacity Function

Sets the number of face records that the NLTemplate [153] can contain.

C++

```
NResult N_API NLTemplateSetRecordCapacity(HNLTemplate hTemplate, NInt value);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] Handle to the NLTemplate [153] object.
NInt value	[in] New number of face records NLTemplate [153] can contain.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	*value* is less than face record count obtained using NLTemplateGetRecordCount [160] function.

Remarks

Face record capacity is the number of face records that the NLTemplate [153] can store. Face record count (see NLTemplateGetRecordCount [160] function) is the number of face records that are actually in the NLTemplate [153].

Capacity is always greater than or equal to count. If count exceeds capacity while adding face records the capacity is

Copyright © 2009-2014 Neurotechnology. All rights reserved.

automatically increased by reallocating the internal array before copying the old elements and adding the new elements.

Module

NLTemplate Unit [153]

7.1.1.9.1.20 HNLTemplate

Handle to the NLTemplate [153] object.

Module

NLTemplate Unit [153]

7.1.1.9.1.21 NLTemplateSetRecord Function

Sets the value of the specified NLRecord [134] to the specified value.

C++

```
NResult N_API NLTemplateSetRecord(HNLTemplate hTemplate, NInt index, HNLRecord hValue);
```

Parameters

Parameters	Description
HNLTemplate hTemplate	[in] Handle to the NLTemplate [153].
NInt index	[in] Index of the NLRecord [134] to be set.
HNLRecord hValue	[in] The new NLRecord [134] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NLTemplate Unit [153]

7.1.1.9.1.22 NLTemplateUnpack Function

Unpacks packed face records from the packed NLTemplate [153].

C++

```
NResult N_API NLTemplateUnpack(const void * pBuffer, NSizeType bufferSize, NVersion_ *
pVersion, NUInt * pSize, NByte * pHeaderSize, NInt * pRecordCount, const void * *
arpRecords, NSizeType * arRecordSizes);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NLTemplate [153].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NLTemplate [153].
NVersion_ * pVersion	[out] Pointer to NByte [1005] that receives version of the packed NLTemplate [153]. Can be NULL [1075].
NUInt * pSize	[out] Pointer to NUInt [1009] that receives size of the packed NLTemplate [153]. Can be NULL [1075].
NByte * pHeaderSize	[out] Pointer to NByte [1005] that receives header size of the packed NLTemplate [153]. Can be NULL [1075].
NInt * pRecordCount	[out] Pointer to NInt [1006] that receives number of face records contained in the packed NLTemplate [153]. Can be NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const void * * arpRecords	[in] Pointer to array of void * that receives pointers to packed face records contained in the packed NLTemplate [153]. Can be NULL [1075].
NSizeType * arRecordSizes	[out] Pointer to array of NSizeType [1008] that receives sizes of packed face records contained in the packed NLTemplate [153]. Can be NULL [1075].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer buffer points to is inconsistent with NLTemplate [153] format.

Remarks

This is a low-level function and can be changed in future version of the library.

This function should be first called with *arpRecords* and *arRecordSizes* parameters set to NULL [1075] to receive face record count through *pRecordCount* parameter. Then arrays of face records count length should be allocated and this function should be called with these array passed in *arpRecords* and *arRecordSizes* parameters.

Module

NLTemplate Unit [153]

7.1.1.9.2 Macros

7.1.1.9.2.1 NLT_MAX_RECORD_COUNT Macro

The maximum number of face records NLTemplate [153] can contain.

C++

```
#define NLT_MAX_RECORD_COUNT 255
```

Module

NLTemplate Unit [153]

7.1.1.9.2.2 NLT_PROCESS_FIRST_RECORD_ONLY Macro

The flag indicating whether only the first face record should be unpacked or packed while unpacking or packing NLTemplate [153].

C++

```
#define NLT_PROCESS_FIRST_RECORD_ONLY 0x00000100
```

Module

NLTemplate Unit [153]

7.1.1.9.2.3 NLTemplateAddRecordCopy Macro

Adds a copy of the face record to the NLTemplate [153].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NLTemplateAddRecordCopy(hTemplate, hSrcRecord, phRecord)
NLTemplateAddRecordCopyEx(hTemplate, hSrcRecord, 0, phRecord)
```

Parameters

Parameters	Description
hTemplate	[in] A handle to the NLTemplate [153].
hSrcRecord	[in] A handle to the NLRecord [134] to be copied.
phRecord	[out] A pointer to a handle to the copy of NLRecord [134].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NLTemplate Unit [153]

7.1.1.9.2.4 NLTemplateAddRecordFromMemory Macro

Unpacks a face record from the specified memory buffer and adds it to the NLTemplate [153].

C++

```
#define NLTemplateAddRecordFromMemory(hTemplate, pBuffer, bufferSize, flags, phRecord)
NLTemplateAddRecordFromMemoryEx(hTemplate, pBuffer, bufferSize, flags, NULL, phRecord)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NLTemplate [153] object.
pBuffer	[in] Pointer to memory buffer that contains packed NLRecord [134].
bufferSize	[in] Size of memory buffer that contains packed NLRecord [134].
flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
phRecord	[out] Pointer to HNLRecord [140] that receives handle to created NLRecord [134] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHRecord*, or *buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NLRecord [134] format.
N_E_INVALID_OPERATION [430]	Number of face records in NLTemplate [153] (see NLTemplateGetRecordCount [160]) is equal to NLT_MAX_RECORD_COUNT [166].
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

For the list of flags that are supported see NLRecordCreateFromMemory [136] function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NLTemplate Unit [153]

7.1.1.9.2.5 NLTemplateClone Macro

Creates a copy of the specified NLTemplate [153].

C++

```
#define NLTemplateClone(hTemplate, phClonedTemplate) NObjectClone(hTemplate, (HNObject
*)phClonedTemplate)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the original NLTemplate [153]. phClonedTemplate. : [out] Pointer to a handle to the copied NLTemplate [153].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NLTemplate Unit [153]

7.1.1.9.2.6 NLTemplateCreate Macro

Creates a new empty NLTemplate [153].

C++

```
#define NLTemplateCreate(phTemplate) NLTemplateCreateEx(0, phTemplate)
```

Parameters

Parameters	Description
phTemplate	[out] Pointer to a handle to the new NLTemplate [153].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NLTemplate Unit [153]

7.1.1.9.2.7 NLTemplateGetSize Macro

Calculates packed size of the NLTemplate [153].

C++

```
#define NLTemplateGetSize(hTemplate, flags, pValue) NObjectGetSize(hTemplate, flags, pValue)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to NLTemplate [153] object.
flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

pSize	[out] Pointer to NSizeType [1008] that receives calculated size of packed NLTemplate [153].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pSize* is NULL [1075].

Remarks

For the list of flags that are supported see NLTemplateSaveToMemory [169] function.

Module

NLTemplate Unit [153]

7.1.1.9.2.8 NLTemplateSaveToMemory Macro

Packs the NLTemplate [153] into the specified memory buffer.

C++

```
#define NLTemplateSaveToMemory(hTemplate, pBuffer, bufferSize, flags, pSize)
NObjectSaveToMemoryDst(hTemplate, pBuffer, bufferSize, flags, pSize)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to NLTemplate [153] object.
pBuffer	[out] Pointer to memory buffer to store packed NLTemplate [153]. If NULL [1075] is passed as a parameter an error occurs. To get buffer size use NLTemplateCalculateSize [156] or NLTemplateGetSize [168] functions.
bufferSize	[in] Size of memory buffer to store packed NLTemplate [153].
flags	[in] Bitwise combination of zero or more flags that control behavior of the function.
pSize	[out] Pointer to NSizeType [1008] that receives size of packed NLTemplate [153].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT [427]	*pBuffer* is not NULL [1075] and *bufferSize* is less than size required to store packed NLTemplate [153].
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pSize* is NULL [1075]. - or - *pBuffer* is NULL [1075] and *bufferSize* is not zero.

Remarks

If *pBuffer* is NULL [1075] and *bufferSize* is zero the function only calculates the size of the buffer needed and has the same effect as NLTemplateGetSize [168] function.

If *pBuffer* is not NULL [1075], bufferSize must not be less than value calculated with NLTemplateGetSize [168] function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

The following flags are supported:

- NLT_PROCESS_FIRST_RECORD_ONLY [166].

Flags supported by NLRecordSaveToMemory [141] function are applied to each face record contained in the NLTemplate [153].

Module

NLTemplate Unit [153]

7.1.1.9.3 Files

7.1.1.9.3.1 NLTemplate.h

Header file for the NLTemplate [153]. Provides functionality for packing, unpacking and editing Neurotechnology Faces [45] Templates (NLTemplates).

Functions

	Name	Description
	NLTemplateAddRecord [154]	Adds an empty face record to the NLTemplate [153].
	NLTemplateAddRecordCopyEx [155]	Adds a copy of the face record to the NLTemplate [153].
	NLTemplateAddRecordEx [156]	Adds a face record to the NLTemplate [153].
	NLTemplateAddRecordFromMemoryEx [154]	Unpacks a face record from the specified memory buffer and adds it to the NLTemplate [153].
	NLTemplateCalculateSize [156]	Calculates the size of a packed NLTemplate [153] containing the specified number of face records of specified size.
	NLTemplateCheck [157]	Checks if format of the packed NLTemplate [153] is correct.
	NLTemplateClearRecords [157]	Removes all face records from the NLTemplate [153].
	NLTemplateCreateEx [159]	Creates an empty NLTemplate [153].
	NLTemplateCreateFromMemory [158]	Unpacks a NLTemplate [153] from the specified memory buffer.
	NLTemplateGetRecord [159]	Retrieves the face record at the specified index of the NLTemplate [153].
	NLTemplateGetRecordCapacity [160]	Retrieves the number of face records that the NLTemplate [153] can contain..
	NLTemplateGetRecordCount [160]	Retrieves the number of face records in the NLTemplate [153].
	NLTemplateGetRecordCountMem [161]	Retrieves the number of face records in the packed NLTemplate [153].
	NLTemplateGetRecordEx [162]	Retrieves the face record at the specified index of the NLTemplate [153].
	NLTemplateGetSizeMem [163]	Retrieves the image size of the packed NLTemplate [153].
	NLTemplateInsertRecord [164]	Inserts the specified NLRecord [134] to the specified NLTemplate [153] at the specified index.
	NLTemplatePack [161]	Packs packed face records as NLTemplate [153] into the specified memory buffer.
	NLTemplateRemoveRecord [163]	Removes the face record at the specified index of the NLTemplate [153].
	NLTemplateSetRecord [165]	Sets the value of the specified NLRecord [134] to the specified value.
	NLTemplateSetRecordCapacity [164]	Sets the number of face records that the NLTemplate [153] can contain.
	NLTemplateUnpack [165]	Unpacks packed face records from the packed NLTemplate [153].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Macros

Name	Description
NLT_MAX_RECORD_COUNT [166]	The maximum number of face records NLTemplate [153] can contain.
NLT_PROCESS_FIRST_RECORD_ONLY [166]	The flag indicating whether only the first face record should be unpacked or packed while unpacking or packing NLTemplate [153].
NLTemplateAddRecordCopy [166]	Adds a copy of the face record to the NLTemplate [153].
NLTemplateAddRecordFromMemory [167]	Unpacks a face record from the specified memory buffer and adds it to the NLTemplate [153].
NLTemplateClone [168]	Creates a copy of the specified NLTemplate [153].
NLTemplateCreate [168]	Creates a new empty NLTemplate [153].
NLTemplateGetSize [168]	Calculates packed size of the NLTemplate [153].
NLTemplateSaveToMemory [169]	Packs the NLTemplate [153] into the specified memory buffer.

Module

NLTemplate Unit [153]

7.1.1.10 NMatcher Unit

Provides functionality for comparing all Neurotechnology Templates (NTemplates), encapsulated in Neurotechnology Matcher (NMatcher) object.

Notes

NMatcher is recommended to use for matching all types of Neurotechnology templates and records.

When NTemplate [219] contains palm records fusion is not available.

Files

Name	Description
NMatcher.h [196]	Header file for the NMatcher. Provides functionality for comparing Neurotechnology Templates (NTemplates), encapsulated in Neurotechnology Matcher (NMatcher) object.
NMatcherParams.h [197]	Header file for the NMatcher.
NMMatchDetails.h [199]	Header file for the NMatcher.

Functions

	Name	Description
	NMCreate [174]	Creates a NMatcher. Created object can be used for matching NTemplate [219].
	NMIdentifyNextEx [175]	Compares the specified packed NTemplate [219] with the one identification was started with using the specified NMatcher.
	NMIdentifyStartEx [176]	Starts identification with the specified packed NTemplate [219] using the specified NMatcher.
	NMIdentifyEnd [177]	Ends identification using the specified NMatcher.
	NMIdentifyNext [177]	Compares the specified packed NTemplate [219] with the one identification was started with using the specified NMatcher.
	NMIdentifyStart [178]	Starts identification with the specified packed NTemplate [219] using the specified NMatcher.
	NMPrepareTemplate [179]	Changes specified template for faster matching.
	NMVerifyEx [179]	Compares two packed NTemplate [219] using the specified NMatcher.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NMMatchDetailsDeserialize [180]	Deserializes NMMatchDetails [185].
	NMMatchDetailsDeserializeFromStream [180]	Deserializes NMMatchDetails [185] contained by the specified stream.
	NMMatchDetailsFree [181]	Deletes NMMatchDetails [185].
	NMVerify [181]	Compares two packed NTemplate [219] using the specified NMatcher.
	NMMatchDetailsSerialize [182]	Serializes NMMatchDetails [185].
	NMMatchDetailsSerializeToStream [183]	Serializes NMMatchDetails [185] object to memory stream.
	HNMatcher [183]	Handle to NMatcher object.

Macros

Name	Description
NM_FINGERS_MODE_ATMEL_FINGERCHIP [188]	Represents Atmel Finger Chip scanner. If this fingerprint scanner is used, it is recommended to set this identifier.
NM_FINGERS_MODE_AUTHENTEC_AES2501B [189]	Represents Authentec Aes 2501B scanner.
NM_FINGERS_MODE_AUTHENTEC_AES4000 [189]	Represents Authentec Aes 4000 scanner.
NM_FINGERS_MODE_AUTHENTEC_AFS2 [189]	Represents Authentec AFS2 scanner.
NM_FINGERS_MODE_BIOMETRIKA_FX2000 [189]	Represents Biometrika FX2000 scanner.
NM_FINGERS_MODE_BIOMETRIKA_FX3000 [189]	Represents Biometrika FX3000 scanner.
NM_FINGERS_MODE_BMF_BLP100 [189]	Represents Bmf Blp 100 scanner.
NM_FINGERS_MODE_CROSSMATCH_VERIFIER300 [189]	Represents Cross Match Verifier 300 scanner.
NM_FINGERS_MODE_DIGENT_IZZIX [189]	Represents Digent Izzix scanner.
NM_FINGERS_MODE_DIGITALPERSONA_UAREU [189]	Represents Digital Persona U are U scanner.
NM_FINGERS_MODE_ETHENTICA [190]	Represents Ethentica scanner.
NM_FINGERS_MODE_FUJITSU_MBF200 [190]	Represents Fujitsu Mbf200 scanner.
NM_FINGERS_MODE_FUTRONIC_FS80 [190]	Represents Futronic FS80 scanner. If this fingerprint scanner is used, it is recommended to set this identifier.
NM_FINGERS_MODE_GENERAL [190]	Represents general scanner.
NM_FINGERS_MODE_IDENTICATORTECHNOLOGY_DF90 [190]	Represents Identicator Technology DF90 scanner.
NM_FINGERS_MODE_IDENTIX_DFR2090 [190]	Represents Identix Dfr2090 scanner.
NM_FINGERS_MODE_IDENTIX_TOUCHVIEW [190]	Represents Identix TouchView scanner.
NM_FINGERS_MODE_KEYTRONIC_SECUREDESKTOP [190]	Represents Keytronic Secure Desktop scanner.
NM_FINGERS_MODE_LIGHTUNING_LTTC500 [190]	Represents Light Tuning LttC500 scanner.
NM_FINGERS_MODE_NITGEN_FINGKEY_HAMSTER [191]	Represents Nitgen Fingkey Hamster scanner.
NM_FINGERS_MODE_PRECISEBIOMETRICS_100CS [191]	Represents Precise Biometrics 100CS scanner.
NM_FINGERS_MODE_SECUGEN_HAMSTER [191]	Represents Secugen Hamster scanner.
NM_FINGERS_MODE_STARTEK_FM200 [191]	Represents Startek FM200 scanner.
NM_FINGERS_MODE_TACOMA_CMOS [191]	Represents Tacoma Cmos scanner.
NM_FINGERS_MODE_TESTECH_BIOI [191]	Represents Testech Bio I scanner.
NM_FINGERS_MODE_UPEK_TOUCHCHIP [191]	Represents Upek Touch Chip scanner.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NM_PART_FACES [191]	Identifier specifying that retrieving or setting parameter for faces matcher (NlsMatcher [143]) contained in NMatcher.
NM_PART_FINGERS [192]	Identifier specifying that retrieving or setting parameter for fingers matcher (NfsMatcher) contained in NMatcher.
NM_PART_IRISES [192]	Identifier specifying that retrieving or setting parameter for irises matcher (NesMatcher) contained in NMatcher.
NM_PART_NONE [192]	Identifier specifying that retrieving or setting parameter for NMatcher.
NM_PART_PALMS [192]	Identifier specifying that retrieving or setting parameter for palmprints matcher (NfsMatcher) contained in NMatcher.
NMP_FACES_MATCHER_BASE [192]	Helper parameter for internal usage.
NMP_FACES_MATCHING_SPEED [192]	Specifies face templates matching speed. Can be Low or High matching speed.
NMP_FACES_MATCHING_THRESHOLD [192]	Matters only if NMP_FUSION_TYPE [194] is NMFusionType.nmftSelectByFaceThenFuse set.
NMP_FACES_MATCHING_THRESHOLD_NEW [193]	Defines fingerprints matching threshold.
NMP_FINGERS_FILL_FULL_MATCHING_DETAILS [193]	This parameter is used when full matching details (mated minutiae) are required to be written to NMMatchDetails [185].
NMP_FINGERS_MATCHER_BASE [193]	Helper parameter for internal usage.
NMP_FINGERS_MATCHING_SPEED [193]	Specifies finger templates matching speed. Can be Low, Medium or High matching speed.
NMP_FINGERS_MAXIMAL_ROTATION [193]	Identifier specifying modulus of maximal rotation allowed between two matched NFRecords parameter of type N_TYPE_BYTE [576]. Parameter value is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).
NMP_FINGERS_MIN_MATCHED_COUNT [193]	Identifier specifying minimal number of fingerprints from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.
NMP_FINGERS_MIN_MATCHED_COUNT_THRESHOLD [193]	Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to NMP_FINGERS_MIN_MATCHED_COUNT [193] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
NMP_FINGERS_MODE [194]	Identifier specifying mode (parameter value set) parameter of type N_TYPE_UINT [578]. Parameter value can be one of the NM_FINGERS_MODE_XXX.
NMP_FUSION_TYPE [194]	Identifier specifies fingerprints, irises and faces fusion. Parameter type is NInt [1006].
NMP_IRISES_MATCHER_BASE [194]	Helper parameter for internal usage.
NMP_IRISES_MATCHING_SPEED [194]	Specifies iris templates matching speed. Can be Low, Medium or High matching speed. Parameter type is N_TYPE_INT [577].
NMP_IRISES_MATCHING_THRESHOLD [194]	Iris templates matching threshold. Matters only if NMP_FUSION_TYPE [194] is NMFusionType.nmftSelectByIrisThenFuse set. Parameter type is N_TYPE_INT [577].
NMP_IRISES_MAXIMAL_ROTATION [194]	Defines the maximal rotation allowed between two matched irises. It is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees). Parameter type is NByte [1005].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMP_IRISES_MIN_MATCHED_COUNT [194]	Identifier specifying minimal number of irises from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.
NMP_IRISES_MIN_MATCHED_COUNT_THRESHOLD [195]	Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to NMP_IRISES_MIN_MATCHED_COUNT [194] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
NMP_MATCHING_THRESHOLD [195]	Identifier specifying matching threshold (biggest allowed FAR) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
NMP_PALMS_FILL_FULL_MATCHING_DETAILS [195]	This parameter is used when full matching details are required to be written to NMMatchDetails [185].
NMP_PALMS_MATCHER_BASE [195]	Helper parameter for internal usage.
NMP_PALMS_MATCHING_SPEED [195]	Specifies plam templates matching speed. Can be Low, Medium or High matching speed.
NMP_PALMS_MAXIMAL_ROTATION [195]	Identifier specifying modulus of maximal rotation allowed between two matched NFRecords parameter of type N_TYPE_BYTE [576]. Parameter value is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).
NMP_PALMS_MIN_MATCHED_COUNT [196]	Identifier specifying minimal number of palmprints from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.
NMP_PALMS_MIN_MATCHED_COUNT_THRESHOLD [196]	Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to NMP_FINGERS_MIN_MATCHED_COUNT [193] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
NMP_VOICES_MATCHER_BASE [196]	Helper parameter for internal usage.
NMP_VOICES_UNIQUE_PHRASES_ONLY [196]	Identifier specifying the usage of unique phrases. If it was set to NTrue [1064] user during verification can say only his unique phrase(-s) (which was enrolled to database). If set to NFalse [1041] user can say any phrase.

Structs, Records, Enums

	Name	Description
	NMatchingSpeed_ [184]	Specifies biometric templates matching speed.
	NMFusionType_ [184]	Specifies the fingerprints and faces matching fusion.
	NMMatchDetails_ [185]	Represents details of matching performed with NMMatcher.
	NMatchingSpeed [184]	Specifies biometric templates matching speed.
	NMFusionType [184]	Specifies the fingerprints and faces matching fusion.
	NMMatchDetails [185]	Represents details of matching performed with NMMatcher.

7.1.1.10.1 Functions

7.1.1.10.1.1 NMCreate Function

Creates a NMatcher [171]. Created object can be used for matching NTemplate [219].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NMCreate(HNMatcher * phMatcher);
```

Parameters

Parameters	Description
HNMatcher * phMatcher	[out] Pointer to HNMatcher [183] that receives handle to created NMatcher [171] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*phMatcher* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

Created object must be deleted using NObjectFree [542] function.

Module

NMatcher Unit [171]

7.1.1.10.1.2 NMIdentifyNextEx Function

Compares the specified packed NTemplate [219] with the one identification was started with using the specified NMatcher [171].

C++

```
NResult N_API NMIdentifyNextEx(HNMatcher hMatcher, const void * pTemplate, NSizeType
templateSize, HNMatchingDetails hMatchingDetails, NInt * pScore);
```

Parameters

Parameters	Description
HNMatcher hMatcher	[in] Handle to the NMatcher [171] object.
const void * pTemplate	[in] Pointer to memory buffer containing packed NTemplate [219].
NSizeType templateSize	[in] Size of packed NTemplate [219].
HNMatchingDetails hMatchingDetails	[out] Pointer to HNMatchingDetails that is filled with details of the matching. Can be NULL [1075].
NInt * pScore	[out] Pointer to NInt [1006] that receives similarity score. See also:

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher*, *pTemplate*, or *pScore* is NULL [1075].
N_E_END_OF_STREAM [429]	*templateSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *pTemplate* points to is inconsistent with NTemplate [219] format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_INVALID_OPERATION [430]	Identification is not started.

Remarks

Value received via *pScore* is zero if similarity is less than matching threshold, i.e. two NTemplates do not match (see NMP_MATCHING_THRESHOLD [195] and NObjectSetParameterEx [564] function), and is greater than or equal to matching threshold otherwise.

If *hMatchingDetails* is not NULL [1075] it should be a pointer obtained using NMIdentifyStartEx [176] function.

Module

NMatcher Unit [171]

7.1.1.10.1.3 NMIdentifyStartEx Function

Starts identification with the specified packed NTemplate [219] using the specified NMatcher [171].

C++

```
NResult N_API NMIdentifyStartEx(HNMatcher hMatcher, const void * pTemplate, NSizeType
templateSize, HNMatchingDetails * phMatchingDetails);
```

Parameters

Parameters	Description
HNMatcher hMatcher	[in] Handle to the NMatcher [171] object.
const void * pTemplate	[in] Pointer to memory buffer containing packed NTemplate [219].
NSizeType templateSize	[in] Size of packed NTemplate [219].
HNMatchingDetails * phMatchingDetails	[out] Pointer to handle HNMatchingDetails that receives pointer to created match details. Can be NULL [1075].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher* or *pTemplate* is NULL [1075].
N_E_END_OF_STREAM [429]	*templateSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *pTemplate* points to is inconsistent with NTemplate [219] format.
N_E_INVALID_OPERATION [430]	Identification is already started.

Remarks

If *phMatchingDetails* is not NULL [1075], the received pointer should be sequent passed to NMIdentifyNextEx [175] function to obtain details of the matching.

Finally match details should be deleted using NMMatchDetailsFree [181] function.

Module

NMatcher Unit [171]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.10.1.4 NMIdentifyEnd Function

Ends identification using the specified NMatcher [171].

C++

```
NResult N_API NMIdentifyEnd(HNMatcher hMatcher);
```

Parameters

Parameters	Description
HNMatcher hMatcher	[in] Handle to the NMatcher [171] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher* is NULL [1075].
N_E_INVALID_OPERATION [430]	Identification is not started.

Module

NMatcher Unit [171]

7.1.1.10.1.5 NMIdentifyNext Function

Compares the specified packed NTemplate [219] with the one identification was started with using the specified NMatcher [171].

C++

```
NResult N_API NMIdentifyNext(HNMatcher hMatcher, const void * pTemplate, NSizeType
templateSize, NMMatchDetails * pMatchDetails, NInt * pScore);
```

Parameters

Parameters	Description
HNMatcher hMatcher	[in] Handle to the NMatcher [171] object.
const void * pTemplate	[in] Pointer to memory buffer containing packed NTemplate [219].
NSizeType templateSize	[in] Size of packed NTemplate [219].
NMMatchDetails * pMatchDetails	[out] Pointer to NMMatchDetails [185] that is filled with details of the matching. Can be NULL [1075].
NInt * pScore	[out] Pointer to NInt [1006] that receives similarity score. See also:

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher*, *templ*, or *pScore* is NULL [1075].
N_E_END_OF_STREAM [429]	*templSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *templ* points to is inconsistent with NTemplate [219] format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_INVALID_OPERATION [430]	Identification is not started.

Remarks

Value received via *pScore* is zero if similarity is less than matching threshold, i.e. two NTemplates do not match (see NMP_MATCHING_THRESHOLD [195] and NMSetParameter function), and is greater than or equal to matching threshold otherwise.

If *pMatchDetails* is not NULL [1075] it should be a pointer obtained using NMIdentifyStart [178] function.

Notes

This function is deprecated, use NMIdentifyNextEx [175] instead

Module

NMatcher Unit [171]

7.1.1.10.1.6 NMIdentifyStart Function

Starts identification with the specified packed NTemplate [219] using the specified NMatcher [171].

C++

```
NResult N_API NMIdentifyStart(HNMatcher hMatcher, const void * pTemplate, NSizeType
templateSize, NMMatchDetails * * ppMatchDetails);
```

Parameters

Parameters	Description
HNMatcher hMatcher	[in] Handle to the NMatcher [171] object.
const void * pTemplate	[in] Pointer to memory buffer containing packed NTemplate [219].
NSizeType templateSize	[in] Size of packed NTemplate [219].
NMMatchDetails * * ppMatchDetails	[out] Pointer to pointer to NMMatchDetails [185] that receives pointer to created match details. Can be NULL [1075].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher* or *templ* is NULL [1075].
N_E_END_OF_STREAM [429]	*templSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *templ* points to is inconsistent with NTemplate [219] format.
N_E_INVALID_OPERATION [430]	Identification is already started.

Remarks

If *ppMatchDetails* is not NULL [1075], the received pointer should be sequent passed to NMIdentifyNext [177] function to obtain details of the matching.

Finally match details should be deleted using NMMatchDetailsFree [181] function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Notes

This function is deprecated, use NMIdentifyStartEx [176] instead

Module

NMatcher Unit [171]

7.1.1.10.1.7 NMPrepareTemplate Function

Changes specified template for faster matching.

C++

```
NResult N_API NMPrepareTemplate(HNMatcher hMatcher, HNTemplate hTemplate);
```

Parameters

Parameters	Description
HNMatcher hMatcher	[in] Handle to the NMatcher [171] object.
HNTemplate hTemplate	[in] Handle to a template which should be chnaged (prepared for faster matching).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function is primarily used by cluster. It is not recommended to use this function directly because template size can increase. When cluster is not used, this function can be used directly to prepare template for faster matching.

Module

NMatcher Unit [171]

7.1.1.10.1.8 NMVerifyEx Function

Compares two packed NTemplate [219] using the specified NMatcher [171].

C++

```
NResult N_API NMVerifyEx(HNMatcher hMatcher, const void * pTemplate1, NSizeType
template1Size, const void * pTemplate2, NSizeType template2Size, HNMatchingDetails *
phMatchingDetails, NInt * pScore);
```

Parameters

Parameters	Description
HNMatcher hMatcher	[in] Handle to the NMatcher [171] object.
const void * pTemplate1	[in] Pointer to memory buffer containing first packed NTemplate [219].
NSizeType template1Size	[in] Size of first packed NTemplate [219].
const void * pTemplate2	[in] Pointer to memory buffer containing second packed NTemplate [219].
NSizeType template2Size	[in] Size of second packed NTemplate [219].
HNMatchingDetails * phMatchingDetails	[out] Pointer to handle to HNMatchingDetails that receives pointer to created match details. Can be NULL [1075].
NInt * pScore	[out] Pointer to NInt [1006] that receives similarity score.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher*, *pTemplate1*, *pTemplate2*, or *pScore* is NULL [1075].
N_E_END_OF_STREAM [429]	*template1Size* or *template2Size* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *pTemplate1* or *pTemplate2* points to is inconsistent with NFTemplate format.
N_E_INVALID_OPERATION [430]	Identification is started.

Remarks

Value received via *pScore* is zero if similarity is less than matching threshold, i.e. two NTemplates do not match (see NMP_MATCHING_THRESHOLD [195] and NObjectSetParameterEx [564] function), and is greater than or equal to matching threshold otherwise.

If *phMatchDetails* is not NULL [1075], the received pointer should be examined to obtain details of the matching.

Finally match details should be deleted using NMMatchDetailsFree [181] function.

Module

NMatcher Unit [171]

7.1.1.10.1.9 NMMatchDetailsDeserialize Function

Deserializes NMMatchDetails [185].

C++

```
NResult N_API NMMatchDetailsDeserialize(const void * pBuffer, NSizeType bufferSize,
NMMatchDetails * * ppMatchDetails);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to void * that receives pointer to serialized NMMatchDetails [185].
NSizeType bufferSize	[in] Size of allocated memory buffer
NMMatchDetails * * ppMatchDetails	[out] Pointer to pointer to NMMatchDetails [185] that returns pointer to created match details. Can be NULL [1075].

Returns

If the function succeeds, the return value is N_OK [433].

Module

NMatcher Unit [171]

7.1.1.10.1.10 NMMatchDetailsDeserializeFromStream Function

Deserializes NMMatchDetails [185] contained by the specified stream.

C++

```
NResult N_API NMMatchDetailsDeserializeFromStream(HNStream hStream, NMMatchDetails * *
ppMatchDetails);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNStream hStream	[in] Handle to memory stream that receives pointer to serialized NMMatchDetails [185].
NMMatchDetails * * ppMatchDetails	[out] Pointer to pointer to NMMatchDetails [185] that returns pointer to created match details. Can be NULL [1075].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hStream* or *ppMatchDetails* is NULL [1075].

Module

NMatcher Unit [171]

7.1.1.10.1.11 NMMatchDetailsFree Function

Deletes NMMatchDetails [185].

C++

```
void N_API NMMatchDetailsFree(NMMatchDetails * pMatchDetails);
```

Parameters

Parameters	Description
NMMatchDetails * pMatchDetails	[in] Pointer to NMMatchDetails [185] to delete.

Remarks

If *pMatchDetails* is NULL [1075], does nothing.

Module

NMatcher Unit [171]

7.1.1.10.1.12 NMVerify Function

Compares two packed NTemplate [219] using the specified NMatcher [171].

C++

```
NResult N_API NMVerify(HNMatcher hMatcher, const void * pTemplate1, NSizeType
template1Size, const void * pTemplate2, NSizeType template2Size, NMMatchDetails * *
ppMatchDetails, NInt * pScore);
```

Parameters

Parameters	Description
HNMatcher hMatcher	[in] Handle to the NMatcher [171] object.
const void * pTemplate1	[in] Pointer to memory buffer containing first packed NTemplate [219].
NSizeType template1Size	[in] Size of first packed NTemplate [219].
const void * pTemplate2	[in] Pointer to memory buffer containing second packed NTemplate [219].
NSizeType template2Size	[in] Size of second packed NTemplate [219].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMMatchDetails * * ppMatchDetails	[out] Pointer to pointer to NMMatchDetails [185] that receives pointer to created match details. Can be NULL [1075].
NInt * pScore	[out] Pointer to NInt [1006] that receives similarity score.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hMatcher*, *template1*, *template2*, or *pScore* is NULL [1075].
N_E_END_OF_STREAM [429]	*template1Size* or *template2Size* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *template1* or *template2* points to is inconsistent with NFTemplate format.
N_E_INVALID_OPERATION [430]	Identification is started.

Remarks

Value received via *pScore* is zero if similarity is less than matching threshold, i.e. two NTemplates do not match (see NMP_MATCHING_THRESHOLD [195] and NMSetParameter function), and is greater than or equal to matching threshold otherwise.

If *ppMatchDetails* is not NULL [1075], the received pointer should be examined to obtain details of the matching.

Finally match details should be deleted using NMMatchDetailsFree [181] function.

Notes

This function is deprecated, use NMVerifytEx instead

Module

NMatcher Unit [171]

7.1.1.10.1.13 NMMatchDetailsSerialize Function

Serializes NMMatchDetails [185].

C++

```
NResult N_API NMMatchDetailsSerialize(NMMatchDetails * pMatchDetails, void * * ppBuffer,
NSizeType * pBufferSize);
```

Parameters

Parameters	Description
NMMatchDetails * pMatchDetails	[out] Pointer to NMMatchDetails [185] to serialize.
void * * ppBuffer	[out] Pointer to void * that receives pointer to allocated memory buffer.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer.

Returns

If the function succeeds, the return value is N_OK [433].

Module

NMatcher Unit [171]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.10.1.14 NMMatchDetailsSerializeToStream Function

Serializes NMMatchDetails [185] object to memory stream.

C++

```
NResult N_API NMMatchDetailsSerializeToStream(NMMatchDetails * pMatchDetails, HNStream
hStream);
```

Parameters

Parameters	Description
NMMatchDetails * pMatchDetails	[out] Pointer to NMMatchDetails [185] to serialize.
HNStream hStream	[out] Handle to memory stream.

Returns

If the function succeeds, the return value is N_OK [433].

Module

NMatcher Unit [171]

7.1.1.10.1.15 HNMatcher

Handle to NMatcher [171] object.

Remarks

The following parameters can be passed to function with corresponding handle:

- NM_PART_FACES [191]
- NM_PART_FINGERS [192]
- NM_PART_PALMS [192]
- NM_PART_IRISES [192]
- NM_PART_NONE [192]
- NMP_FACES_MATCHING_THRESHOLD [192]
- NMP_FACES_MATCHING_THRESHOLD_NEW [193]
- NMP_IRISES_MATCHING_THRESHOLD [194]
- NMP_FUSION_TYPE [194]
- NMP_MATCHING_THRESHOLD [195]
- NMP_FINGERS_MATCHER_BASE [193]
- NMP_FINGERS_MAXIMAL_ROTATION [193]
- NMP_FINGERS_MATCHING_SPEED [193]
- NMP_FINGERS_FILL_FULL_MATCHING_DETAILS [193]
- NMP_FINGERS_MODE [194]
- NMP_FINGERS_MIN_MATCHED_COUNT [193]
- NMP_FINGERS_MIN_MATCHED_COUNT_THRESHOLD [193]
- NMP_FACES_MATCHER_BASE [192]
- NMP_FACES_MATCHING_SPEED [192]
- NMP_IRISES_MATCHER_BASE [194]
- NMP_IRISES_MATCHING_SPEED [194]
- NMP_IRISES_MAXIMAL_ROTATION [194]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- NMP_IRISES_MIN_MATCHED_COUNT [194]
- NMP_IRISES_MIN_MATCHED_COUNT_THRESHOLD [195]
- NMP_PALMS_MATCHER_BASE [195]
- NMP_PALMS_MAXIMAL_ROTATION [195]
- NMP_PALMS_MATCHING_SPEED [195]
- NMP_PALMS_FILL_FULL_MATCHING_DETAILS [195]
- NMP_PALMS_MIN_MATCHED_COUNT [196]
- NMP_PALMS_MIN_MATCHED_COUNT_THRESHOLD [196]
- NMP_VOICES_MATCHER_BASE [196]
- NMP_VOICES_UNIQUE_PHRASES_ONLY [196]

See Also

NObjectGetParameterWithPartEx [546]

Module

NMatcher Unit [171]

7.1.1.10.2 Structs, Records, Enums

7.1.1.10.2.1 NMatchingSpeed Enumeration

Specifies biometric templates matching speed.

C++

```
typedef enum NMatchingSpeed_ {
  nmsLow = 0,
  nmsMedium = 128,
  nmsHigh = 256
} NMatchingSpeed;
```

Members

Members	Description
nmsLow = 0	Low matching speed (slower but more accurate).
nmsMedium = 128	Medium matching speed.
nmsHigh = 256	High matching speed (faster but less accurate).

Module

NMatcher Unit [171]

7.1.1.10.2.2 NMFusionType Enumeration

Specifies the fingerprints and faces matching fusion.

C++

```
typedef enum NMFusionType_ {
  nmftFuseAlways = 0,
  nmftSelectByFaceThenFuse = 1,
  nmftSelectByIrisThenFuse = 2
} NMFusionType;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Members

Members	Description
nmftFuseAlways = 0	Matches the fingerprints, palmprints, irises and faces. The results of matchings are fused also.
nmftSelectByFaceThenFuse = 1	Matches faces. If result is greater than NMP_FACES_MATCHING_THRESHOLD_NEW [193], then matches other modalities and fuses results; otherwise, score is zero.
nmftSelectByIrisThenFuse = 2	Matches Iris. If result is greater than NMP_IRISES_MATCHING_THRESHOLD [194], then matches other modalities and fuses results; otherwise, score is zero.

Module

NMatcher Unit [171]

7.1.1.10.2.3 NMMatchDetails Structure

Represents details of matching performed with NMMatcher.

C++

```
typedef struct NMMatchDetails_ {
  NInt Score;
  NfsmMatchDetails * FingersMatchDetails;
  NlsmMatchDetails * FacesMatchDetails;
  NesmMatchDetails * IrisesMatchDetails;
  NfsmMatchDetails * PalmsMatchDetails;
} NMMatchDetails;
```

Members

Members	Description
NInt Score;	Similarity score of two matched NTemplates.
NfsmMatchDetails * FingersMatchDetails;	Pointer to NfsmMatchDetails containing matching details from NfsMatcher.
NlsmMatchDetails * FacesMatchDetails;	Pointer to NlsmMatchDetails [150] containing matching details from NfsMatcher.
NesmMatchDetails * IrisesMatchDetails;	Pointer to NesmIrisMatchDetails containing matching details from NfsMatcher.
NfsmMatchDetails * PalmsMatchDetails;	Pointer to NfsmMatchDetails containing matching details from NfsMatcher.

Module

NMatcher Unit [171]

7.1.1.10.3 Macros

To get a parameter value, use NObjectGetParameterWithPartEx [546] function. To set parameter value - NObjectSetParameterWithPartEx [557] function.

Parameter *partId* passed to these functions can be one of the following:

- NM_PART_FACES [191] + any *partId* of NlsMatcher macros [150]
- NM_PART_FINGERS [192] + any *partId* of NfsMatcher macros
- NM_PART_PALMS [192] + any *partId* of NfsMatcher macros
- NM_PART_IRISES [192] + any *partId* of NesMatcher macros

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- NM_PART_NONE [192]

If *partId* is NM_PART_FINGERS [192] + any *partId* of NfsMatcher macros, *parameterId* must be any of NfsMatcher macros except for NFSMP_MATCHING_THRESHOLD.

If *partId* is NM_PART_FACES [191] + any *partId* of NlsMatcher macros [150], *parameterId* must be any of NlsMatcher macros [150] except for NLSMP_MATCHING_THRESHOLD [151].

If *partId* is NM_PART_IRISES [192] + any *partId* of NesMatcher macros, *parameterId* must be any of NesMatcher macros except for NESMP_MATCHING_THRESHOLD.

If *partId* is NM_PART_NONE [192], the following values can be used for *parameterId*:

- NMP_FACES_MATCHING_THRESHOLD [192]
- NMP_FACES_MATCHING_THRESHOLD [192]
- NMP_FUSION_TYPE [194]
- NMP_MATCHING_THRESHOLD [195]

Macros

Name	Description
NM_FINGERS_MODE_ATMEL_FINGERCHIP [188]	Represents Atmel Finger Chip scanner. If this fingerprint scanner is used, it is recommended to set this identifier.
NM_FINGERS_MODE_AUTHENTEC_AES2501B [189]	Represents Authentec Aes 2501B scanner.
NM_FINGERS_MODE_AUTHENTEC_AES4000 [189]	Represents Authentec Aes 4000 scanner.
NM_FINGERS_MODE_AUTHENTEC_AFS2 [189]	Represents Authentec AFS2 scanner.
NM_FINGERS_MODE_BIOMETRIKA_FX2000 [189]	Represents Biometrika FX2000 scanner.
NM_FINGERS_MODE_BIOMETRIKA_FX3000 [189]	Represents Biometrika FX3000 scanner.
NM_FINGERS_MODE_BMF_BLP100 [189]	Represents Bmf Blp 100 scanner.
NM_FINGERS_MODE_CROSSMATCH_VERIFIER300 [189]	Represents Cross Match Verifier 300 scanner.
NM_FINGERS_MODE_DIGENT_IZZIX [189]	Represents Digent Izzix scanner.
NM_FINGERS_MODE_DIGITALPERSONA_UAREU [189]	Represents Digital Persona U are U scanner.
NM_FINGERS_MODE_ETHENTICA [190]	Represents Ethentica scanner.
NM_FINGERS_MODE_FUJITSU_MBF200 [190]	Represents Fujitsu Mbf200 scanner.
NM_FINGERS_MODE_FUTRONIC_FS80 [190]	Represents Futronic FS80 scanner. If this fingerprint scanner is used, it is recommended to set this identifier.
NM_FINGERS_MODE_GENERAL [190]	Represents general scanner.
NM_FINGERS_MODE_IDENTICATORTECHNOLOGY_DF90 [190]	Represents Identicator Technology DF90 scanner.
NM_FINGERS_MODE_IDENTIX_DFR2090 [190]	Represents Identix Dfr2090 scanner.
NM_FINGERS_MODE_IDENTIX_TOUCHVIEW [190]	Represents Identix TouchView scanner.
NM_FINGERS_MODE_KEYTRONIC_SECUREDESKTOP [190]	Represents Keytronic Secure Desktop scanner.
NM_FINGERS_MODE_LIGHTUNING_LTTC500 [190]	Represents Light Tuning LttC500 scanner.
NM_FINGERS_MODE_NITGEN_FINGKEY_HAMSTER [191]	Represents Nitgen Fingkey Hamster scanner.
NM_FINGERS_MODE_PRECISEBIOMETRICS_100CS [191]	Represents Precise Biometrics 100CS scanner.
NM_FINGERS_MODE_SECUGEN_HAMSTER [191]	Represents Secugen Hamster scanner.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NM_FINGERS_MODE_STARTEK_FM200 [191]	Represents Startek FM200 scanner.
NM_FINGERS_MODE_TACOMA_CMOS [191]	Represents Tacoma Cmos scanner.
NM_FINGERS_MODE_TESTECH_BIOI [191]	Represents Testech Bio I scanner.
NM_FINGERS_MODE_UPEK_TOUCHCHIP [191]	Represents Upek Touch Chip scanner.
NM_PART_FACES [191]	Identifier specifying that retrieving or setting parameter for faces matcher (NlsMatcher [143]) contained in NMatcher [171].
NM_PART_FINGERS [192]	Identifier specifying that retrieving or setting parameter for fingers matcher (NfsMatcher) contained in NMatcher [171].
NM_PART_IRISES [192]	Identifier specifying that retrieving or setting parameter for irises matcher (NesMatcher) contained in NMatcher [171].
NM_PART_NONE [192]	Identifier specifying that retrieving or setting parameter for NMatcher [171].
NM_PART_PALMS [192]	Identifier specifying that retrieving or setting parameter for palmprints matcher (NfsMatcher) contained in NMatcher [171].
NMP_FACES_MATCHER_BASE [192]	Helper parameter for internal usage.
NMP_FACES_MATCHING_SPEED [192]	Specifies face templates matching speed. Can be Low or High matching speed.
NMP_FACES_MATCHING_THRESHOLD [192]	Matters only if NMP_FUSION_TYPE [194] is NMFusionType.nmftSelectByFaceThenFuse set.
NMP_FACES_MATCHING_THRESHOLD_NEW [193]	Defines fingerprints matching threshold.
NMP_FINGERS_FILL_FULL_MATCHING_DETAILS [193]	This parameter is used when full matching details (mated minutiae) are required to be written to NMMatchDetails [185].
NMP_FINGERS_MATCHER_BASE [193]	Helper parameter for internal usage.
NMP_FINGERS_MATCHING_SPEED [193]	Specifies finger templates matching speed. Can be Low, Medium or High matching speed.
NMP_FINGERS_MAXIMAL_ROTATION [193]	Identifier specifying modulus of maximal rotation allowed between two matched NFRecords parameter of type N_TYPE_BYTE [576]. Parameter value is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).
NMP_FINGERS_MIN_MATCHED_COUNT [193]	Identifier specifying minimal number of fingerprints from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.
NMP_FINGERS_MIN_MATCHED_COUNT_THRESHOLD [193]	Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to NMP_FINGERS_MIN_MATCHED_COUNT [193] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
NMP_FINGERS_MODE [194]	Identifier specifying mode (parameter value set) parameter of type N_TYPE_UINT [578]. Parameter value can be one of the NM_FINGERS_MODE_XXX.
NMP_FUSION_TYPE [194]	Identifier specifies fingerprints, irises and faces fusion. Parameter type is NInt [1006].
NMP_IRISES_MATCHER_BASE [194]	Helper parameter for internal usage.
NMP_IRISES_MATCHING_SPEED [194]	Specifies iris templates matching speed. Can be Low, Medium or High matching speed. Parameter type is N_TYPE_INT [577].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMP_IRISES_MATCHING_THRESHOLD [194]	Iris templates matching threshold. Matters only if NMP_FUSION_TYPE [194] is NMFusionType.nmftSelectByIrisThenFuse set. Parameter type is N_TYPE_INT [577].
NMP_IRISES_MAXIMAL_ROTATION [194]	Defines the maximal rotation allowed between two matched irises. It is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees). Parameter type is NByte [1005].
NMP_IRISES_MIN_MATCHED_COUNT [194]	Identifier specifying minimal number of irises from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.
NMP_IRISES_MIN_MATCHED_COUNT_THRESHOLD [195]	Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to NMP_IRISES_MIN_MATCHED_COUNT [194] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
NMP_MATCHING_THRESHOLD [195]	Identifier specifying matching threshold (biggest allowed FAR) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
NMP_PALMS_FILL_FULL_MATCHING_DETAILS [195]	This parameter is used when full matching details are required to be written to NMMatchDetails [185].
NMP_PALMS_MATCHER_BASE [195]	Helper parameter for internal usage.
NMP_PALMS_MATCHING_SPEED [195]	Specifies plam templates matching speed. Can be Low, Medium or High matching speed.
NMP_PALMS_MAXIMAL_ROTATION [195]	Identifier specifying modulus of maximal rotation allowed between two matched NFRecords parameter of type N_TYPE_BYTE [576]. Parameter value is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).
NMP_PALMS_MIN_MATCHED_COUNT [196]	Identifier specifying minimal number of palmprints from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.
NMP_PALMS_MIN_MATCHED_COUNT_THRESHOLD [196]	Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to NMP_FINGERS_MIN_MATCHED_COUNT [193] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
NMP_VOICES_MATCHER_BASE [196]	Helper parameter for internal usage.
NMP_VOICES_UNIQUE_PHRASES_ONLY [196]	Identifier specifying the usage of unique phrases. If it was set to NTrue [1064] user during verification can say only his unique phrase(-s) (which was enrolled to database). If set to NFalse [1041] user can say any phrase.

Module

NMatcher Unit [171]

7.1.1.10.3.1 NM_FINGERS_MODE_ATMEL_FINGERCHIP Macro

Represents Atmel Finger Chip scanner. If this fingerprint scanner is used, it is recommended to set this identifier.

C++

```
#define NM_FINGERS_MODE_ATMEL_FINGERCHIP 900
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.10.3.2 NM_FINGERS_MODE_AUTHENTEC_AES2501B Macro

Represents Authentec Aes 2501B scanner.

C++

```
#define NM_FINGERS_MODE_AUTHENTEC_AES2501B 811
```

7.1.1.10.3.3 NM_FINGERS_MODE_AUTHENTEC_AES4000 Macro

Represents Authentec Aes 4000 scanner.

C++

```
#define NM_FINGERS_MODE_AUTHENTEC_AES4000 810
```

7.1.1.10.3.4 NM_FINGERS_MODE_AUTHENTEC_AFS2 Macro

Represents Authentec AFS2 scanner.

C++

```
#define NM_FINGERS_MODE_AUTHENTEC_AFS2 800
```

7.1.1.10.3.5 NM_FINGERS_MODE_BIOMETRIKA_FX2000 Macro

Represents Biometrika FX2000 scanner.

C++

```
#define NM_FINGERS_MODE_BIOMETRIKA_FX2000 200
```

7.1.1.10.3.6 NM_FINGERS_MODE_BIOMETRIKA_FX3000 Macro

Represents Biometrika FX3000 scanner.

C++

```
#define NM_FINGERS_MODE_BIOMETRIKA_FX3000 201
```

7.1.1.10.3.7 NM_FINGERS_MODE_BMF_BLP100 Macro

Represents Bmf Blp 100 scanner.

C++

```
#define NM_FINGERS_MODE_BMF_BLP100 1000
```

7.1.1.10.3.8 NM_FINGERS_MODE_CROSSMATCH_VERIFIER300 Macro

Represents Cross Match Verifier 300 scanner.

C++

```
#define NM_FINGERS_MODE_CROSSMATCH_VERIFIER300 1300
```

7.1.1.10.3.9 NM_FINGERS_MODE_DIGENT_IZZIX Macro

Represents Digent Izzix scanner.

C++

```
#define NM_FINGERS_MODE_DIGENT_IZZIX 1600
```

7.1.1.10.3.10 NM_FINGERS_MODE_DIGITALPERSONA_UAREU Macro

Represents Digital Persona U are U scanner.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NM_FINGERS_MODE_DIGITALPERSONA_UAREU 100
```

7.1.1.10.3.11 NM_FINGERS_MODE_ETHENTICA Macro

Represents Ethentica scanner.

C++

```
#define NM_FINGERS_MODE_ETHENTICA 1200
```

7.1.1.10.3.12 NM_FINGERS_MODE_FUJITSU_MBF200 Macro

Represents Fujitsu Mbf200 scanner.

C++

```
#define NM_FINGERS_MODE_FUJITSU_MBF200 1800
```

7.1.1.10.3.13 NM_FINGERS_MODE_FUTRONIC_FS80 Macro

Represents Futronic FS80 scanner. If this fingerprint scanner is used, it is recommended to set this identifier.

C++

```
#define NM_FINGERS_MODE_FUTRONIC_FS80 1900
```

7.1.1.10.3.14 NM_FINGERS_MODE_GENERAL Macro

Represents general scanner.

C++

```
#define NM_FINGERS_MODE_GENERAL 0
```

7.1.1.10.3.15 NM_FINGERS_MODE_IDENTICATORTECHNOLOGY_DF90 Macro

Represents Identicator Technology DF90 scanner.

C++

```
#define NM_FINGERS_MODE_IDENTICATORTECHNOLOGY_DF90 700
```

7.1.1.10.3.16 NM_FINGERS_MODE_IDENTIX_DFR2090 Macro

Represents Identix Dfr2090 scanner.

C++

```
#define NM_FINGERS_MODE_IDENTIX_DFR2090 401
```

7.1.1.10.3.17 NM_FINGERS_MODE_IDENTIX_TOUCHVIEW Macro

Represents Identix TouchView scanner.

C++

```
#define NM_FINGERS_MODE_IDENTIX_TOUCHVIEW 400
```

7.1.1.10.3.18 NM_FINGERS_MODE_KEYTRONIC_SECUREDESKTOP Macro

Represents Keytronic Secure Desktop scanner.

C++

```
#define NM_FINGERS_MODE_KEYTRONIC_SECUREDESKTOP 300
```

7.1.1.10.3.19 NM_FINGERS_MODE_LIGHTUNING_LTTC500 Macro

Represents Light Tuning LttC500 scanner.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NM_FINGERS_MODE_LIGHTUNING_LTTC500 2000
```

7.1.1.10.3.20 NM_FINGERS_MODE_NITGEN_FINGKEY_HAMSTER Macro

Represents Nitgen Fingkey Hamster scanner.

C++

```
#define NM_FINGERS_MODE_NITGEN_FINGKEY_HAMSTER 1400
```

7.1.1.10.3.21 NM_FINGERS_MODE_PRECISEBIOMETRICS_100CS Macro

Represents Precise Biometrics 100CS scanner.

C++

```
#define NM_FINGERS_MODE_PRECISEBIOMETRICS_100CS 500
```

7.1.1.10.3.22 NM_FINGERS_MODE_SECUGEN_HAMSTER Macro

Represents Secugen Hamster scanner.

C++

```
#define NM_FINGERS_MODE_SECUGEN_HAMSTER 1100
```

7.1.1.10.3.23 NM_FINGERS_MODE_STARTEK_FM200 Macro

Represents Startek FM200 scanner.

C++

```
#define NM_FINGERS_MODE_STARTEK_FM200 1700
```

7.1.1.10.3.24 NM_FINGERS_MODE_TACOMA_CMOS Macro

Represents Tacoma Cmos scanner.

C++

```
#define NM_FINGERS_MODE_TACOMA_CMOS 2100
```

7.1.1.10.3.25 NM_FINGERS_MODE_TESTECH_BIOI Macro

Represents Testech Bio I scanner.

C++

```
#define NM_FINGERS_MODE_TESTECH_BIOI 1500
```

7.1.1.10.3.26 NM_FINGERS_MODE_UPEK_TOUCHCHIP Macro

Represents Upek Touch Chip scanner.

C++

```
#define NM_FINGERS_MODE_UPEK_TOUCHCHIP 600
```

7.1.1.10.3.27 NM_PART_FACES Macro

Identifier specifying that retrieving or setting parameter for faces matcher (NlsMatcher [143]) contained in NMatcher [171].

C++

```
#define NM_PART_FACES 20
```

See Also

NMGetParameter

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMSetParameter

7.1.1.10.3.28 NM_PART_FINGERS Macro

Identifier specifying that retrieving or setting parameter for fingers matcher (NfsMatcher) contained in NMatcher [171].

C++

```
#define NM_PART_FINGERS 10
```

See Also

NMGetParameter

NMSetParameter

7.1.1.10.3.29 NM_PART_IRISES Macro

Identifier specifying that retrieving or setting parameter for irises matcher (NesMatcher) contained in NMatcher [171].

C++

```
#define NM_PART_IRISES 30
```

7.1.1.10.3.30 NM_PART_NONE Macro

Identifier specifying that retrieving or setting parameter for NMatcher [171].

C++

```
#define NM_PART_NONE 0
```

See Also

NMGetParameter

NMSetParameter

7.1.1.10.3.31 NM_PART_PALMS Macro

Identifier specifying that retrieving or setting parameter for palmprints matcher (NfsMatcher) contained in NMatcher [171].

C++

```
#define NM_PART_PALMS 40
```

7.1.1.10.3.32 NMP_FACES_MATCHER_BASE Macro

Helper parameter for internal usage.

C++

```
#define NMP_FACES_MATCHER_BASE 11536
```

7.1.1.10.3.33 NMP_FACES_MATCHING_SPEED Macro

Specifies face templates matching speed. Can be Low or High matching speed.

C++

```
#define NMP_FACES_MATCHING_SPEED (NMP_FACES_MATCHER_BASE +    1)
```

7.1.1.10.3.34 NMP_FACES_MATCHING_THRESHOLD Macro

Matters only if NMP_FUSION_TYPE [194] is NMFusionType.nmftSelectByFaceThenFuse set.

C++

```
#define NMP_FACES_MATCHING_THRESHOLD 210
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Notes

This macro is obsolete. Use NMP_FACES_MATCHING_THRESHOLD_NEW [193] instead.

7.1.1.10.3.35 NMP_FACES_MATCHING_THRESHOLD_NEW Macro

Defines fingerprints matching threshold.

C++

```
#define NMP_FACES_MATCHING_THRESHOLD_NEW 215
```

See Also

Read more about fingerprints matching threshold [13].

7.1.1.10.3.36 NMP_FINGERS_FILL_FULL_MATCHING_DETAILS Macro

This parameter is used when full matching details (mated minutiae) are required to be written to NMMatchDetails [185].

C++

```
#define NMP_FINGERS_FILL_FULL_MATCHING_DETAILS (NMP_FINGERS_MATCHER_BASE +  900)
```

7.1.1.10.3.37 NMP_FINGERS_MATCHER_BASE Macro

Helper parameter for internal usage.

C++

```
#define NMP_FINGERS_MATCHER_BASE 3536
```

7.1.1.10.3.38 NMP_FINGERS_MATCHING_SPEED Macro

Specifies finger templates matching speed. Can be Low, Medium or High matching speed.

C++

```
#define NMP_FINGERS_MATCHING_SPEED (NMP_FINGERS_MATCHER_BASE +  220)
```

Notes

High speed is equal to Medium.

7.1.1.10.3.39 NMP_FINGERS_MAXIMAL_ROTATION Macro

Identifier specifying modulus of maximal rotation allowed between two matched NFRecords parameter of type N_TYPE_BYTE [576]. Parameter value is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).

C++

```
#define NMP_FINGERS_MAXIMAL_ROTATION (NMP_FINGERS_MATCHER_BASE +  201)
```

7.1.1.10.3.40 NMP_FINGERS_MIN_MATCHED_COUNT Macro

Identifier specifying minimal number of fingerprints from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.

C++

```
#define NMP_FINGERS_MIN_MATCHED_COUNT (NMP_FINGERS_MATCHER_BASE + 1201)
```

7.1.1.10.3.41 NMP_FINGERS_MIN_MATCHED_COUNT_THRESHOLD Macro

Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to NMP_FINGERS_MIN_MATCHED_COUNT [193] parameter value) parameter of type N_TYPE_INT [577]. Parameter

Copyright © 2009-2014 Neurotechnology. All rights reserved.

value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

C++

```
#define NMP_FINGERS_MIN_MATCHED_COUNT_THRESHOLD (NMP_FINGERS_MATCHER_BASE + 1202)
```

7.1.1.10.3.42 NMP_FINGERS_MODE Macro

Identifier specifying mode (parameter value set) parameter of type N_TYPE_UINT [578]. Parameter value can be one of the NM_FINGERS_MODE_XXX.

C++

```
#define NMP_FINGERS_MODE (NMP_FINGERS_MATCHER_BASE + 1000)
```

7.1.1.10.3.43 NMP_FUSION_TYPE Macro

Identifier specifies fingerprints, irises and faces fusion. Parameter type is NInt [1006].

C++

```
#define NMP_FUSION_TYPE 200
```

7.1.1.10.3.44 NMP_IRISES_MATCHER_BASE Macro

Helper parameter for internal usage.

C++

```
#define NMP_IRISES_MATCHER_BASE 19536
```

7.1.1.10.3.45 NMP_IRISES_MATCHING_SPEED Macro

Specifies iris templates matching speed. Can be Low, Medium or High matching speed. Parameter type is N_TYPE_INT [577].

C++

```
#define NMP_IRISES_MATCHING_SPEED (NMP_IRISES_MATCHER_BASE +    1)
```

Notes

High speed is equal to Medium.

7.1.1.10.3.46 NMP_IRISES_MATCHING_THRESHOLD Macro

Iris templates matching threshold. Matters only if NMP_FUSION_TYPE [194] is NMFusionType.nmftSelectByIrisThenFuse set. Parameter type is N_TYPE_INT [577].

C++

```
#define NMP_IRISES_MATCHING_THRESHOLD 220
```

See Also

For more information about matching threshold see Matching Threshold and Score [13] chapter.

7.1.1.10.3.47 NMP_IRISES_MAXIMAL_ROTATION Macro

Defines the maximal rotation allowed between two matched irises. It is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees). Parameter type is NByte [1005].

C++

```
#define NMP_IRISES_MAXIMAL_ROTATION (NMP_IRISES_MATCHER_BASE +    2)
```

7.1.1.10.3.48 NMP_IRISES_MIN_MATCHED_COUNT Macro

Identifier specifying minimal number of irises from probe template to match parameter of type N_TYPE_INT [577].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameter value is from zero to NFT_MAX_RECORD_COUNT.

C++

```
#define NMP_IRISES_MIN_MATCHED_COUNT (NMP_IRISES_MATCHER_BASE + 1201)
```

7.1.1.10.3.49 NMP_IRISES_MIN_MATCHED_COUNT_THRESHOLD Macro

Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to NMP_IRISES_MIN_MATCHED_COUNT [194] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

C++

```
#define NMP_IRISES_MIN_MATCHED_COUNT_THRESHOLD (NMP_IRISES_MATCHER_BASE + 1202)
```

7.1.1.10.3.50 NMP_MATCHING_THRESHOLD Macro

Identifier specifying matching threshold (biggest allowed FAR) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

C++

```
#define NMP_MATCHING_THRESHOLD 100
```

See Also

Matching Threshold and FAR/FRR [13]

7.1.1.10.3.51 NMP_PALMS_FILL_FULL_MATCHING_DETAILS Macro

This parameter is used when full matching details are required to be written to NMMatchDetails [185].

C++

```
#define NMP_PALMS_FILL_FULL_MATCHING_DETAILS (NMP_PALMS_MATCHER_BASE +  900)
```

7.1.1.10.3.52 NMP_PALMS_MATCHER_BASE Macro

Helper parameter for internal usage.

C++

```
#define NMP_PALMS_MATCHER_BASE 27536
```

7.1.1.10.3.53 NMP_PALMS_MATCHING_SPEED Macro

Specifies plam templates matching speed. Can be Low, Medium or High matching speed.

C++

```
#define NMP_PALMS_MATCHING_SPEED (NMP_PALMS_MATCHER_BASE +  220)
```

Notes

High speed is equal to Medium.

7.1.1.10.3.54 NMP_PALMS_MAXIMAL_ROTATION Macro

Identifier specifying modulus of maximal rotation allowed between two matched NFRecords parameter of type N_TYPE_BYTE [576]. Parameter value is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).

C++

```
#define NMP_PALMS_MAXIMAL_ROTATION (NMP_PALMS_MATCHER_BASE +  201)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.10.3.55 NMP_PALMS_MIN_MATCHED_COUNT Macro

Identifier specifying minimal number of palmprints from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.

C++

```
#define NMP_PALMS_MIN_MATCHED_COUNT (NMP_PALMS_MATCHER_BASE + 1201)
```

7.1.1.10.3.56 NMP_PALMS_MIN_MATCHED_COUNT_THRESHOLD Macro

Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to NMP_FINGERS_MIN_MATCHED_COUNT [193] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

C++

```
#define NMP_PALMS_MIN_MATCHED_COUNT_THRESHOLD (NMP_PALMS_MATCHER_BASE + 1202)
```

7.1.1.10.3.57 NMP_VOICES_MATCHER_BASE Macro

Helper parameter for internal usage.

C++

```
#define NMP_VOICES_MATCHER_BASE 35536
```

7.1.1.10.3.58 NMP_VOICES_UNIQUE_PHRASES_ONLY Macro

Identifier specifying the usage of unique phrases. If it was set to NTrue [1064] user during verification can say only his unique phrase(-s) (which was enrolled to database). If set to NFalse [1041] user can say any phrase.

C++

```
#define NMP_VOICES_UNIQUE_PHRASES_ONLY (NMP_VOICES_MATCHER_BASE +    2)
```

Remarks

The default value for this parameter is NTrue [1064]. It is recommended to use this value. If NFalse [1041] is used false rejection rate (FRR) is higher.

Also it is recommended that each person's phrase is unique throughout the whole database of users.

7.1.1.10.4 Files

7.1.1.10.4.1 NMatcher.h

Header file for the NMatcher [171]. Provides functionality for comparing Neurotechnology Templates (NTemplates), encapsulated in Neurotechnology Matcher (NMatcher [171]) object.

Functions

	Name	Description
	NMCreate [174]	Creates a NMatcher [171]. Created object can be used for matching NTemplate [219].
	NMIdentifyEnd [177]	Ends identification using the specified NMatcher [171].
	NMIdentifyNext [177]	Compares the specified packed NTemplate [219] with the one identification was started with using the specified NMatcher [171].
	NMIdentifyNextEx [175]	Compares the specified packed NTemplate [219] with the one identification was started with using the specified NMatcher [171].
	NMIdentifyStart [178]	Starts identification with the specified packed NTemplate [219] using the specified NMatcher [171].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NMIdentifyStartEx [176]	Starts identification with the specified packed NTemplate [219] using the specified NMatcher [171].
	NMPrepareTemplate [179]	Changes specified template for faster matching.
	NMVerify [181]	Compares two packed NTemplate [219] using the specified NMatcher [171].
	NMVerifyEx [179]	Compares two packed NTemplate [219] using the specified NMatcher [171].

Module

NMatcher Unit [171]

7.1.1.10.4.2 NMatcherParams.h

Header file for the NMatcher [171].

Enumerations

	Name	Description
	NMatchingSpeed_ [184]	Specifies biometric templates matching speed.
	NMFusionType_ [184]	Specifies the fingerprints and faces matching fusion.
	NMatchingSpeed [184]	Specifies biometric templates matching speed.
	NMFusionType [184]	Specifies the fingerprints and faces matching fusion.

Macros

Name	Description
NM_FINGERS_MODE_ATMEL_FINGERCHIP [188]	Represents Atmel Finger Chip scanner. If this fingerprint scanner is used, it is recommended to set this identifier.
NM_FINGERS_MODE_AUTHENTEC_AES2501B [189]	Represents Authentec Aes 2501B scanner.
NM_FINGERS_MODE_AUTHENTEC_AES4000 [189]	Represents Authentec Aes 4000 scanner.
NM_FINGERS_MODE_AUTHENTEC_AFS2 [189]	Represents Authentec AFS2 scanner.
NM_FINGERS_MODE_BIOMETRIKA_FX2000 [189]	Represents Biometrika FX2000 scanner.
NM_FINGERS_MODE_BIOMETRIKA_FX3000 [189]	Represents Biometrika FX3000 scanner.
NM_FINGERS_MODE_BMF_BLP100 [189]	Represents Bmf Blp 100 scanner.
NM_FINGERS_MODE_CROSSMATCH_VERIFIER300 [189]	Represents Cross Match Verifier 300 scanner.
NM_FINGERS_MODE_DIGENT_IZZIX [189]	Represents Digent Izzix scanner.
NM_FINGERS_MODE_DIGITALPERSONA_UAREU [189]	Represents Digital Persona U are U scanner.
NM_FINGERS_MODE_ETHENTICA [190]	Represents Ethentica scanner.
NM_FINGERS_MODE_FUJITSU_MBF200 [190]	Represents Fujitsu Mbf200 scanner.
NM_FINGERS_MODE_FUTRONIC_FS80 [190]	Represents Futronic FS80 scanner. If this fingerprint scanner is used, it is recommended to set this identifier.
NM_FINGERS_MODE_GENERAL [190]	Represents general scanner.
NM_FINGERS_MODE_IDENTICATORTECHNOLOGY_DF90 [190]	Represents Identicator Technology DF90 scanner.
NM_FINGERS_MODE_IDENTIX_DFR2090 [190]	Represents Identix Dfr2090 scanner.
NM_FINGERS_MODE_IDENTIX_TOUCHVIEW [190]	Represents Identix TouchView scanner.
NM_FINGERS_MODE_KEYTRONIC_SECUREDESKTOP [190]	Represents Keytronic Secure Desktop scanner.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NM_FINGERS_MODE_LIGHTUNING_LTTC500 [190]	Represents Light Tuning LttC500 scanner.
NM_FINGERS_MODE_NITGEN_FINGKEY_HAMSTER [191]	Represents Nitgen Fingkey Hamster scanner.
NM_FINGERS_MODE_PRECISEBIOMETRICS_100CS [191]	Represents Precise Biometrics 100CS scanner.
NM_FINGERS_MODE_SECUGEN_HAMSTER [191]	Represents Secugen Hamster scanner.
NM_FINGERS_MODE_STARTEK_FM200 [191]	Represents Startek FM200 scanner.
NM_FINGERS_MODE_TACOMA_CMOS [191]	Represents Tacoma Cmos scanner.
NM_FINGERS_MODE_TESTECH_BIOI [191]	Represents Testech Bio I scanner.
NM_FINGERS_MODE_UPEK_TOUCHCHIP [191]	Represents Upek Touch Chip scanner.
NM_PART_FACES [191]	Identifier specifying that retrieving or setting parameter for faces matcher (NlsMatcher [143]) contained in NMatcher [171].
NM_PART_FINGERS [192]	Identifier specifying that retrieving or setting parameter for fingers matcher (NfsMatcher) contained in NMatcher [171].
NM_PART_IRISES [192]	Identifier specifying that retrieving or setting parameter for irises matcher (NesMatcher) contained in NMatcher [171].
NM_PART_NONE [192]	Identifier specifying that retrieving or setting parameter for NMatcher [171].
NM_PART_PALMS [192]	Identifier specifying that retrieving or setting parameter for palmprints matcher (NfsMatcher) contained in NMatcher [171].
NMP_FACES_MATCHER_BASE [192]	Helper parameter for internal usage.
NMP_FACES_MATCHING_SPEED [192]	Specifies face templates matching speed. Can be Low or High matching speed.
NMP_FACES_MATCHING_THRESHOLD [192]	Matters only if NMP_FUSION_TYPE [194] is NMFusionType.nmftSelectByFaceThenFuse set.
NMP_FACES_MATCHING_THRESHOLD_NEW [193]	Defines fingerprints matching threshold.
NMP_FINGERS_FILL_FULL_MATCHING_DETAILS [193]	This parameter is used when full matching details (mated minutiae) are required to be written to NMMatchDetails [185].
NMP_FINGERS_MATCHER_BASE [193]	Helper parameter for internal usage.
NMP_FINGERS_MATCHING_SPEED [193]	Specifies finger templates matching speed. Can be Low, Medium or High matching speed.
NMP_FINGERS_MAXIMAL_ROTATION [193]	Identifier specifying modulus of maximal rotation allowed between two matched NFRecords parameter of type N_TYPE_BYTE [576]. Parameter value is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).
NMP_FINGERS_MIN_MATCHED_COUNT [193]	Identifier specifying minimal number of fingerprints from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.
NMP_FINGERS_MIN_MATCHED_COUNT_THRESHOLD [193]	Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to NMP_FINGERS_MIN_MATCHED_COUNT [193] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
NMP_FINGERS_MODE [194]	Identifier specifying mode (parameter value set) parameter of type N_TYPE_UINT [578]. Parameter value can be one of the NM_FINGERS_MODE_XXX.
NMP_FUSION_TYPE [194]	Identifier specifies fingerprints, irises and faces fusion. Parameter type is NInt [1006].
NMP_IRISES_MATCHER_BASE [194]	Helper parameter for internal usage.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMP_IRISES_MATCHING_SPEED [194]	Specifies iris templates matching speed. Can be Low, Medium or High matching speed. Parameter type is N_TYPE_INT [577].
NMP_IRISES_MATCHING_THRESHOLD [194]	Iris templates matching threshold. Matters only if NMP_FUSION_TYPE [194] is NMFusionType.nmftSelectByIrisThenFuse set. Parameter type is N_TYPE_INT [577].
NMP_IRISES_MAXIMAL_ROTATION [194]	Defines the maximal rotation allowed between two matched irises. It is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees). Parameter type is NByte [1005].
NMP_IRISES_MIN_MATCHED_COUNT [194]	Identifier specifying minimal number of irises from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.
NMP_IRISES_MIN_MATCHED_COUNT_THRESHOLD [195]	Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to NMP_IRISES_MIN_MATCHED_COUNT [194] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
NMP_MATCHING_THRESHOLD [195]	Identifier specifying matching threshold (biggest allowed FAR) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
NMP_PALMS_FILL_FULL_MATCHING_DETAILS [195]	This parameter is used when full matching details are required to be written to NMMatchDetails [185].
NMP_PALMS_MATCHER_BASE [195]	Helper parameter for internal usage.
NMP_PALMS_MATCHING_SPEED [195]	Specifies plam templates matching speed. Can be Low, Medium or High matching speed.
NMP_PALMS_MAXIMAL_ROTATION [195]	Identifier specifying modulus of maximal rotation allowed between two matched NFRecords parameter of type N_TYPE_BYTE [576]. Parameter value is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).
NMP_PALMS_MIN_MATCHED_COUNT [196]	Identifier specifying minimal number of palmprints from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.
NMP_PALMS_MIN_MATCHED_COUNT_THRESHOLD [196]	Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to NMP_FINGERS_MIN_MATCHED_COUNT [193] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
NMP_VOICES_MATCHER_BASE [196]	Helper parameter for internal usage.
NMP_VOICES_UNIQUE_PHRASES_ONLY [196]	Identifier specifying the usage of unique phrases. If it was set to NTrue [1064] user during verification can say only his unique phrase(-s) (which was enrolled to database). If set to NFalse [1041] user can say any phrase.

Module

NMatcher Unit [171]

7.1.1.10.4.3 NMMatchDetails.h

Header file for the NMatcher [171].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NMMatchDetailsDeserialize [180]	Deserializes NMMatchDetails [185].
	NMMatchDetailsDeserializeFromStream [180]	Deserializes NMMatchDetails [185] contained by the specified stream.
	NMMatchDetailsFree [181]	Deletes NMMatchDetails [185].
	NMMatchDetailsSerialize [182]	Serializes NMMatchDetails [185].
	NMMatchDetailsSerializeToStream [183]	Serializes NMMatchDetails [185] object to memory stream.

Module

NMatcher Unit [171]

Structures

	Name	Description
	NMMatchDetails_ [185]	Represents details of matching performed with NMMatcher.
	NMMatchDetails [185]	Represents details of matching performed with NMMatcher.

7.1.1.11 NMatchers Unit

Provides functionality for retrieving information about NMatchers library.

Files

Name	Description
NMatchers.h [200]	Header file for the NMatchers library. Provides functions for retrieving NMatchers library information.

7.1.1.11.1 Files

7.1.1.11.1.1 NMatchers.h

Header file for the NMatchers library. Provides functions for retrieving NMatchers library information.

Module

NMatchers Unit [200]

7.1.1.12 NMatchingDetails Unit

Provides functionality to retrieve Neurotechnology templates matching details (such as score, records count, etc.).

Files

Name	Description
NMatchingDetails.h [218]	Header file for the NMatchingDetails.

Functions

	Name	Description
	NMatchingDetailsCreateFromMemory [201]	Creates NMatchingDetails from specified memory buffer.
	NMatchingDetailsCreateFromMemoryN [202]	Creates NMatchingDetails from specified memory buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NMatchingDetailsCreateFromStream [202]	Creates NMatchingDetails from specified stream.
	NMatchingDetailsGetFace [203]	Gets face matching details (NLMatchingDetails [134]) from specified NMatchingDetails.
	NMatchingDetailsGetFaceCount [203]	Gets face templates count in NMatchingDetails.
	NMatchingDetailsGetFaceEx [204]	Gets face matching details (NLMatchingDetails [134]) from specified NMatchingDetails.
	NMatchingDetailsGetFacesMatchedIndex [204]	Gets index of matched face in the second matched NLTemplate [153] (or NTemplate [219]))
	NMatchingDetailsGetFacesScore [205]	Gets similarity score of two matched face templates (NLTemplate [153]).
	NMatchingDetailsGetFinger [206]	Gets fingerprint matching details (NFMatchingDetails) from specified NMatchingDetails.
	NMatchingDetailsGetFingerCount [206]	Gets fingerprint templates count in NMatchingDetails.
	NMatchingDetailsGetFingerEx [207]	Gets fingerprint matching details (NFMatchingDetails) from specified NMatchingDetails.
	NMatchingDetailsGetFingersScore [207]	Gets similarity score of two matched fingerprint templates (NFTemplate).
	NMatchingDetailsGetId [208]	Retrieves an ID of the specified NMatchingDetails object.
	NMatchingDetailsGetIris [209]	Gets iris matching details (NEMatchingDetails) from specified NMatchingDetails.
	NMatchingDetailsGetIrisCount [209]	Gets iris templates count in NMatchingDetails.
	NMatchingDetailsGetIrisesScore [210]	Gets similarity score of two matched iris templates (NETemplate).
	NMatchingDetailsGetIrisEx [211]	Gets iris matching details (NEMatchingDetails) from specified NMatchingDetails.
	NMatchingDetailsGetPalm [211]	Gets palmprint matching details (NFMatchingDetails) from specified NMatchingDetails.
	NMatchingDetailsGetPalmCount [212]	Gets palmprint templates count in NMatchingDetails.
	NMatchingDetailsGetPalmEx [212]	Gets palmprint matching details (NFMatchingDetails) from specified NMatchingDetails.
	NMatchingDetailsGetPalmsScore [213]	Gets similarity score of two matched palmprint templates (NFTemplate).
	NMatchingDetailsGetVoice [214]	Gets voice matching details (NSMatchingDetails) from specified NMatchingDetails.
	NMatchingDetailsGetVoiceCount [215]	Gets voice templates count in NMatchingDetails.
	NMatchingDetailsGetVoiceEx [215]	Gets voice matching details (NSMatchingDetails) from specified NMatchingDetails.
	NMatchingDetailsGetVoicesScore [215]	Gets similarity score of two matched voice templates (NSTemplate).
	NMatchingDetailsSaveToMemory [217]	Packs the NMatchingDetailsBase into memory buffer.

Macros

Name	Description
NMatchingDetailsSaveToStream [217]	Saves NMatchingDetailsBase to the memory stream.

7.1.1.12.1 Functions

7.1.1.12.1.1 NMatchingDetailsCreateFromMemory Function

Creates NMatchingDetails [200] from specified memory buffer.

C++

```
NResult N_API NMatchingDetailsCreateFromMemory(const void * pBuffer, NSizeType bufferSize,
NUInt flags, NSizeType * pSize, HNMatchingDetails * phMatchingDetails);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains NMatchingDetails [200].
NSizeType bufferSize	[in] Size of memory buffer *pBuffer*.
NUInt flags	[in] Must be 0 (reserved for future use).
NSizeType * pSize	[out] Actual size of data read from pBuffer (bytes).
HNMatchingDetails * phMatchingDetails	[out] Pointer to HNMatchingDetails which received handle to newly created NMatchingDetails [200] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMatchingDetails Unit [200]

7.1.1.12.1.2 NMatchingDetailsCreateFromMemoryN Function

Creates NMatchingDetails [200] from specified memory buffer.

C++

```
NResult N_API NMatchingDetailsCreateFromMemoryN(HNBuffer hBuffer, NUInt flags, NSizeType *
pSize, HNMatchingDetails * phMatchingDetails);
```

Parameters

Parameters	Description
HNBuffer hBuffer	[in] Handle to memory buffer that contains NMatchingDetails [200].
NUInt flags	[in] Must be 0 (reserved for future use).
NSizeType * pSize	[out] Actual size of data read from hBuffer (bytes).
HNMatchingDetails * phMatchingDetails	[out] Pointer to HNMatchingDetails which received handle to newly created NMatchingDetails [200] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMatchingDetails Unit [200]

7.1.1.12.1.3 NMatchingDetailsCreateFromStream Function

Creates NMatchingDetails [200] from specified stream.

C++

```
NResult N_API NMatchingDetailsCreateFromStream(HNStream hStream, NUInt flags,
HNMatchingDetails * phMatchingDetails);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to memory stream which contains matching details.
NUInt flags	[in] Must be 0 (reserved for future use).
HNMatchingDetails * phMatchingDetails	[out] Pointer to HNMatchingDetails which received handle to newly created NMatchingDetails [200] object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NMatchingDetailsCreateFromMemory [201]

Module

NMatchingDetails Unit [200]

7.1.1.12.1.4 NMatchingDetailsGetFace Function

Gets face matching details (NLMatchingDetails [134]) from specified NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetFace(HNMatchingDetails hMatchingDetails, NInt index,
HNLMatchingDetails * phValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt index	[in] Face index in NMatchingDetails [200] (NLMatchingDetails [134] will be returned for face template specified by index). Index should be greater than 0 and less than face count in NMatchingDetails [200]. Face count can be retrieved by calling NMatchingDetailsGetFaceCount [203] function.
HNLMatchingDetails * phValue	[out] Pointer to HNLMatchingDetails which contains information about face templates matching (faces match result).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NMatchingDetailsGetFaceCount [203]

NMatchingDetailsGetFacesScore [205]

Module

NMatchingDetails Unit [200]

7.1.1.12.1.5 NMatchingDetailsGetFaceCount Function

Gets face templates count in NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetFaceCount(HNMatchingDetails hMatchingDetails, NInt *
pValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt * pValue	[out] Faces [45] count in NMatchingDetails [200].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

See Also

NMatchingDetailsGetFacesScore [205]

Module

NMatchingDetails Unit [200]

7.1.1.12.1.6 NMatchingDetailsGetFaceEx Function

Gets face matching details (NLMatchingDetails [134]) from specified NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetFaceEx(HNMatchingDetails hMatchingDetails, NInt index,
HNLMatchingDetails * phValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt index	[in] Face index in NMatchingDetails [200] (NLMatchingDetails [134] will be returned for face template specified by index). Index should be greater than 0 and less than face count in NMatchingDetails [200]. Face count can be retrieved by calling NMatchingDetailsGetFaceCount [203] function.
HNLMatchingDetails * phValue	[out] Pointer to HNLMatchingDetails which contains information about face templates matching (faces match result).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMatchingDetails Unit [200]

7.1.1.12.1.7 NMatchingDetailsGetFacesMatchedIndex Function

Gets index of matched face in the second matched NLTemplate [153] (or NTemplate [219]))

C++

```
NResult N_API NMatchingDetailsGetFacesMatchedIndex(HNMatchingDetails hMatchingDetails, NInt
* pValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt * pValue	[out] Index of matched face in the second template.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NMatchingDetailsGetFacesScore [205]

Module

NMatchingDetails Unit [200]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.12.1.8 NMatchingDetailsGetFacesScore Function

Gets similarity score of two matched face templates (NLTemplate [153]).

C++

```
NResult N_API NMatchingDetailsGetFacesScore(HNMatchingDetails hMatchingDetails, NInt *
pValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt * pValue	[out] Faces [45] similarity score.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Example

Function for printing matching details (face index and score):

```
//Prints faces similarity score and index from specified handle
static NResult PrintMatchingDetails(HNMatchingDetails hMatchingDetails)
{
  NResult nr;
  NBiometricType biometricType;
  NInt i, count, index, score;

  //Retrieves biometric type of specified NMatchingDetails (which template type matching
details are saved)
  nr = NMatchingDetailsBaseGetBiometricType(hMatchingDetails, &biometricType);
  //If NMatchingDetails type detection failed an error code is returned. This value is less
than 0.
  //To find more about error codes, read NResult Type
  if (NFailed(nr)) return nr;
  //Checks if NMatchingDetails saves data about faces matching
  if ((biometricType & nbtFace) == nbtFace)
  {
    //Trying to retrieve face templates similarity score
    nr = NMatchingDetailsGetFacesScore(hMatchingDetails, &score);
    //If failed to retrieve similarity score, an error code is returned
    if (NFailed(nr)) return nr;
    //Prints on a screen similarity score
    printf(N_T("    Faces match details:"));
    printf(N_T(" score = %d\n"), score);

    //Retrieving face count in specified NMatchingDetails
    nr = NMatchingDetailsGetFaceCount(hMatchingDetails, &count);
    if (NFailed(nr)) return nr;
    //Trying to get index of second matched face template
    for (i = 0; i < count; i++)
    {
      HNLMatchingDetails hFaceMatchingDetails;
      //Gets face matching details (NLMatchingDetails)
      nr = NMatchingDetailsGetFace(hMatchingDetails, i, &hFaceMatchingDetails);
      if (NFailed(nr)) return nr;
      //Using face matching details gets index
      nr = NXMatchingDetailsGetMatchedIndex(hFaceMatchingDetails, &index);
      if (NFailed(nr)) return nr;
      //Gets faces similarity score by calling base function for retrieving similarity score
      //Note: If handle to NMatchingDetails is specified, similarity score for all
modalities is returned
      nr = NMatchingDetailsBaseGetScore(hFaceMatchingDetails, &score);
      if (NFailed(nr)) return nr;
      printf(N_T("    face index: %d; score: %d;\n"), index, score);
    }
  }
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
    return nr;
}
```

Note: This function should be called after templates identification (NMIdentifyNextEx [175] and NMIdentifyNextEx [175]) or verification (NMVerifyEx [179]).

Module

NMatchingDetails Unit [200]

7.1.1.12.1.9 NMatchingDetailsGetFinger Function

Gets fingerprint matching details (NFMatchingDetails) from specified NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetFinger(HNMatchingDetails hMatchingDetails, NInt index,
HNFMatchingDetails * phValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt index	[in] Fingerprint index in NMatchingDetails [200] (NFMatchingDetails will be returned for fingerprint template specified by index). Index should be greater than 0 and less than face count in NMatchingDetails [200]. Fingerprint count can be retrieved by calling NMatchingDetailsGetFingerCount [206] function.
HNFMatchingDetails * phValue	[out] Pointer to HNFMatchingDetails which contains information about fingerprint templates matching (fingerprints match result).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NMatchingDetailsGetFingerCount [206]

NMatchingDetailsGetFingerScore

Module

NMatchingDetails Unit [200]

7.1.1.12.1.10 NMatchingDetailsGetFingerCount Function

Gets fingerprint templates count in NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetFingerCount(HNMatchingDetails hMatchingDetails, NInt *
pValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt * pValue	[out] Fingerprint count in NMatchingDetails [200].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

See Also

NMatchingDetailsGetFingerScore

Module

NMatchingDetails Unit [200]

7.1.1.12.1.11 NMatchingDetailsGetFingerEx Function

Gets fingerprint matching details (NFMatchingDetails) from specified NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetFingerEx(HNMatchingDetails hMatchingDetails, NInt index,
HNFMatchingDetails * phValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt index	[in] Fingerprint index in NMatchingDetails [200] (NFMatchingDetails will be returned for fingerprint template specified by index). Index should be greater than 0 and less than face count in NMatchingDetails [200]. Fingerprint count can be retrieved by calling NMatchingDetailsGetFingerCount [206] function.
HNFMatchingDetails * phValue	[out] Pointer to HNFMatchingDetails which contains information about fingerprint templates matching (fingerprints match result).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMatchingDetails Unit [200]

7.1.1.12.1.12 NMatchingDetailsGetFingersScore Function

Gets similarity score of two matched fingerprint templates (NFTemplate).

C++

```
NResult N_API NMatchingDetailsGetFingersScore(HNMatchingDetails hMatchingDetails, NInt *
pValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt * pValue	[out] Fingerprints similarity score.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Example

Function for printing matching details (fingerprint index and score):

```
//Prints fingerprints similarity score and index from specified handle
static NResult PrintMatchingDetails(HNMatchingDetails hMatchingDetails)
{
  NResult nr;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
  NBiometricType biometricType;
  NInt i, count, index, score;

  //Retrieves biometric type of specified NMatchingDetails (which template type matching
details are saved)
  nr = NMatchingDetailsBaseGetBiometricType(hMatchingDetails, &biometricType);
  //If NMatchingDetails type detection failed an error code is returned. This value is less
than 0.
  //To find more about error codes, read NResult Type
  if (NFailed(nr)) return nr;
  //Checks if NMatchingDetails saves data about fingerprints matching
  if ((biometricType & nbtFinger) == nbtFinger)
  {
    //Trying to retrieve fingerprint templates similarity score
    nr = NMatchingDetailsGetFingersScore(hMatchingDetails, &score);
    //If failed to retrieve similarity score, an error code is returned
    if (NFailed(nr)) return nr;
    //Prints on a screen similarity score
    printf(N_T("    Fingerprint match details:"));
    printf(N_T(" score = %d\n"), score);

    //Retrieving fingerprint count in specified NMatchingDetails
    nr = NMatchingDetailsGetFingerCount(hMatchingDetails, &count);
    if (NFailed(nr)) return nr;
    //Trying to get index of second matched fingerprint template
    for (i = 0; i < count; i++)
    {
      HNFMatchingDetails hFingerMatchingDetails;
      //Gets face matching details (NLMatchingDetails)
      nr = NMatchingDetailsGetFinger(hMatchingDetails, i, &hFingerMatchingDetails);
      if (NFailed(nr)) return nr;
      //Using fingerprint matching details gets index
      nr = NXMatchingDetailsGetMatchedIndex(hFingerMatchingDetails, &index);
      if (NFailed(nr)) return nr;
      //Gets fingerprints similarity score by calling base function for retrieving
similarity score
      //Note: If handle to NMatchingDetails is specified, similarity score for all
modalities is returned
      nr = NMatchingDetailsBaseGetScore(hFingerMatchingDetails, &score);
      if (NFailed(nr)) return nr;
      printf(N_T("    fingerprint index: %d; score: %d;\n"), index, score);
    }
  }

  return nr;
}
```

Note: This function should be called after templates identification (`NMIdentifyNextEx` [175] and `NMIdentifyNextEx` [175]) or verification (`NMVerifyEx` [179]).

Module

NMatchingDetails Unit [200]

7.1.1.12.1.13 NMatchingDetailsGetId Function

Retrieves an ID of the specified NMatchingDetails [200] object.

C++

```
NResult N_API NMatchingDetailsGetId(HNMatchingDetails hMatchingDetails, const void * *
pValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to the NMatchingDetails [200].
const void * * pValue	[in] Pointer to the ID of the NMatchingDetails [200].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMatchingDetails Unit [200]

7.1.1.12.1.14 NMatchingDetailsGetIris Function

Gets iris matching details (NEMatchingDetails) from specified NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetIris(HNMatchingDetails hMatchingDetails, NInt index,
HNEMatchingDetails * phValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt index	[in] Iris index in NMatchingDetails [200] (NEMatchingDetails will be returned for iris template specified by index). Index should be greater than 0 and less than iris count in NMatchingDetails [200]. Iris count can be retrieved by calling NMatchingDetailsGetIrisCount [209] function.
HNEMatchingDetails * phValue	[out] Pointer to HNEMatchingDetails which contains information about iris templates matching (irises match result).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NMatchingDetailsGetIrisCount [209]

NMatchingDetailsGetIrisesScore [210]

Module

NMatchingDetails Unit [200]

7.1.1.12.1.15 NMatchingDetailsGetIrisCount Function

Gets iris templates count in NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetIrisCount(HNMatchingDetails hMatchingDetails, NInt *
pValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt * pValue	[out] Irises count in NMatchingDetails [200].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NMatchingDetailsGetIrisesScore [210]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NMatchingDetails Unit [200]

7.1.1.12.1.16 NMatchingDetailsGetIrisesScore Function

Gets similarity score of two matched iris templates (NETemplate).

C++

```
NResult N_API NMatchingDetailsGetIrisesScore(HNMatchingDetails hMatchingDetails, NInt *
pValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt * pValue	[out] Irises similarity score.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Example

Function for printing matching details (iris index and score):

```
//Prints irises similarity score and index from specified handle
static NResult PrintMatchingDetails(HNMatchingDetails hMatchingDetails)
{
  NResult nr;
  NBiometricType biometricType;
  NInt i, count, index, score;

  //Retrieves biometric type of specified NMatchingDetails (which template type matching
details are saved)
  nr = NMatchingDetailsBaseGetBiometricType(hMatchingDetails, &biometricType);
  //If NMatchingDetails type detection failed an error code is returned. This value is less
than 0.
  //To find more about error codes, read NResult Type
  if (NFailed(nr)) return nr;
  //Checks if NMatchingDetails saves data about irises matching
  if ((biometricType & nbtIris) == nbtIris)
  {
    //Trying to retrieve iris templates similarity score
    nr = NMatchingDetailsGetIrisesScore(hMatchingDetails, &score);
    //If failed to retrieve similarity score, an error code is returned
    if (NFailed(nr)) return nr;
    //Prints on a screen similarity score
    printf(N_T("    Irises match details:"));
    printf(N_T(" score = %d\n"), score);

    //Retrieving irises count in specified NMatchingDetails
    nr = NMatchingDetailsGetIrisCount(hMatchingDetails, &count);
    if (NFailed(nr)) return nr;
    //Trying to get index of second matched iris template
    for (i = 0; i < count; i++)
     {
      HNEMatchingDetails hIrisMatchingDetails;
      //Gets iris matching details (NEMatchingDetails)
      nr = NMatchingDetailsGetIris(hMatchingDetails, i, &hIrisMatchingDetails);
      if (NFailed(nr)) return nr;
      //Using iris matching details gets index
      nr = NXMatchingDetailsGetMatchedIndex(hIrisMatchingDetails, &index);
      if (NFailed(nr)) return nr;
      //Gets irises similarity score by calling base function for retrieving similarity
score
      //Note: If handle to NMatchingDetails is specified, similarity score for all
modalities is returned
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
        nr = NMatchingDetailsBaseGetScore(hIrisMatchingDetails, &score);
        if (NFailed(nr)) return nr;
        printf(N_T("    iris index: %d; score: %d;\n"), index, score);
      }
    }

    return nr;
  }
```

Note: This function should be called after templates identification (`NMIdentifyNextEx` [175] and `NMIdentifyNextEx` [175]) or verification (`NMVerifyEx` [179]).

Module

NMatchingDetails Unit [200]

7.1.1.12.1.17 NMatchingDetailsGetIrisEx Function

Gets iris matching details (NEMatchingDetails) from specified NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetIrisEx(HNMatchingDetails hMatchingDetails, NInt index,
HNEMatchingDetails * phValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt index	[in] Iris index in NMatchingDetails [200] (NEMatchingDetails will be returned for iris template specified by index). Index should be greater than 0 and less than iris count in NMatchingDetails [200]. Iris count can be retrieved by calling NMatchingDetailsGetIrisCount [209] function.
HNEMatchingDetails * phValue	[out] Pointer to HNEMatchingDetails which contains information about iris templates matching (irises match result).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMatchingDetails Unit [200]

7.1.1.12.1.18 NMatchingDetailsGetPalm Function

Gets palmprint matching details (NFMatchingDetails) from specified NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetPalm(HNMatchingDetails hMatchingDetails, NInt index,
HNFMatchingDetails * phValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt index	[in] Palmprint index in NMatchingDetails [200] (NFMatchingDetails will be returned for palmprint template specified by index). Index should be greater than 0 and less than palmprints count in NMatchingDetails [200]. Palmprints count can be retrieved by calling NMatchingDetailsGetPalmCount [212] function.
HNFMatchingDetails * phValue	[out] Pointer to HNFMatchingDetails which contains information about palmprint templates matching (palmprints match result).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NMatchingDetailsGetPalmCount [212]

NMatchingDetailsGetPalmsScore [213]

Module

NMatchingDetails Unit [200]

7.1.1.12.1.19 NMatchingDetailsGetPalmCount Function

Gets palmprint templates count in NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetPalmCount(HNMatchingDetails hMatchingDetails, NInt *
pValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt * pValue	[out] Palmprints count in NMatchingDetails [200].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NMatchingDetailsGetPalmsScore [213]

Module

NMatchingDetails Unit [200]

7.1.1.12.1.20 NMatchingDetailsGetPalmEx Function

Gets palmprint matching details (NFMatchingDetails) from specified NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetPalmEx(HNMatchingDetails hMatchingDetails, NInt index,
HNFMatchingDetails * phValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt index	[in] Palmprint index in NMatchingDetails [200] (NFMatchingDetails will be returned for palmprint template specified by index). Index should be greater than 0 and less than palmprints count in NMatchingDetails [200]. Palmprints count can be retrieved by calling NMatchingDetailsGetPalmCount [212] function.
HNFMatchingDetails * phValue	[out] Pointer to HNFMatchingDetails which contains information about palmprint templates matching (palmprints match result).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMatchingDetails Unit [200]

7.1.1.12.1.21 NMatchingDetailsGetPalmsScore Function

Gets similarity score of two matched palmprint templates (NFTemplate).

C++

```
NResult N_API NMatchingDetailsGetPalmsScore(HNMatchingDetails hMatchingDetails, NInt *
pValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt * pValue	[out] Palmprints similarity score.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Example

Function for printing matching details (palmprint index and score):

```
//Prints palmprints similarity score and index from specified handle
static NResult PrintMatchingDetails(HNMatchingDetails hMatchingDetails)
{
  NResult nr;
  NBiometricType biometricType;
  NInt i, count, index, score;

  //Retrieves biometric type of specified NMatchingDetails (which template type matching
details are saved)
  nr = NMatchingDetailsBaseGetBiometricType(hMatchingDetails, &biometricType);
  //If NMatchingDetails type detection failed an error code is returned. This value is less
than 0.
  //To find more about error codes, read NResult Type
  if (NFailed(nr)) return nr;
  //Checks if NMatchingDetails saves data about palmprints matching
  if ((biometricType & nbtPalm) == nbtPalm)
  {
    //Trying to retrieve palmprint templates similarity score
    nr = NMatchingDetailsGetPalmsScore(hMatchingDetails, &score);
    //If failed to retrieve similarity score, an error code is returned
    if (NFailed(nr)) return nr;
    //Prints on a screen similarity score
    printf(N_T("    Palms match details:"));
    printf(N_T(" score = %d\n"), score);

    //Retrieving palmprints count in specified NMatchingDetails
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
    nr = NMatchingDetailsGetPalmCount(hMatchingDetails, &count);
    if (NFailed(nr)) return nr;
    //Trying to get index of second matched palmprint template
    for (i = 0; i < count; i++)
     {
      HNFMatchingDetails hPalmMatchingDetails;
      //Gets palmprint matching details (NFMatchingDetails)
      nr = NMatchingDetailsGetPalm(hMatchingDetails, i, &hPalmMatchingDetails);
      if (NFailed(nr)) return nr;
      //Using palmprint matching details gets index
      nr = NXMatchingDetailsGetMatchedIndex(hPalmMatchingDetails, &index);
      if (NFailed(nr)) return nr;
      //Gets palmprints similarity score by calling base function for retrieving similarity
score
      //Note: If handle to NMatchingDetails is specified, similarity score for all
modalities is returned
      nr = NMatchingDetailsBaseGetScore(hPalmMatchingDetails, &score);
      if (NFailed(nr)) return nr;
      printf(N_T("    palm index: %d; score: %d;\n"), index, score);
    }
  }

  return nr;
}
```

Note: This function should be called after templates identification (NMIdentifyNextEx [175] and NMIdentifyNextEx [175]) or verification (NMVerifyEx [179]).

Module

NMatchingDetails Unit [200]

7.1.1.12.1.22 NMatchingDetailsGetVoice Function

Gets voice matching details (NSMatchingDetails) from specified NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetVoice(HNMatchingDetails hMatchingDetails, NInt index,
HNSMatchingDetails * phValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt index	[in] Voice template index in NMatchingDetails [200] (NSMatchingDetails will be returned for voice template specified by index). Index should be greater than 0 and less than voice templates count in NMatchingDetails [200]. Voices count can be retrieved by calling NMatchingDetailsGetVoiceCount [215] function.
pValue	[out] Pointer to HNSMatchingDetails which contains information about voice templates matching (voices match result).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NMatchingDetailsGetVoiceCount [215]

NMatchingDetailsGetVoicesScore [215]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NMatchingDetails Unit [200]

7.1.1.12.1.23 NMatchingDetailsGetVoiceCount Function

Gets voice templates count in NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetVoiceCount(HNMatchingDetails hMatchingDetails, NInt *
pValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt * pValue	[out] Voice count in NMatchingDetails [200].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NMatchingDetailsGetVoicesScore [215]

Module

NMatchingDetails Unit [200]

7.1.1.12.1.24 NMatchingDetailsGetVoiceEx Function

Gets voice matching details (NSMatchingDetails) from specified NMatchingDetails [200].

C++

```
NResult N_API NMatchingDetailsGetVoiceEx(HNMatchingDetails hMatchingDetails, NInt index,
HNSMatchingDetails * phValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt index	[in] Voice template index in NMatchingDetails [200] (NSMatchingDetails will be returned for voice template specified by index). Index should be greater than 0 and less than voice templates count in NMatchingDetails [200]. Voices count can be retrieved by calling NMatchingDetailsGetVoiceCount [215] function.
HNSMatchingDetails * phValue	[out] Pointer to HNSMatchingDetails which contains information about voice templates matching (voices match result).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMatchingDetails Unit [200]

7.1.1.12.1.25 NMatchingDetailsGetVoicesScore Function

Gets similarity score of two matched voice templates (NSTemplate).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NMatchingDetailsGetVoicesScore(HNMatchingDetails hMatchingDetails, NInt *
pValue);
```

Parameters

Parameters	Description
HNMatchingDetails hMatchingDetails	[in] Handle to NMatchingDetails [200].
NInt * pValue	[out] Voices similarity score.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Example

Function for printing matching details (voice index and score):

```
//Prints voices similarity score and index from specified handle
static NResult PrintMatchingDetails(HNMatchingDetails hMatchingDetails)
{
  NResult nr;
  NBiometricType biometricType;
  NInt i, count, index, score;

  //Retrieves biometric type of specified NMatchingDetails (which template type matching
details are saved)
  nr = NMatchingDetailsBaseGetBiometricType(hMatchingDetails, &biometricType);
  //If NMatchingDetails type detection failed an error code is returned. This value is less
than 0.
  //To find more about error codes, read NResult Type
  if (NFailed(nr)) return nr;
  //Checks if NMatchingDetails saves data about voices matching
  if ((biometricType & nbtVoice) == nbtVoice)
  {
    //Trying to retrieve voice templates similarity score
    nr = NMatchingDetailsGetVoicesScore(hMatchingDetails, &score);
    //If failed to retrieve similarity score, an error code is returned
    if (NFailed(nr)) return nr;
    //Prints on a screen similarity score
    printf(N_T("    Voices match details:"));
    printf(N_T(" score = %d\n"), score);

    //Retrieving voices count in specified NMatchingDetails
    nr = NMatchingDetailsGetVoiceCount(hMatchingDetails, &count);
    if (NFailed(nr)) return nr;
    //Trying to get index of second matched voice template
    for (i = 0; i < count; i++)
     {
      HNSMatchingDetails hVoiceMatchingDetails;
      //Gets voice matching details (NSMatchingDetails)
      nr = NMatchingDetailsGetVoice(hMatchingDetails, i, &hVoiceMatchingDetails);
      if (NFailed(nr)) return nr;
      //Using voice matching details gets index
      nr = NXMatchingDetailsGetMatchedIndex(hVoiceMatchingDetails, &index);
      if (NFailed(nr)) return nr;
      //Gets voices similarity score by calling base function for retrieving similarity
score
      //Note: If handle to NMatchingDetails is specified, similarity score for all
modalities is returned
      nr = NMatchingDetailsBaseGetScore(hVoiceMatchingDetails, &score);
      if (NFailed(nr)) return nr;
      printf(N_T("    voice index: %d; score: %d;\n"), index, score);
    }
  }

  return nr;
}
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Note: This function should be called after templates identification (NMIdentifyNextEx [175] and NMIdentifyNextEx [175]) or verification (NMVerifyEx [179]).

Module

NMatchingDetails Unit [200]

7.1.1.12.1.26 NMatchingDetailsSaveToMemory Function

Packs the NMatchingDetailsBase into memory buffer.

C++

```
NResult N_API NMatchingDetailsSaveToMemory(HNMatchingDetailsBase hMatchingDetails, NUInt flags, void * * ppBuffer, NSizeType * pBufferSize);
```

Parameters

Parameters	Description
HNMatchingDetailsBase hMatchingDetails	[in] Handle to NMatchingDetailsBase.
NUInt flags	[in] Must be 0 (reserved for future use).
void * * ppBuffer	[out] Pointer to allocated memory buffer. In this buffer NMatchingDetailsBase is saved.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer ppBuffer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMatchingDetails Unit [200]

7.1.1.12.2 Macros

7.1.1.12.2.1 NMatchingDetailsSaveToStream Macro

Saves NMatchingDetailsBase to the memory stream.

C++

```
#define NMatchingDetailsSaveToStream(hMatchingDetails, hStream, flags)
NObjectSaveToStream(hMatchingDetails, hStream, flags)
```

Parameters

Parameters	Description
hMatchingDetails	[in] Handle to NMatchingDetailsBase.
hStream	[in] Handle to memory stream where matching details will be saved.
flags	[in] Must be 0 (reserved for future use).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMatchingDetails Unit [200]

7.1.1.12.3 Files

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.12.3.1 NMatchingDetails.h

Header file for the NMatchingDetails [200].

Functions

	Name	Description
	NMatchingDetailsCreateFromMemory [201]	Creates NMatchingDetails [200] from specified memory buffer.
	NMatchingDetailsCreateFromMemoryN [202]	Creates NMatchingDetails [200] from specified memory buffer.
	NMatchingDetailsCreateFromStream [202]	Creates NMatchingDetails [200] from specified stream.
	NMatchingDetailsGetFace [203]	Gets face matching details (NLMatchingDetails [134]) from specified NMatchingDetails [200].
	NMatchingDetailsGetFaceCount [203]	Gets face templates count in NMatchingDetails [200].
	NMatchingDetailsGetFaceEx [204]	Gets face matching details (NLMatchingDetails [134]) from specified NMatchingDetails [200].
	NMatchingDetailsGetFacesMatchedIndex [204]	Gets index of matched face in the second matched NLTemplate [153] (or NTemplate [219]))
	NMatchingDetailsGetFacesScore [205]	Gets similarity score of two matched face templates (NLTemplate [153]).
	NMatchingDetailsGetFinger [206]	Gets fingerprint matching details (NFMatchingDetails) from specified NMatchingDetails [200].
	NMatchingDetailsGetFingerCount [206]	Gets fingerprint templates count in NMatchingDetails [200].
	NMatchingDetailsGetFingerEx [207]	Gets fingerprint matching details (NFMatchingDetails) from specified NMatchingDetails [200].
	NMatchingDetailsGetFingersScore [207]	Gets similarity score of two matched fingerprint templates (NFTemplate).
	NMatchingDetailsGetId [208]	Retrieves an ID of the specified NMatchingDetails [200] object.
	NMatchingDetailsGetIris [209]	Gets iris matching details (NEMatchingDetails) from specified NMatchingDetails [200].
	NMatchingDetailsGetIrisCount [209]	Gets iris templates count in NMatchingDetails [200].
	NMatchingDetailsGetIrisesScore [210]	Gets similarity score of two matched iris templates (NETemplate).
	NMatchingDetailsGetIrisEx [211]	Gets iris matching details (NEMatchingDetails) from specified NMatchingDetails [200].
	NMatchingDetailsGetPalm [211]	Gets palmprint matching details (NFMatchingDetails) from specified NMatchingDetails [200].
	NMatchingDetailsGetPalmCount [212]	Gets palmprint templates count in NMatchingDetails [200].
	NMatchingDetailsGetPalmEx [212]	Gets palmprint matching details (NFMatchingDetails) from specified NMatchingDetails [200].
	NMatchingDetailsGetPalmsScore [213]	Gets similarity score of two matched palmprint templates (NFTemplate).
	NMatchingDetailsGetVoice [214]	Gets voice matching details (NSMatchingDetails) from specified NMatchingDetails [200].
	NMatchingDetailsGetVoiceCount [215]	Gets voice templates count in NMatchingDetails [200].
	NMatchingDetailsGetVoiceEx [215]	Gets voice matching details (NSMatchingDetails) from specified NMatchingDetails [200].
	NMatchingDetailsGetVoicesScore [215]	Gets similarity score of two matched voice templates (NSTemplate).
	NMatchingDetailsSaveToMemory [217]	Packs the NMatchingDetailsBase into memory buffer.

Macros

Name	Description
NMatchingDetailsSaveToStream [217]	Saves NMatchingDetailsBase to the memory stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NMatchingDetails Unit [200]

7.1.1.13 NTemplate Unit

Provides functionality for packing, unpacking and editing Neurotechnology Templates (NTemplates).

Notes

When NTemplate contains palm records fusion is not available.

Files

Name	Description
NTemplate.h [254]	Header file for the NTemplate. Provides functionality for packing, unpacking and editing Neurotechnology Templates (NTemplates).

Functions

	Name	Description
	NTemplateAddFacesCopyEx [221]	Adds a copy of an existing face template to the NTemplate.
	NTemplateAddFacesEx [221]	Adds an empty faces template to the NTemplate.
	NTemplateAddFacesFromMemoryEx [222]	Unpacks a faces template from the specified memory buffer and adds it to the NTemplate.
	NTemplateAddFingersCopyEx [222]	Adds a copy of an existing fingers template to the NTemplate.
	NTemplateAddFingersEx [223]	Adds an empty fingers template to the NTemplate.
	NTemplateAddFingersFromMemoryEx [223]	Unpacks a fingers template from the specified memory buffer and adds it to the NTemplate.
	NTemplateAddIrisesCopyEx [224]	Adds a copy of an existing iris template to the NTemplate.
	NTemplateAddIrisesEx [225]	Adds an empty irises template to the NTemplate.
	NTemplateAddPalmsCopyEx [225]	Adds a copy of an existing palm template to the NTemplate.
	NTemplateAddIrisesFromMemoryEx [226]	Unpacks a irises template from the specified memory buffer and adds it to the NTemplate.
	NTemplateAddPalmsEx [226]	Adds an empty palms template to the NTemplate.
	NTemplateAddVoicesCopyEx [227]	Adds a copy of an existing voice template to the NTemplate.
	NTemplateAddPalmsFromMemoryEx [227]	Unpacks a palms template from the specified memory buffer and adds it to the NTemplate.
	NTemplateAddVoices [228]	Adds an empty voice template to the NTemplate.
	NTemplateCheck [229]	Checks if format of the packed NTemplate is correct.
	NTemplateClear [229]	Removes all templates from the NTemplate.
	NTemplateAddVoicesFromMemory [229]	Unpacks a voice template from the specified memory buffer and adds it to the NTemplate.
	NTemplateCalculateSize [230]	Calculates the size of a packed NTemplate containing fingers, palms, irises or faces templates of the specified size.
	NTemplateGetFacesEx [231]	Retrieves the faces template of the NTemplate.
	NTemplateCreateEx [232]	Creates an empty NTemplate with specified flags.
	NTemplateCreateFromMemory [232]	Unpacks a NTemplate from the specified memory buffer.
	NTemplateGetFingersEx [233]	Retrieves the fingers template of the NTemplate.
	NTemplateGetFaces [234]	Retrieves the faces template of the NTemplate.
	NTemplateGetFingers [234]	Retrieves the fingers template of the NTemplate.
	NTemplateGetIrisesEx [235]	Retrieves the irises template of the NTemplate.
	NTemplateGetIrises [235]	Retrieves the irises template of the NTemplate.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NTemplateGetPalms [236]	Retrieves the palms template of the NTemplate.
	NTemplateGetPalmsEx [236]	Retrieves the palms template of the NTemplate.
	NTemplateGetSizeMem [237]	Retrieves the size of the packed NTemplate.
	NTemplateGetVoices [237]	Retrieves the voices template of the NTemplate.
	NTemplatePack [238]	Packs packed fingers, palms, irises, faces and voices templates as NTemplate into the specified memory buffer.
	NTemplateGetVoicesEx [239]	Retrieves the voices template of the NTemplate.
	NTemplateRemoveFaces [240]	Removes faces template (NLTemplate [153]) from the NTemplate.
	NTemplateRemoveFingers [240]	Removes fingers template (NFTemplate) from the NTemplate.
	NTemplateRemoveIrises [241]	Removes irises template (NETemplate) from the NTemplate.
	NTemplateRemovePalms [241]	Removes palmprint template (NFTemplate) from the NTemplate.
	NTemplateRemoveVoices [242]	Removes voice template (NSTemplate) from the NTemplate.
	NTemplateSetFaces [242]	Sets the faces template of the NTemplate to the specified value.
	HNTemplate [243]	Handle to NTemplate object.
	NTemplateSetFingers [243]	Sets the fingers template of the NTemplate to the specified value.
	NTemplateSetIrises [243]	Sets the itises template of the NTemplate to the specified value.
	NTemplateSetPalms [243]	Sets the palms template of the NTemplate to the specified value.
	NTemplateSetVoices [244]	Sets the voices template of the NTemplate to the specified value.
	NTemplateUnpack [244]	Unpacks packed fingers, palms, iris, face or voice template from the packed NTemplate.

Macros

Name	Description
NTemplateAddFaces [245]	Adds an empty faces template to the NTemplate.
NTemplateAddFacesCopy [246]	Adds a copy of an existing fingers template to the NTemplate. **Parameters**
NTemplateAddFacesFromMemory [246]	Unpacks a faces template from the specified memory buffer and adds it to the NTemplate.
NTemplateAddFingers [247]	Adds an empty fingers template to the NTemplate.
NTemplateAddFingersCopy [247]	Adds a copy of an existing fingers template to the NTemplate.
NTemplateAddFingersFromMemory [248]	Unpacks a fingers template from the specified memory buffer and adds it to the NTemplate.
NTemplateAddIrises [249]	Adds an empty irises template to the NTemplate.
NTemplateAddIrisesCopy [249]	Adds a copy of an existing iris template to the NTemplate.
NTemplateAddIrisesFromMemory [249]	Unpacks a irises template from the specified memory buffer and adds it to the NTemplate.
NTemplateAddPalms [250]	Adds an empty palms template to the NTemplate.
NTemplateAddPalmsCopy [251]	Adds a copy of an existing palm template to the NTemplate.
NTemplateAddPalmsFromMemory [251]	Unpacks a palms template from the specified memory buffer and adds it to the NTemplate.
NTemplateAddVoicesCopy [252]	Adds a copy of an existing voice template to the NTemplate.
NTemplateClone [252]	Creates a copy of the specified NTemplate.
NTemplateCreate [252]	Creates an empty NTemplate.
NTemplateGetSize [253]	Calculates packed size of the NTemplate.
NTemplateSaveToMemory [253]	Packs the NTemplate into the specified memory buffer.

7.1.1.13.1 Functions

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.13.1.1 NTemplateAddFacesCopyEx Function

Adds a copy of an existing face template to the NTemplate [219].

C++

```
NResult N_API NTemplateAddFacesCopyEx(HNTemplate hTemplate, HNLTemplate hSrcFaces, NUInt
flags, HNLTemplate * phFaces);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219].
HNLTemplate hSrcFaces	[in] A handle to the NLTemplate [153] to be copied.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. This parameter is reserved, must be zero.
HNLTemplate * phFaces	[out] A pointer to a handle to the copy of NLTemplate [153].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.1.2 NTemplateAddFacesEx Function

Adds an empty faces template to the NTemplate [219].

C++

```
NResult N_API NTemplateAddFacesEx(HNTemplate hTemplate, NUInt flags, HNLTemplate * phFaces);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. This parameter is reserved, must be zero.
HNLTemplate * phFaces	[out] Pointer to a HNLTemplate [165] that receives handle to created NLTemplate [153] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHFaces* is NULL [1075].
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains faces template. See NTemplateGetFaces [234] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Module

NTemplate Unit [219]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.13.1.3 NTemplateAddFacesFromMemoryEx Function

Unpacks a faces template from the specified memory buffer and adds it to the NTemplate [219].

C++

```
NResult N_API NTemplateAddFacesFromMemoryEx(HNTemplate hTemplate, const void * pBuffer,
NSizeType bufferSize, NUInt flags, NSizeType * pSize, HNLTemplate * phFaces);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
const void * pBuffer	[in] Pointer to memory buffer that contains packed NLTemplate [153].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NLTemplate [153].
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
NSizeType * pSize	[out] Actual size of data read from pBuffer (bytes).
HNLTemplate * phFaces	[out] Pointer to HNLTemplate [165] that receives handle to created NLTemplate [153] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHFaces* or *buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NLTemplate [153] format.
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains faces template. See NTemplateGetFaces [234] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

For the list of flags that are supported see NLTemplateCreateFromMemory [158] function.

Module

NTemplate Unit [219]

7.1.1.13.1.4 NTemplateAddFingersCopyEx Function

Adds a copy of an existing fingers template to the NTemplate [219].

C++

```
NResult N_API NTemplateAddFingersCopyEx(HNTemplate hTemplate, HNFTemplate hSrcFingers,
NUInt flags, HNFTemplate * phFingers);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNFTemplate hSrcFingers	[in] A handle to the NFTemplate to be copied.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. This parameter is reserved, must be zero.
HNFTemplate * phFingers	[out] A pointer to a handle to the copy of NFTemplate.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.1.5 NTemplateAddFingersEx Function

Adds an empty fingers template to the NTemplate [219].

C++

```
NResult N_API NTemplateAddFingersEx(HNTemplate hTemplate, NUInt flags, HNFTemplate *
phFingers);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. This parameter is reserved, must be zero.
HNFTemplate * phFingers	[out] Pointer to a HNFTemplate that receives handle to created NFTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHFingers* is NULL [1075].
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains fingers template. See NTemplateGetFingers [234] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Module

NTemplate Unit [219]

7.1.1.13.1.6 NTemplateAddFingersFromMemoryEx Function

Unpacks a fingers template from the specified memory buffer and adds it to the NTemplate [219].

C++

```
NResult N_API NTemplateAddFingersFromMemoryEx(HNTemplate hTemplate, const void * pBuffer,
NSizeType bufferSize, NUInt flags, NSizeType * pSize, HNFTemplate * phFingers);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const void * pBuffer	[in] Pointer to memory buffer that contains packed NFTemplate.
NSizeType bufferSize	[in] Size of memory buffer that contains packed NFTemplate.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
NSizeType * pSize	[out] Actual size of data read from pBuffer (bytes).
HNFTemplate * phFingers	[out] Pointer to HNFTemplate that receives handle to created NFTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHFingers*, or *buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NFTemplate format.
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains fingers template. See NTemplateGetFingers [234] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

For the list of flags that are supported see NFTemplateCreateFromMemory function.

Module

NTemplate Unit [219]

7.1.1.13.1.7 NTemplateAddIrisesCopyEx Function

Adds a copy of an existing iris template to the NTemplate [219].

C++

```
NResult N_API NTemplateAddIrisesCopyEx(HNTemplate hTemplate, HNETemplate hSrcIrises, NUInt
flags, HNETemplate * phIrises);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219].
HNETemplate hSrcIrises	[in] A handle to the NETemplate to be copied.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. This parameter is reserved, must be zero.
HNETemplate * phIrises	[out] A pointer to a handle to the copy of NETemplate.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.13.1.8 NTemplateAddIrisesEx Function

Adds an empty irises template to the NTemplate [219].

C++

```
NResult N_API NTemplateAddIrisesEx(HNTemplate hTemplate, NUInt flags, HNETemplate *
phIrises);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. This parameter is reserved, must be zero.
HNETemplate * phIrises	[out] Pointer to a HNETemplate that receives handle to created NETemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHIrises* is NULL [1075].
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains irises template. See NTemplateGetIrises [235] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Module

NTemplate Unit [219]

7.1.1.13.1.9 NTemplateAddPalmsCopyEx Function

Adds a copy of an existing palm template to the NTemplate [219].

C++

```
NResult N_API NTemplateAddPalmsCopyEx(HNTemplate hTemplate, HNFTemplate hSrcPalms, NUInt
flags, HNFTemplate * phPalms);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
HNFTemplate hSrcPalms	[in] A handle to the NFTemplate to be copied.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. This parameter is reserved, must be zero.
HNFTemplate * phPalms	[out] A pointer to a handle to the copy of NFTemplate.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.13.1.10 NTemplateAddIrisesFromMemoryEx Function

Unpacks a irises template from the specified memory buffer and adds it to the NTemplate [219].

C++

```
NResult N_API NTemplateAddIrisesFromMemoryEx(HNTemplate hTemplate, const void * pBuffer,
NSizeType bufferSize, NUInt flags, NSizeType * pSize, HNETemplate * phIrises);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
const void * pBuffer	[in] Pointer to memory buffer that contains packed NETemplate.
NSizeType bufferSize	[in] Size of memory buffer that contains packed NETemplate.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
NSizeType * pSize	[out] Actual size of data read from pBuffer (bytes).
HNETemplate * phIrises	[out] Pointer to HNETemplate that receives handle to created NETemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHIrises* or *buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NETemplate format.
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains irises template. See NTemplateGetFingers [234] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

For the list of flags that are supported see NFTemplateCreateFromMemory function.

Module

NTemplate Unit [219]

7.1.1.13.1.11 NTemplateAddPalmsEx Function

Adds an empty palms template to the NTemplate [219].

C++

```
NResult N_API NTemplateAddPalmsEx(HNTemplate hTemplate, NUInt flags, HNFTemplate * phPalms);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. This parameter is reserved, must be zero.
HNFTemplate * phPalms	[out] Pointer to a HNFTemplate that receives handle to created NFTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHPalms* is NULL [1075].
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains palms template. See NTemplateGetPalms [236] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Module

NTemplate Unit [219]

7.1.1.13.1.12 NTemplateAddVoicesCopyEx Function

Adds a copy of an existing voice template to the NTemplate [219].

C++

```
NResult N_API NTemplateAddVoicesCopyEx(HNTemplate hTemplate, HNSTemplate hSrcVoices, NUInt
flags, HNSTemplate * phVoices);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
HNSTemplate hSrcVoices	[in] A handle to the NSTemplate to be copied.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. This parameter is reserved, must be zero.
HNSTemplate * phVoices	[out] A pointer to a handle to the copy of NSTemplate.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.1.13 NTemplateAddPalmsFromMemoryEx Function

Unpacks a palms template from the specified memory buffer and adds it to the NTemplate [219].

C++

```
NResult N_API NTemplateAddPalmsFromMemoryEx(HNTemplate hTemplate, const void * pBuffer,
NSizeType bufferSize, NUInt flags, NSizeType * pSize, HNFTemplate * phPalms);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const void * pBuffer	[in] Pointer to memory buffer that contains packed NFTemplate.
NSizeType bufferSize	[in] Size of memory buffer that contains packed NFTemplate.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
NSizeType * pSize	[out] Actual size of data read from pBuffer (bytes).
HNFTemplate * phPalms	[out] Pointer to HNFTemplate that receives handle to created NFTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHPalms* or *buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NFTemplate format.
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains irises template. See NTemplateGetPalms [236] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

For the list of flags that are supported see NFTemplateCreateFromMemory function.

Module

NTemplate Unit [219]

7.1.1.13.1.14 NTemplateAddVoices Function

Adds an empty voice template to the NTemplate [219].

C++

```
NResult N_API NTemplateAddVoices(HNTemplate hTemplate, NUInt flags, HNSTemplate * phVoices);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object. This object will be used for adding a new voice template.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
HNSTemplate * phVoices	[out] Pointer to a HNSTemplate that receives handle to created NSTemplate object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.13.1.15 NTemplateCheck Function

Checks if format of the packed NTemplate [219] is correct.

C++

```
NResult N_API NTemplateCheck(const void * pBuffer, NSizeType bufferSize);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NTemplate [219].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NTemplate [219].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NTemplate [219] format.

Module

NTemplate Unit [219]

7.1.1.13.1.16 NTemplateClear Function

Removes all templates from the NTemplate [219].

C++

```
NResult N_API NTemplateClear(HNTemplate hTemplate);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* is NULL [1075].

Module

NTemplate Unit [219]

7.1.1.13.1.17 NTemplateAddVoicesFromMemory Function

Unpacks a voice template from the specified memory buffer and adds it to the NTemplate [219].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NTemplateAddVoicesFromMemory(HNTemplate hTemplate, const void * pBuffer,
NSizeType bufferSize, NUInt flags, NSizeType * pSize, HNSTemplate * phVoices);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
const void * pBuffer	[in] Pointer to memory buffer that contains packed NSTemplate.
NSizeType bufferSize	[in] Size of memory buffer that contains packed NSTemplate.
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
NSizeType * pSize	[out] Actual size of data read from pBuffer (bytes).
HNSTemplate * phVoices	[out] Pointer to HNSTemplate that receives handle to created NSTemplate object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.1.18 NTemplateCalculateSize Function

Calculates the size of a packed NTemplate [219] containing fingers, palms, irises or faces templates of the specified size.

C++

```
NResult N_API NTemplateCalculateSize(NSizeType fingersTemplateSize, NSizeType
facesTemplateSize, NSizeType irisesTemplateSize, NSizeType palmsTemplateSize, NSizeType
voicesTemplateSize, NSizeType * pSize);
```

Parameters

Parameters	Description
NSizeType fingersTemplateSize	[in] The size of packed fingers template.
NSizeType facesTemplateSize	[in] The size of packed faces template.
NSizeType irisesTemplateSize	[in] The size of packed irises template.
NSizeType palmsTemplateSize	[in] The size of packed palms template.
NSizeType voicesTemplateSize	[in] The size of packed voices template.
NSizeType * pSize	[out] Pointer to NSizeType [1008] that receives calculated size of packed NTemplate [219].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT [427]	*fingersTemplateSize* is not zero and is less than minimal size of packed fingers template. - or - *facesTemplateSize* is not zero and is less than minimal size of packed fingers template. - or - *irisesTemplateSize* is not zero and is less than minimal size of packed irises template. - or - *palmsTemplateSize* is not zero and is less than minimal size of packed palms template. - or - *voicesTemplateSize* is not zero and is less than minimal size of packed palms template.
N_E_ARGUMENT_NULL [427]	*pSize* is NULL [1075].

Remarks

This is a low-level function and can be changed in future version of the library.

In order to calculate the size of the whole NTemplate [219] you should know the size of each template. It can be achieved by using functions listed below:

- For fingerprint and palmprint templates:
  - NTemplateGetFingers [234] function is used to check if the NTemplate [219] contains NFTemplate with fingerprint records. If the return value is not NULL [1075] the NTemplate [219] contains NFTemplate with fingerprint records. The size of the NFTemplate can be calculated by using NFTemplateCalculateSize function.
  - NTemplateGetPalms [236] function is used to check if the NTemplate [219] contains NFTemplate with palmprint records. If the return value is not NULL [1075] the NTemplate [219] contains NFTemplate with palmprint records. The size of the NFTemplate can be calculated by using NFTemplateCalculateSize function.
- For face templates:
  - NTemplateGetFaces [234] function is used to check if the NTemplate [219] contains NLTemplate [153] with face records. If the return value is not NULL [1075] the NTemplate [219] contains NLTemplate [153] with face records. The size of the NLTemplate [153] can be calculated by using NLTemplateCalculateSize [156] function.
- For iris templates:
  - NTemplateGetIrises [235] function is used to check if the NTemplate [219] contains NFTemplate with fingerprint records. If the return value is not NULL [1075] the NTemplate [219] contains NETemplate with iris records. The size of the NETemplate can be calculated by using NETemplateCalculateSize function.
- For voice templates:
  - NTemplateGetVoices [237] function is used to check if the NTemplate [219] contains NSTemplate with voice records. If the return value is not NULL [1075] the NTemplate [219] contains NSTemplate with voice records. The size of the NSTemplate can be calculated by using NSTemplateCalculateSize function.

Module

NTemplate Unit [219]

7.1.1.13.1.19 NTemplateGetFacesEx Function

Retrieves the faces template of the NTemplate [219].

C++

```
NResult N_API NTemplateGetFacesEx(HNTemplate hTemplate, HNLTemplate * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
HNLTemplate * phValue	[out] Points to HNLTemplate [165] that receives handle to NLTemplate [153] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *phValue* is NULL [1075].

Remarks

If the NTemplate [219] does not contain palms template (NLTemplate [153]), the returned handle is NULL [1075].

Module

NTemplate Unit [219]

7.1.1.13.1.20 NTemplateCreateEx Function

Creates an empty NTemplate [219] with specified flags.

C++

```
NResult N_API NTemplateCreateEx(NUInt flags, HNTemplate * phTemplate);
```

Parameters

Parameters	Description
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function. This parameter is reserved, must be zero.
HNTemplate * phTemplate	[out] Pointer to HNTemplate [243] that receives handle to created NTemplate [219] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*phTemplate* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

Created object must be deleted using NObjectFree [542] function.

Module

NTemplate Unit [219]

7.1.1.13.1.21 NTemplateCreateFromMemory Function

Unpacks a NTemplate [219] from the specified memory buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NTemplateCreateFromMemory(const void * pBuffer, NSizeType bufferSize, NUInt flags, NSizeType * pSize, HNTemplate * phTemplate);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NTemplate [219].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NTemplate [219].
NUInt flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
NSizeType * pSize	[out] Actual size of data read from pBuffer (bytes).
HNTemplate * phTemplate	[out] Pointer to HNTemplate [243] that receives handle to newly created NTemplate [219] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*phTemplate* or *buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer buffer points to is inconsistent with NFTemplate format.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

When created object is not necessary anymore it must be deleted using NObjectFree [542] function.

Module

NTemplate Unit [219]

7.1.1.13.1.22 NTemplateGetFingersEx Function

Retrieves the fingers template of the NTemplate [219].

C++

```
NResult N_API NTemplateGetFingersEx(HNTemplate hTemplate, HNFTemplate * phValue);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
HNFTemplate * phValue	[out] Points to HNFTemplate that receives handle to NFTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *phValue* is NULL [1075].

Remarks

If the NTemplate [219] does not contain fingers template (NFTemplate), the returned handle is NULL [1075].

Module

NTemplate Unit [219]

7.1.1.13.1.23 NTemplateGetFaces Function

Retrieves the faces template of the NTemplate [219].

C++

```
NResult N_API NTemplateGetFaces(HNTemplate hTemplate, HNLTemplate * phValue);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
HNLTemplate * phValue	[out] Points to HNLTemplate [165] that receives handle to NLTemplate [153] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *phValue* is NULL [1075].

Remarks

If the NTemplate [219] does not contain face template, the returned handle is NULL [1075].

Module

NTemplate Unit [219]

7.1.1.13.1.24 NTemplateGetFingers Function

Retrieves the fingers template of the NTemplate [219].

C++

```
NResult N_API NTemplateGetFingers(HNTemplate hTemplate, HNFTemplate * phValue);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
HNFTemplate * phValue	[out] Points to HNFTemplate that receives handle to NFTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *phValue* is NULL [1075].

Remarks

If the NTemplate [219] does not contain fingers template, the returned handle is NULL [1075].

Module

NTemplate Unit [219]

7.1.1.13.1.25 NTemplateGetIrisesEx Function

Retrieves the irises template of the NTemplate [219].

C++

```
NResult N_API NTemplateGetIrisesEx(HNTemplate hTemplate, HNETemplate * phValue);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
HNETemplate * phValue	[out] Points to HNETemplate that receives handle to NETemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *phValue* is NULL [1075].

Remarks

If the NTemplate [219] does not contain irises template (NETemplate), the returned handle is NULL [1075].

Module

NTemplate Unit [219]

7.1.1.13.1.26 NTemplateGetIrises Function

Retrieves the irises template of the NTemplate [219].

C++

```
NResult N_API NTemplateGetIrises(HNTemplate hTemplate, HNETemplate * phValue);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
HNETemplate * phValue	[out] Points to HNETemplate that receives handle to NETemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *phValue* is NULL [1075].

Remarks

If the NTemplate [219] does not contain irises template (NETemplate), the returned handle is NULL [1075].

Module

NTemplate Unit [219]

7.1.1.13.1.27 NTemplateGetPalms Function

Retrieves the palms template of the NTemplate [219].

C++

```
NResult N_API NTemplateGetPalms(HNTemplate hTemplate, HNFTemplate * phValue);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
HNFTemplate * phValue	[out] Points to HNFTemplate that receives handle to NFTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *phValue* is NULL [1075].

Remarks

If the NTemplate [219] does not contain palms template (NFTemplate), the returned handle is NULL [1075].

Module

NTemplate Unit [219]

7.1.1.13.1.28 NTemplateGetPalmsEx Function

Retrieves the palms template of the NTemplate [219].

C++

```
NResult N_API NTemplateGetPalmsEx(HNTemplate hTemplate, HNFTemplate * phValue);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
HNFTemplate * phValue	[out] Points to HNFTemplate that receives handle to NFTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *phValue* is NULL [1075].

Remarks

If the NTemplate [219] does not contain palms template (NFTemplate), the returned handle is NULL [1075].

Module

NTemplate Unit [219]

7.1.1.13.1.29 NTemplateGetSizeMem Function

Retrieves the size of the packed NTemplate [219].

C++

```
NResult N_API NTemplateGetSizeMem(const void * pBuffer, NSizeType bufferSize, NSizeType *
pValue);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NTemplate [219].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NTemplate [219].
NSizeType * pValue	[out] Pointer to NSizeType [1008] that receives size of sound template.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.1.30 NTemplateGetVoices Function

Retrieves the voices template of the NTemplate [219].

C++

```
NResult N_API NTemplateGetVoices(HNTemplate hTemplate, HNSTemplate * phValue);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
HNSTemplate * phValue	[out] Points to HNSTemplate that receives handle to NSTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *phValue* is NULL [1075].

Remarks

If the NTemplate [219] does not contain voices template (NSTemplate), the returned handle is NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTemplate Unit [219]

7.1.1.13.1.31 NTemplatePack Function

Packs packed fingers, palms, irises, faces and voices templates as NTemplate [219] into the specified memory buffer.

C++

```
NResult N_API NTemplatePack(const void * pFingersTemplate, NSizeType fingersTemplateSize,
const void * pFacesTemplate, NSizeType facesTemplateSize, const void * pIrisesTemplate,
NSizeType irisesTemplateSize, const void * pPalmsTemplate, NSizeType palmsTemplateSize,
const void * pVoicesTemplate, NSizeType voicesTemplateSize, void * pBuffer, NSizeType
bufferSize, NSizeType * pSize);
```

Parameters

Parameters	Description
const void * pFingersTemplate	[in] Pointer to memory buffer that contains packed fingers template.
NSizeType fingersTemplateSize	[in] Size of memory buffer that contains packed fingers template.
const void * pFacesTemplate	[in] Pointer to memory buffer that contains packed faces template.
NSizeType facesTemplateSize	[in] Size of memory buffer that contains packed faces template.
const void * pIrisesTemplate	[in] Pointer to memory buffer that contains packed irises template.
NSizeType irisesTemplateSize	[in] Size of memory buffer that contains packed irises template.
const void * pPalmsTemplate	[in] Pointer to memory buffer that contains packed palms template.
NSizeType palmsTemplateSize	[in] Size of memory buffer that contains packed palms template.
const void * pVoicesTemplate	[in] Pointer to memory buffer that contains packed voices template.
NSizeType voicesTemplateSize	[in] Size of memory buffer that contains packed voices template.
void * pBuffer	[out] Pointer to memory buffer to store packed NTemplate [219]. If NULL [1075] is passed as a parameter an error occurs. To get buffer size use NTemplateCalculateSize [230] or NTemplateGetSize [253] functions.
NSizeType bufferSize	[in] Size of memory buffer to store packed NTemplate [219].
NSizeType * pSize	[out] Pointer to NSizeType [1008] that receives size of packed NTemplate [219].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT [427]	*bufferSize* is less than size required to store packed NTemplate [219]. - or - *fingersTemplateSize* is less than minimal fingers template size. - or - *fingersTemplateSize* is not equal to size stored in memory buffer *pFingersTemplate* points to. - or - *facesTemplateSize* is less than minimal faces template size. - or - *facesTemplateSize* is not equal to size stored in memory buffer *pFacesTemplate* points to. - or - *irisesTemplateSize* is less than minimal irises template size. - or - *irisesTemplateSize* is not equal to size stored in memory buffer *pIrisesTemplate* points to. - or - *palmsTemplateSize* is less than minimal palms template size. - or - *palmsTemplateSize* is not equal to size stored in memory buffer *pPalmsTemplate* points to. - or - *voicesTemplateSize* is less than minimal palms template size. - or - *voicesTemplateSize* is not equal to size stored in memory buffer *pVoicesTemplate* points to.
N_E_ARGUMENT_NULL [427]	*pFingersTemplate* is NULL [1075] and *fingersTemplateSize* is not equal to zero. - or - *pFacesTemplate* is NULL [1075] and *facesTemplateSize* is not equal to zero. - or - *pIrisesTemplate* is NULL [1075] and *irisesTemplateSize* is not equal to zero. - or - *pPalmsTemplate* is NULL [1075] and *palmsTemplateSize* is not equal to zero. - or - *pVoicesTemplate* is NULL [1075] and voices*TemplateSize* is not equal to zero. - or - *buffer* or *pSize* is NULL [1075].
N_E_FORMAT [429]	Data in memory buffer *pFingersTemplate* points to is inconsistent with NFTemplate format.

Remarks

bufferSize must not be less than value calculated with NTemplateCalculateSize [230] function with the same parameter values.

Module

NTemplate Unit [219]

7.1.1.13.1.32 NTemplateGetVoicesEx Function

Retrieves the voices template of the NTemplate [219].

C++

```
NResult N_API NTemplateGetVoicesEx(HNTemplate hTemplate, HNSTemplate * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.
HNSTemplate * phValue	[out] Points to HNSTemplate that receives handle to NSTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *phValue* is NULL [1075].

Remarks

If the NTemplate [219] does not contain voices template (NSTemplate), the returned handle is NULL [1075].

Module

NTemplate Unit [219]

7.1.1.13.1.33 NTemplateRemoveFaces Function

Removes faces template (NLTemplate [153]) from the NTemplate [219].

C++

```
NResult N_API NTemplateRemoveFaces(HNTemplate hTemplate);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* is NULL [1075].

Remarks

If the NTemplate [219] does not contain faces template, does nothing.

Module

NTemplate Unit [219]

7.1.1.13.1.34 NTemplateRemoveFingers Function

Removes fingers template (NFTemplate) from the NTemplate [219].

C++

```
NResult N_API NTemplateRemoveFingers(HNTemplate hTemplate);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* is NULL [1075].

Remarks

If the NTemplate [219] does not contain fingers template, does nothing.

Module

NTemplate Unit [219]

7.1.1.13.1.35 NTemplateRemoveIrises Function

Removes irises template (NETemplate) from the NTemplate [219].

C++

```
NResult N_API NTemplateRemoveIrises(HNTemplate hTemplate);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* is NULL [1075].

Remarks

If the NTemplate [219] does not contain irises template, does nothing.

Module

NTemplate Unit [219]

7.1.1.13.1.36 NTemplateRemovePalms Function

Removes palmprint template (NFTemplate) from the NTemplate [219].

C++

```
NResult N_API NTemplateRemovePalms(HNTemplate hTemplate);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* is NULL [1075].

Remarks

If the NTemplate [219] does not contain palmprint template, does nothing.

Module

NTemplate Unit [219]

7.1.1.13.1.37 NTemplateRemoveVoices Function

Removes voice template (NSTemplate) from the NTemplate [219].

C++

```
NResult N_API NTemplateRemoveVoices(HNTemplate hTemplate);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* is NULL [1075].

Remarks

If the NTemplate [219] does not contain voice template, does nothing.

Module

NTemplate Unit [219]

7.1.1.13.1.38 NTemplateSetFaces Function

Sets the faces template of the NTemplate [219] to the specified value.

C++

```
NResult N_API NTemplateSetFaces(HNTemplate hTemplate, HNLTemplate hValue);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219].
HNLTemplate hValue	[in] Handle to the new NLTemplate [153] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTemplate Unit [219]

7.1.1.13.1.39 HNTemplate

Handle to NTemplate [219] object.

Module

NTemplate Unit [219]

7.1.1.13.1.40 NTemplateSetFingers Function

Sets the fingers template of the NTemplate [219] to the specified value.

C++

```
NResult N_API NTemplateSetFingers(HNTemplate hTemplate, HNFTemplate hValue);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219].
HNFTemplate hValue	[in] Handle to the new NFTemplate value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.1.41 NTemplateSetIrises Function

Sets the itises template of the NTemplate [219] to the specified value.

C++

```
NResult N_API NTemplateSetIrises(HNTemplate hTemplate, HNETemplate hValue);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219].
HNETemplate hValue	[in] Handle to the new NETemplate value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.1.42 NTemplateSetPalms Function

Sets the palms template of the NTemplate [219] to the specified value.

C++

```
NResult N_API NTemplateSetPalms(HNTemplate hTemplate, HNFTemplate hValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219].
HNFTemplate hValue	[in] Handle to the new NFTemplate value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.1.43 NTemplateSetVoices Function

Sets the voices template of the NTemplate [219] to the specified value.

C++

```
NResult N_API NTemplateSetVoices(HNTemplate hTemplate, HNSTemplate hValue);
```

Parameters

Parameters	Description
HNTemplate hTemplate	[in] Handle to the NTemplate [219].
HNSTemplate hValue	[in] Handle to the new NSTemplate value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.1.44 NTemplateUnpack Function

Unpacks packed fingers, palms, iris, face or voice template from the packed NTemplate [219].

C++

```
NResult N_API NTemplateUnpack(const void * pBuffer, NSizeType bufferSize, NVersion_ *
pVersion, NUInt * pSize, NByte * pHeaderSize, const void * * ppFingersTemplate, NSizeType *
pFingersTemplateSize, const void * * ppFacesTemplate, NSizeType * pFacesTemplateSize, const
void * * ppIrisesTemplate, NSizeType * pIrisesTemplateSize, const void * * ppPalmsTemplate,
NSizeType * pPalmsTemplateSize, const void * * ppVoicesTemplate, NSizeType *
pVoicesTemplateSize);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer that contains packed NTemplate [219].
NSizeType bufferSize	[in] Size of memory buffer that contains packed NTemplate [219].
NVersion_ * pVersion	[out] Pointer to NByte [1005] that receives version of the packed NTemplate [219]. Can be NULL [1075].
NUInt * pSize	[out] Pointer to NUInt [1009] that receives size of the packed NTemplate [219]. Can be NULL [1075].
NByte * pHeaderSize	[out] Pointer to NByte [1005] that receives header size of the packed NTemplate [219]. Can be NULL [1075].
const void * * ppFingersTemplate	[in] Pointer to void * that receives pointer to packed fingers template contained in the packed NTemplate [219]. Can be NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSizeType * pFingersTemplateSize	[out] Pointer to NSizeType [1008] that receives size of packed fingers template contained in the packed NTemplate [219]. Can be NULL [1075].
const void * * ppFacesTemplate	[in] Pointer to void * that receives pointer to packed faces template contained in the packed NTemplate [219]. Can be NULL [1075].
NSizeType * pFacesTemplateSize	[out] Pointer to NSizeType [1008] that receives size of packed faces template contained in the packed NTemplate [219]. Can be NULL [1075].
const void * * ppIrisesTemplate	[out] Pointer to void * that receives pointer to packed irises template contained in the packed NTemplate [219]. Can be NULL [1075].
NSizeType * pIrisesTemplateSize	[in] Pointer to NSizeType [1008] that receives size of packed irises template contained in the packed NTemplate [219]. Can be NULL [1075].
const void * * ppPalmsTemplate	[in] Pointer to void * that receives pointer to packed palms template contained in the packed NTemplate [219]. Can be NULL [1075].
NSizeType * pPalmsTemplateSize	[out] Pointer to NSizeType [1008] that receives size of packed palms template contained in the packed NTemplate [219]. Can be NULL [1075].
const void * * ppVoicesTemplate	[in] Pointer to void * that receives pointer to packed voices template contained in the packed NTemplate [219]. Can be NULL [1075].
NSizeType * pVoicesTemplateSize	[out] Pointer to NSizeType [1008] that receives size of packed voices template contained in the packed NTemplate [219]. Can be NULL [1075].
pMinorVersion	[out] Pointer to NByte [1005] that receives minor version of the packed NTemplate [219]. Can be NULL [1075].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NTemplate [219] format.

Module

NTemplate Unit [219]

7.1.1.13.2 Macros

7.1.1.13.2.1 NTemplateAddFaces Macro

Adds an empty faces template to the NTemplate [219].

C++

```
#define NTemplateAddFaces(hTemplate, phFaces) NTemplateAddFacesEx(hTemplate, 0, phFaces)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219] object.
phFaces	[out] Pointer to a HNLTemplate [165] that receives handle to created NLTemplate [153] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHFaces* is NULL [1075].
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains faces template. See NTemplateGetFaces [234] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Module

NTemplate Unit [219]

7.1.1.13.2.2 NTemplateAddFacesCopy Macro

Adds a copy of an existing fingers template to the NTemplate [219].

Parameters

C++

```
#define NTemplateAddFacesCopy(hTemplate, hSrcFaces, phFaces)
NTemplateAddFacesCopyEx(hTemplate, hSrcFaces, 0, phFaces)
```

Module

NTemplate Unit [219]

7.1.1.13.2.3 NTemplateAddFacesFromMemory Macro

Unpacks a faces template from the specified memory buffer and adds it to the NTemplate [219].

C++

```
#define NTemplateAddFacesFromMemory(hTemplate, pBuffer, bufferSize, flags, phFaces)
NTemplateAddFacesFromMemoryEx(hTemplate, pBuffer, bufferSize, flags, NULL, phFaces)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219] object.
pBuffer	[in] Pointer to memory buffer that contains packed NLTemplate [153].
bufferSize	[in] Size of memory buffer that contains packed NLTemplate [153].
flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
phFaces	[out] Pointer to HNLTemplate [165] that receives handle to created NLTemplate [153] object.

Returns

If the function succeeds, the return value is N_OK [433].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHFaces* or *buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NLTemplate [153] format.
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains faces template. See NTemplateGetFaces [234] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

For the list of flags that are supported see NLTemplateCreateFromMemory [158] function.

Module

NTemplate Unit [219]

7.1.1.13.2.4 NTemplateAddFingers Macro

Adds an empty fingers template to the NTemplate [219].

C++

```
#define NTemplateAddFingers(hTemplate, phFingers) NTemplateAddFingersEx(hTemplate, 0,
phFingers)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219] object.
phFingers	[out] Pointer to a HNFTemplate that receives handle to created NFTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHFingers* is NULL [1075].
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains fingers template. See NTemplateGetFingers [234] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Module

NTemplate Unit [219]

7.1.1.13.2.5 NTemplateAddFingersCopy Macro

Adds a copy of an existing fingers template to the NTemplate [219].

C++

```
#define NTemplateAddFingersCopy(hTemplate, hSrcFingers, phFingers)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
NTemplateAddFingersCopyEx(hTemplate, hSrcFingers, 0, phFingers)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219].
hSrcFingers	[in] A handle to the NFTemplate to be copied.
phFingers	[out] A pointer to a handle to the copy of NFTemplate.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.2.6 NTemplateAddFingersFromMemory Macro

Unpacks a fingers template from the specified memory buffer and adds it to the NTemplate [219].

C++

```
#define NTemplateAddFingersFromMemory(hTemplate, pBuffer, bufferSize, flags, phFingers)
NTemplateAddFingersFromMemoryEx(hTemplate, pBuffer, bufferSize, flags, NULL, phFingers)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219] object.
pBuffer	[in] Pointer to memory buffer that contains packed NFTemplate.
bufferSize	[in] Size of memory buffer that contains packed NFTemplate.
flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
phFingers	[out] Pointer to HNFTemplate that receives handle to created NFTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHFingers*, or *buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NFTemplate format.
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains fingers template. See NTemplateGetFingers [234] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

For the list of flags that are supported see NFTemplateCreateFromMemory function.

Module

NTemplate Unit [219]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.1.13.2.7 NTemplateAddIrises Macro

Adds an empty irises template to the NTemplate [219].

C++

```
#define NTemplateAddIrises(hTemplate, phIrises) NTemplateAddIrisesEx(hTemplate, 0, phIrises)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219] object.
phIrises	[out] Pointer to a HNETemplate that receives handle to created NETemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHIrises* is NULL [1075].
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains irises template. See NTemplateGetIrises [235] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Module

NTemplate Unit [219]

7.1.1.13.2.8 NTemplateAddIrisesCopy Macro

Adds a copy of an existing iris template to the NTemplate [219].

C++

```
#define NTemplateAddIrisesCopy(hTemplate, hSrcIrises, phIrises)
NTemplateAddIrisesCopyEx(hTemplate, hSrcIrises, 0, phIrises)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219].
hSrcIrises	[in] A handle to the NETemplate to be copied.
phIrises	[out] A pointer to a handle to the copy of NETemplate.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.2.9 NTemplateAddIrisesFromMemory Macro

Unpacks a irises template from the specified memory buffer and adds it to the NTemplate [219].

C++

```
#define NTemplateAddIrisesFromMemory(hTemplate, pBuffer, bufferSize, flags, phIrises)
NTemplateAddIrisesFromMemoryEx(hTemplate, pBuffer, bufferSize, flags, NULL, phIrises)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219] object.
pBuffer	[in] Pointer to memory buffer that contains packed NETemplate.
bufferSize	[in] Size of memory buffer that contains packed NETemplate.
flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
phIrises	[out] Pointer to HNETemplate that receives handle to created NETemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHIrises* or *buffer* is NULL [1075].
N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NETemplate format.
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains irises template. See NTemplateGetFingers [234] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

For the list of flags that are supported see NETemplateCreateFromMemory function.

Module

NTemplate Unit [219]

7.1.1.13.2.10 NTemplateAddPalms Macro

Adds an empty palms template to the NTemplate [219].

C++

```
#define NTemplateAddPalms(hTemplate, phPalms) NTemplateAddPalmsEx(hTemplate, 0, phPalms)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219] object.
phPalms	[out] Pointer to a HNFTemplate that receives handle to created NFTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHPalms* is NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_INVALID_OPERATION [430]	NTemplate [219] already contains palms template. See NTemplateGetPalms [236] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Module

NTemplate Unit [219]

7.1.1.13.2.11 NTemplateAddPalmsCopy Macro

Adds a copy of an existing palm template to the NTemplate [219].

C++

```
#define NTemplateAddPalmsCopy(hTemplate, hSrcPalms, phPalms)
NTemplateAddPalmsCopyEx(hTemplate, hSrcPalms, 0, phPalms)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219] object.
hSrcPalms	[in] A handle to the NFTemplate to be copied.
phPalms	[out] A pointer to a handle to the copy of NFTemplate.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.2.12 NTemplateAddPalmsFromMemory Macro

Unpacks a palms template from the specified memory buffer and adds it to the NTemplate [219].

C++

```
#define NTemplateAddPalmsFromMemory(hTemplate, pBuffer, bufferSize, flags, phPalms)
NTemplateAddPalmsFromMemoryEx(hTemplate, pBuffer, bufferSize, flags, NULL, phPalms)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219] object.
pBuffer	[in] Pointer to memory buffer that contains packed NFTemplate.
bufferSize	[in] Size of memory buffer that contains packed NFTemplate.
flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
phPalms	[out] Pointer to HNFTemplate that receives handle to created NFTemplate object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pHPalms* or *buffer* is NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_END_OF_STREAM [429]	*bufferSize* is less than expected.
N_E_FORMAT [429]	Data in memory buffer *buffer* points to is inconsistent with NFTemplate format.
N_E_INVALID_OPERATION [430]	NTemplate [219] already contains irises template. See NTemplateGetPalms [236] function.
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

For the list of flags that are supported see NFTemplateCreateFromMemory function.

Module

NTemplate Unit [219]

7.1.1.13.2.13 NTemplateAddVoicesCopy Macro

Adds a copy of an existing voice template to the NTemplate [219].

C++

```
#define NTemplateAddVoicesCopy(hTemplate, hSrcVoices, phVoices)
NTemplateAddVoicesCopyEx(hTemplate, hSrcVoices, 0, phVoices)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219] object.
hSrcVoices	[in] A handle to the NSTemplate to be copied.
phVoices	[out] A pointer to a handle to the copy of NSTemplate.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTemplate Unit [219]

7.1.1.13.2.14 NTemplateClone Macro

Creates a copy of the specified NTemplate [219].

C++

```
#define NTemplateClone(hTemplate, phClonedTemplate) NObjectClone(hTemplate, (HNObject
*)phClonedTemplate)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the original NTemplate [219].
phClonedTemplate	[out] Handle to the copied NTemplate [219].

Module

NTemplate Unit [219]

7.1.1.13.2.15 NTemplateCreate Macro

Creates an empty NTemplate [219].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NTemplateCreate(phTemplate) NTemplateCreateEx(0, phTemplate)
```

Parameters

Parameters	Description
phTemplate	[out] Pointer to HNTemplate [243] that receives handle to created NTemplate [219] object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*phTemplate* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	There was not enough memory.

Remarks

Created object must be deleted using NObjectFree [542] function.

Module

NTemplate Unit [219]

7.1.1.13.2.16 NTemplateGetSize Macro

Calculates packed size of the NTemplate [219].

C++

```
#define NTemplateGetSize(hTemplate, flags, pValue) NObjectGetSize(hTemplate, flags, pValue)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219] object.
flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
pSize	[out] Pointer to NSizeType [1008] that receives calculated size of packed NTemplate [219].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pSize* is NULL [1075].

Remarks

For the list of flags that are supported see NTemplateSaveToMemory [253] function.

Module

NTemplate Unit [219]

7.1.1.13.2.17 NTemplateSaveToMemory Macro

Packs the NTemplate [219] into the specified memory buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NTemplateSaveToMemory(hTemplate, pBuffer, bufferSize, flags, pSize)
NObjectSaveToMemoryDst(hTemplate, pBuffer, bufferSize, flags, pSize)
```

Parameters

Parameters	Description
hTemplate	[in] Handle to the NTemplate [219] object.
pBuffer	[out] Pointer to memory buffer to store packed NTemplate [219]. If NULL [1075] is passed as a parameter an error occurs. To get buffer size use NTemplateCalculateSize [230] or NTemplateGetSize [253] functions.
bufferSize	[in] Size of memory buffer to store packed NTemplate [219].
flags	[in] Bitwise combination of zero or more flags that controls behavior of the function.
pSize	[out] Pointer to NSizeType [1008] that receives size of packed NTemplate [219].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT [427]	*buffer* is not NULL [1075] and *bufferSize* is less than size required to store packed NTemplate [219].
N_E_ARGUMENT_NULL [427]	*hTemplate* or *pSize* is NULL [1075]. - or - *buffer* is NULL [1075] and *bufferSize* is not zero.

Remarks

If *buffer* is NULL [1075] and *bufferSize* is zero the function only calculates the size of the buffer needed and has the same effect as NTemplateGetSize [253] function.

If *buffer* is not NULL [1075], *bufferSize* must not be less than value calculated with NTemplateGetSize [253] function.

Module

NTemplate Unit [219]

7.1.1.13.3 Files

7.1.1.13.3.1 NTemplate.h

Header file for the NTemplate [219]. Provides functionality for packing, unpacking and editing Neurotechnology Templates (NTemplates).

Functions

	Name	Description
	NTemplateAddFacesCopyEx [221]	Adds a copy of an existing face template to the NTemplate [219].
	NTemplateAddFacesEx [221]	Adds an empty faces template to the NTemplate [219].
	NTemplateAddFacesFromMemoryEx [222]	Unpacks a faces template from the specified memory buffer and adds it to the NTemplate [219].
	NTemplateAddFingersCopyEx [222]	Adds a copy of an existing fingers template to the NTemplate [219].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NTemplateAddFingersEx [223]	Adds an empty fingers template to the NTemplate [219].
	NTemplateAddFingersFromMemoryEx [223]	Unpacks a fingers template from the specified memory buffer and adds it to the NTemplate [219].
	NTemplateAddIrisesCopyEx [224]	Adds a copy of an existing iris template to the NTemplate [219].
	NTemplateAddIrisesEx [225]	Adds an empty irises template to the NTemplate [219].
	NTemplateAddIrisesFromMemoryEx [226]	Unpacks a irises template from the specified memory buffer and adds it to the NTemplate [219].
	NTemplateAddPalmsCopyEx [225]	Adds a copy of an existing palm template to the NTemplate [219].
	NTemplateAddPalmsEx [226]	Adds an empty palms template to the NTemplate [219].
	NTemplateAddPalmsFromMemoryEx [227]	Unpacks a palms template from the specified memory buffer and adds it to the NTemplate [219].
	NTemplateAddVoices [228]	Adds an empty voice template to the NTemplate [219].
	NTemplateAddVoicesCopyEx [227]	Adds a copy of an existing voice template to the NTemplate [219].
	NTemplateAddVoicesFromMemory [229]	Unpacks a voice template from the specified memory buffer and adds it to the NTemplate [219].
	NTemplateCalculateSize [230]	Calculates the size of a packed NTemplate [219] containing fingers, palms, irises or faces templates of the specified size.
	NTemplateCheck [229]	Checks if format of the packed NTemplate [219] is correct.
	NTemplateClear [229]	Removes all templates from the NTemplate [219].
	NTemplateCreateEx [232]	Creates an empty NTemplate [219] with specified flags.
	NTemplateCreateFromMemory [232]	Unpacks a NTemplate [219] from the specified memory buffer.
	NTemplateGetFaces [234]	Retrieves the faces template of the NTemplate [219].
	NTemplateGetFacesEx [231]	Retrieves the faces template of the NTemplate [219].
	NTemplateGetFingers [234]	Retrieves the fingers template of the NTemplate [219].
	NTemplateGetFingersEx [233]	Retrieves the fingers template of the NTemplate [219].
	NTemplateGetIrises [235]	Retrieves the irises template of the NTemplate [219].
	NTemplateGetIrisesEx [235]	Retrieves the irises template of the NTemplate [219].
	NTemplateGetPalms [236]	Retrieves the palms template of the NTemplate [219].
	NTemplateGetPalmsEx [236]	Retrieves the palms template of the NTemplate [219].
	NTemplateGetSizeMem [237]	Retrieves the size of the packed NTemplate [219].
	NTemplateGetVoices [237]	Retrieves the voices template of the NTemplate [219].
	NTemplateGetVoicesEx [239]	Retrieves the voices template of the NTemplate [219].
	NTemplatePack [238]	Packs packed fingers, palms, irises, faces and voices templates as NTemplate [219] into the specified memory buffer.
	NTemplateRemoveFaces [240]	Removes faces template (NLTemplate [153]) from the NTemplate [219].
	NTemplateRemoveFingers [240]	Removes fingers template (NFTemplate) from the NTemplate [219].
	NTemplateRemoveIrises [241]	Removes irises template (NETemplate) from the NTemplate [219].
	NTemplateRemovePalms [241]	Removes palmprint template (NFTemplate) from the NTemplate [219].
	NTemplateRemoveVoices [242]	Removes voice template (NSTemplate) from the NTemplate [219].
	NTemplateSetFaces [242]	Sets the faces template of the NTemplate [219] to the specified value.
	NTemplateSetFingers [243]	Sets the fingers template of the NTemplate [219] to the specified value.
	NTemplateSetIrises [243]	Sets the itises template of the NTemplate [219] to the specified value.
	NTemplateSetPalms [243]	Sets the palms template of the NTemplate [219] to the specified value.
	NTemplateSetVoices [244]	Sets the voices template of the NTemplate [219] to the specified value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NTemplateUnpack [244]	Unpacks packed fingers, palms, iris, face or voice template from the packed NTemplate [219].

Macros

Name	Description
NTemplateAddFaces [245]	Adds an empty faces template to the NTemplate [219].
NTemplateAddFacesCopy [246]	Adds a copy of an existing fingers template to the NTemplate [219]. **Parameters**
NTemplateAddFacesFromMemory [246]	Unpacks a faces template from the specified memory buffer and adds it to the NTemplate [219].
NTemplateAddFingers [247]	Adds an empty fingers template to the NTemplate [219].
NTemplateAddFingersCopy [247]	Adds a copy of an existing fingers template to the NTemplate [219].
NTemplateAddFingersFromMemory [248]	Unpacks a fingers template from the specified memory buffer and adds it to the NTemplate [219].
NTemplateAddIrises [249]	Adds an empty irises template to the NTemplate [219].
NTemplateAddIrisesCopy [249]	Adds a copy of an existing iris template to the NTemplate [219].
NTemplateAddIrisesFromMemory [249]	Unpacks a irises template from the specified memory buffer and adds it to the NTemplate [219].
NTemplateAddPalms [250]	Adds an empty palms template to the NTemplate [219].
NTemplateAddPalmsCopy [251]	Adds a copy of an existing palm template to the NTemplate [219].
NTemplateAddPalmsFromMemory [251]	Unpacks a palms template from the specified memory buffer and adds it to the NTemplate [219].
NTemplateAddVoicesCopy [252]	Adds a copy of an existing voice template to the NTemplate [219].
NTemplateClone [252]	Creates a copy of the specified NTemplate [219].
NTemplateCreate [252]	Creates an empty NTemplate [219].
NTemplateGetSize [253]	Calculates packed size of the NTemplate [219].
NTemplateSaveToMemory [253]	Packs the NTemplate [219] into the specified memory buffer.

Module

NTemplate Unit [219]

7.1.1.14 NTemplates Unit

Provides functionality for retrieving information about NTemplates library.

Files

Name	Description
NTemplates.h [256]	Header file for the NTemplates library.

7.1.1.14.1 Files

7.1.1.14.1.1 NTemplates.h

Header file for the NTemplates library.

Module

NTemplates Unit [256]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2 NCore Library

Provides infrastructure for Neurotechnology components.

Remarks

Requirements (Windows)

- **Import Library**: NCore.dll.lib.
- **DLL**: NCore.dll.

Requirements (Linux)

- **Shared object**: libNCore.so.

Enumerations

Name	Description
NEncoding [389]	Represents a character encoding.

Modules

Name	Description
NArray Unit [259]	Provides functionality for working with arrays.
NBinaryReader Unit [291]	Provides functions to read primitive data types in specific encoding.
NBinaryWriter Unit [303]	Provides functions to write primitive types in binary to a stream.
NBuffer Unit [313]	Provides access to unmanaged blocks of memory from managed code.
NBufferedStream Unit [320]	Provides functions for adding a buffering layer to read and write operations on another stream.
NCallback Unit [321]	Provides function callback functionality.
NConsole Unit [333]	Represents the standard input, output, and error streams for console applications.
NCore Unit [352]	Provides infrastructure/basic functionality for Neurotechnology components.
NCustomParameterDescriptor Unit [357]	Provides custom parameter descriptor management functionality.
NCustomPropertyDescriptor Unit [358]	Provides custom property descriptor management functionality.
NCustomStream Unit [359]	
NDefs Unit [378]	
NDescriptor Unit [384]	Provides descriptor management functionality.
NDirectory Unit [385]	
NEnum Unit [386]	Provides base functionality for enumerations.
NError Unit [398]	Defines error codes used in Neurotechnology components.
NEvent Unit [448]	
NExpandableObject Unit [457]	Provides functionality for retrieving information about specified object.
NFile Unit [458]	Provides static methods for the creation, copying, deletion, moving, and opening of files, and aids in the creation of FileStream objects.
NFileStream Unit [483]	Provides functions supporting both synchronous and asynchronous file read and write operations.
NInterlocked Unit [488]	Provides atomic operations for variables that are shared by multiple threads.
NIOTypes Unit [493]	Defines input/output types used in Neurotechnology components.
NLibraryInfo Unit [498]	Provides definitions of library info structure type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMemberDescriptor Unit [501]	Provides member descriptor management functionality.
NMemory Unit [502]	Provides memory management for Neurotechnology components.
NMemoryStream Unit [510]	Provides functions for working with memory streams.
NModule Unit [516]	Each separate piece of code/functionality is a module. The functions in this unit allow to get the information about the module it's name, version, etc..
NModuleInternal Unit [529]	Each separate piece of code/functionality is a module. The functions in this unit allow to set the information about the module it's name, version, etc..
NMonitor Unit [532]	Lightweight, cross platform primitives for critical section.
NMutex Unit [534]	Cross platform implementation of critical sections.
NNoDeprecate Unit [536]	
NObject Unit [536]	Provides functionality for retrieving information about specified object.
NParameterBag Unit [569]	A helper class that aids in implementation of PropertyBag for ParameterDescriptor class.
NParameters Unit [575]	Provides functionality for working with parameters for Neurotechnology components.
NParameterDescriptor Unit [579]	Provides functions for getting information about parameter.
NPath Unit [583]	Performs operations on String instances that contain file or directory path information. These operations are performed in a cross-platform manner.
NPlugin Unit [594]	Plug-in management framework for use in other Neurotechnology components.
NPluginInternal Unit [605]	
NPluginManager Unit [607]	Provides functions for managing plugins.
NPluginModule Unit [629]	Provides functions to retrieve information about plugins.
NPluginModuleInternal Unit [635]	Provides functions to modify information about plugins.
NProcessorInfo Unit [637]	Provides functionality for getting processor information.
NReDeprecate Unit [651]	
NRWLock Unit [651]	
NSemaphore Unit [654]	Cross platform semaphore (synchronization primitive) implementation.
NStopwatch Unit [656]	Provides a set of methods and properties that you can use to accurately measure elapsed time.
NStream Unit [660]	Supports internal Neurotechnology libraries infrastructure and should not be used directly in your code.
NStreamReader Unit [668]	Implements a TextReader that reads characters from a byte stream in a particular encoding.
NStreamWriter Unit [673]	Implements a TextWriter for writing characters to a stream in a particular encoding.
NString Unit [678]	Provides functionality for working with strings.
NStringBuilder Unit [769]	Represents a mutable string of characters.
NStringReader Unit [822]	Implements a TextReader that reads from a string.
NStringWriter Unit [824]	Implements a TextWriter for writing information to a string. The information is stored in an underlying StringBuilder.
NSyncEvent Unit [827]	Cross platform synchronization primitive. It allows for threads to wait for event without consuming CPU. The other thread(s) would notify waiting thread(s) of the event by setting it to signalled state. See the NSyncEventCreate [827] function for more details.
NTextReader Unit [829]	Represents a reader that can read a sequential series of characters.
NTextWriter Unit [833]	
NThread Unit [851]	Cross platform threading.
NTls Unit [868]	Represents Thread Local Storage.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NType Unit [871]	Manages object types.
NTypeDescriptor Unit [903]	Provides information about the characteristics for a component, such as its attributes, properties, and events.
NTypes Unit [905]	Defines types and macros used in Neurotechnology components.
NValue Unit [1094]	Provides functionality for management and conversion of values of different types.
NWaitObject Unit [1123]	Base unit for common functionality of some cross platform synchronization primitives (supporting timed wait). See derived units for complete description: • NMutex [534] • NSemaphore [654] • NSyncEvent [827]
NWindows Unit [1124]	

Types

Name	Description
NDateTime [361]	Represents an instant in time, typically expressed as a date and time of day.
NTimeSpan [855]	Represents a time interval.

7.1.2.1 NArray Unit

Provides functionality for working with arrays.

Files

Name	Description
NArray.h [289]	Header file for the NArray unit.

Functions

	Name	Description
	NArrayCopyTo [261]	Copies the specified array into another array.
	NArrayCopyToBooleanArray [262]	Copies the specified Boolean array into another array.
	NArrayCopyToByteArray [262]	Copies the specified byte array into another array.
	NArrayCopyToCharArray [262]	Copies the specified char array into another array.
	NArrayCopyToDoubleArray [263]	Copies the specified double value array into another array.
	NArrayCopyToInt16Array [263]	Copies the specified short integer array into another array.
	NArrayCopyToInt32Array [264]	Copies the specified integer array into another array.
	NArrayCopyToInt64Array [264]	Copies the specified long integer array into another array.
	NArrayCopyToObjectArray [264]	Copies the specified object array into another array.
	NArrayCopyToObjectArrayP [265]	Copies the specified object array into another array.
	NArrayCopyToP [265]	Copies the specified array into another array.
	NArrayCopyToPointerArray [266]	Copies the specified pointer array into another array.
	NArrayCopyToResultArray [266]	Copies the specified NResult [1007] value array into another array.
	NArrayCopyToSByteArray [267]	Copies the specified signed byte array into another array.
	NArrayCopyToSingleArray [267]	Copies the specified floating point value array into another array.
	NArrayCopyToSizeTypeArray [267]	Copies the specified NSizeType [1008] value array into another array.
	NArrayCopyToSSizeTypeArray [268]	Copies the specified NSSizeType [1009] value array into another array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NArrayCopyToStringArray [268]	Copies the specified string array into another array.
	NArrayCopyToUInt16Array [269]	Copies the specified unsigned short integer array into another array.
	NArrayCopyToUInt32Array [269]	Copies the specified unsigned integer array into another array.
	NArrayCopyToUInt64Array [269]	Copies the specified unsigned long integer array into another array.
	NArrayCreate [270]	Creates an array of specified type populated by specified values.
	NArrayCreateCustom [270]	Creates an array of specified type with additional options populated by specified values.
	NArrayCreateFromBooleanArray [271]	Creates an array from an array of Boolean values.
	NArrayCreateFromByteArray [271]	Creates an array from an array of bytes.
	NArrayCreateFromCharArray [272]	Creates an array from an array of chars.
	NArrayCreateFromDoubleArray [272]	Creates an array from an array of double values.
	NArrayCreateFromInt16Array [273]	Creates an array from an array of short integer values.
	NArrayCreateFromInt32Array [273]	Creates an array from an array of integer values.
	NArrayCreateFromInt64Array [273]	Creates an array from an array of long integer values.
	NArrayCreateFromObjectArray [274]	Creates an array from an array of object values.
	NArrayCreateFromObjectArrayP [274]	Creates an array from an array of object values.
	NArrayCreateFromPointerArray [275]	Creates an array from an array of pointer values.
	NArrayCreateFromResultArray [275]	Creates an array from an array of NResult [1007] values.
	NArrayCreateFromSByteArray [275]	Creates an array from an array of signed byte values.
	NArrayCreateFromSingleArray [276]	Creates an array from an array of floating point values.
	NArrayCreateFromSizeTypeArray [276]	Creates an array from an array of NSizeType [1008] values.
	NArrayCreateFromSSizeTypeArray [277]	Creates an array from an array of NSSizeType [1009] values.
	NArrayCreateFromStringArray [277]	Creates an array from an array of string values.
	NArrayCreateFromUInt16Array [277]	Creates an array from an array of unsinged short integer values.
	NArrayCreateFromUInt32Array [278]	Creates an array from an array of unsinged integer values.
	NArrayCreateFromUInt64Array [278]	Creates an array from an array of unsinged long integer values.
	NArrayCreateP [279]	Creates an array of specified type populated by specified values.
	NArrayGetLength [279]	Retrieves the length of the array.
	NArrayGetValue [279]	Retrieves the specified value in the array.
	NArrayGetValueAs [280]	Retrieves the of the specified value of the specified type in the array.
	NArrayGetValueAsBoolean [280]	Retrieves the of the specified Boolean value in the array.
	NArrayGetValueAsByte [281]	Retrieves the of the specified byte value in the array.
	NArrayGetValueAsChar [281]	Retrieves the of the specified char value in the array.
	NArrayGetValueAsDouble [281]	Retrieves the of the specified double value in the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NArrayGetValueAsInt16 [282]	Retrieves the of the specified short integer value in the array.
	NArrayGetValueAsInt32 [282]	Retrieves the of the specified integer value in the array.
	NArrayGetValueAsInt64 [283]	Retrieves the of the specified long integer value in the array.
	NArrayGetValueAsObject [283]	Retrieves the of the specified object value in the array.
	NArrayGetValueAsObjectP [283]	Retrieves the of the specified object value in the array.
	NArrayGetValueAsP [284]	Retrieves the of the specified value of the specified type in the array.
	NArrayGetValueAsPointer [284]	Retrieves the of the specified pointer value in the array.
	NArrayGetValueAsResult [285]	Retrieves the of the specified NResult [1007] value in the array.
	NArrayGetValueAsSByte [285]	Retrieves the of the specified signed byte value in the array.
	NArrayGetValueAsSingle [285]	Retrieves the of the specified floating point integer value in the array.
	NArrayGetValueAsSizeType [286]	Retrieves the of the specified NSizeType [1008] value in the array.
	NArrayGetValueAsSSizeType [286]	Retrieves the of the specified NSSizeType [1009] value in the array.
	NArrayGetValueAsString [287]	Retrieves the of the specified string value in the array.
	NArrayGetValueAsUInt16 [287]	Retrieves the of the specified unsigned short integer value in the array.
	NArrayGetValueAsUInt32 [287]	Retrieves the of the specified unsigned integer value in the array.
	NArrayGetValueAsUInt64 [288]	Retrieves the of the specified unsigned long integer value in the array.

Macros

Name	Description
NArrayCopyToCharArray [288]	Picks a Unicode (W) or ANSCII (A) version of the NArrayCopyToCharArray function.
NArrayCreateFromCharArray [289]	Picks a Unicode (W) or ANSCII (A) version of the NArrayCreateFromCharArray function.
NArrayCreateFromStringArray [289]	Picks a Unicode (W) or ANSCII (A) version of the NArrayCreateFromStringArray function.
NArrayGetValueAsChar [289]	Picks a Unicode (W) or ANSCII (A) version of the NArrayGetValueAsChar function.

7.1.2.1.1 Functions

7.1.2.1.1.1 NArrayCopyTo Function

Copies the specified array into another array.

C++

```
NResult N_API NArrayCopyTo(HNArray hArray, HNType hType, NAttributes attributes, HNString
hFormat, void * arValues, NSizeType valuesSize, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
HNType hType	[in] Handle to the type of values in the array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
void * arValues	[out] Pointer to a the destination array.
NSizeType valuesSize	[in] Size of the values in the array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NArray Unit [259]

7.1.2.1.1.2 NArrayCopyToBooleanArray Function

Copies the specified Boolean array into another array.

C++

```
NResult N_API NArrayCopyToBooleanArray(HNArray hArray, NAttributes attributes, HNString
hFormat, NBool * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NBool * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.3 NArrayCopyToByteArray Function

Copies the specified byte array into another array.

C++

```
NResult N_API NArrayCopyToByteArray(HNArray hArray, NAttributes attributes, HNString
hFormat, NByte * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NByte * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.4 NArrayCopyToCharArray Function

Copies the specified char array into another array.

C++

```
NResult N_API NArrayCopyToCharArray(HNArray hArray, NAttributes attributes, HNString
hFormat, NChar * arValues, NInt valuesLength);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NChar * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.5 NArrayCopyToDoubleArray Function

Copies the specified double value array into another array.

C++

```
NResult N_API NArrayCopyToDoubleArray(HNArray hArray, NAttributes attributes, HNString
hFormat, NDouble * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NDouble * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.6 NArrayCopyToInt16Array Function

Copies the specified short integer array into another array.

C++

```
NResult N_API NArrayCopyToInt16Array(HNArray hArray, NAttributes attributes, HNString
hFormat, NShort * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NShort * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.7 NArrayCopyToInt32Array Function

Copies the specified integer array into another array.

C++

```
NResult N_API NArrayCopyToInt32Array(HNArray hArray, NAttributes attributes, HNString
hFormat, NInt * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NInt * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.8 NArrayCopyToInt64Array Function

Copies the specified long integer array into another array.

C++

```
NResult N_API NArrayCopyToInt64Array(HNArray hArray, NAttributes attributes, HNString
hFormat, NLong * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NLong * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.9 NArrayCopyToObjectArray Function

Copies the specified object array into another array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NArrayCopyToObjectArray(HNArray hArray, HNType hType, NAttributes attributes,
HNString hFormat, HNObject * arhValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
HNType hType	[in] Handle to the type of objects in the array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
HNObject * arhValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.10 NArrayCopyToObjectArrayP Function

Copies the specified object array into another array.

C++

```
NResult N_API NArrayCopyToObjectArrayP(HNArray hArray, NTypeOfProc pTypeOf, NAttributes
attributes, HNString hFormat, HNObject * arhValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NTypeOfProc pTypeOf	[in] Pointer to the function that retrieves the type of the objects in the array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
HNObject * arhValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.11 NArrayCopyToP Function

Copies the specified array into another array.

C++

```
NResult N_API NArrayCopyToP(HNArray hArray, NTypeOfProc pTypeOf, NAttributes attributes,
HNString hFormat, void * arValues, NSizeType valuesSize, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTypeOfProc pTypeOf	[in] Pointer to the function that retrieves the type of the values in the array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
void * arValues	[out] Pointer to the destination array.
NSizeType valuesSize	[in] Size of the values in the array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.12 NArrayCopyToPointerArray Function

Copies the specified pointer array into another array.

C++

```
NResult N_API NArrayCopyToPointerArray(HNArray hArray, NAttributes attributes, HNString
hFormat, void * * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
void * * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.13 NArrayCopyToResultArray Function

Copies the specified NResult [1007] value array into another array.

C++

```
NResult N_API NArrayCopyToResultArray(HNArray hArray, NAttributes attributes, HNString
hFormat, NResult * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NResult * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NArray Unit [259]

7.1.2.1.1.14 NArrayCopyToSByteArray Function

Copies the specified signed byte array into another array.

C++

```
NResult N_API NArrayCopyToSByteArray(HNArray hArray, NAttributes attributes, HNString
hFormat, NSByte * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NSByte * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.15 NArrayCopyToSingleArray Function

Copies the specified floating point value array into another array.

C++

```
NResult N_API NArrayCopyToSingleArray(HNArray hArray, NAttributes attributes, HNString
hFormat, NFloat * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NFloat * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.16 NArrayCopyToSizeTypeArray Function

Copies the specified NSizeType [1008] value array into another array.

C++

```
NResult N_API NArrayCopyToSizeTypeArray(HNArray hArray, NAttributes attributes, HNString
hFormat, NSizeType * arValues, NInt valuesLength);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NSizeType * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.17 NArrayCopyToSSizeTypeArray Function

Copies the specified NSSizeType [1009] value array into another array.

C++

```
NResult N_API NArrayCopyToSSizeTypeArray(HNArray hArray, NAttributes attributes, HNString
hFormat, NSSizeType * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NSSizeType * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.18 NArrayCopyToStringArray Function

Copies the specified string array into another array.

C++

```
NResult N_API NArrayCopyToStringArray(HNArray hArray, NAttributes attributes, HNString
hFormat, HNString * arhValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
HNString * arhValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.19 NArrayCopyToUInt16Array Function

Copies the specified unsigned short integer array into another array.

C++

```
NResult N_API NArrayCopyToUInt16Array(HNArray hArray, NAttributes attributes, HNString
hFormat, NUShort * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NUShort * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.20 NArrayCopyToUInt32Array Function

Copies the specified unsigned integer array into another array.

C++

```
NResult N_API NArrayCopyToUInt32Array(HNArray hArray, NAttributes attributes, HNString
hFormat, NUInt * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NUInt * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.21 NArrayCopyToUInt64Array Function

Copies the specified unsigned long integer array into another array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NArrayCopyToUInt64Array(HNArray hArray, NAttributes attributes, HNString
hFormat, NULong * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the source array.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NULong * arValues	[out] Pointer to the destination array.
NInt valuesLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.22 NArrayCreate Function

Creates an array of specified type populated by specified values.

C++

```
NResult N_API NArrayCreate(HNType hType, const void * arValues, NSizeType valuesSize, NInt
valuesLength, NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type of values that the array will hold.
const void * arValues	[in] An array of pointers to the values that the new array will hold.
NSizeType valuesSize	[in] Size of the values that the array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.23 NArrayCreateCustom Function

Creates an array of specified type with additional options populated by specified values.

C++

```
NResult N_API NArrayCreateCustom(const void * const * arValues, NInt valuesLength,
NPointerFreeProc pFree, void * pFreeParam, NPointerGetHashCodeProc pGetHashCode, void *
pGetHashCodeParam, NPointerEqualsProc pEquals, void * pEqualsParam, NAttributes attributes,
HNArray * phValue);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const void * const * arValues	[in] An array of pointers to the values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NPointerFreeProc pFree	[in] Pointer to a function that frees an array element.
void * pFreeParam	[out] Parameters of a function that frees an array element.
NPointerGetHashCodeProc pGetHashCode	[in] Pointer to a function that retrieves the hash code of an array element
void * pGetHashCodeParam	[out] Parameters of a function that retrieves the hash code of an array element.
NPointerEqualsProc pEquals	[in] Pointer to a function that compares two array elements.
void * pEqualsParam	[out] Parameters of a function that compares two array elements.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.24 NArrayCreateFromBooleanArray Function

Creates an array from an array of Boolean values.

C++

```
NResult N_API NArrayCreateFromBooleanArray(const NBool * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NBool * arValues	[in] Pointer to the Boolean values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.25 NArrayCreateFromByteArray Function

Creates an array from an array of bytes.

C++

```
NResult N_API NArrayCreateFromByteArray(const NByte * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const NByte * arValues	[in] Pointer to the byte values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.26 NArrayCreateFromCharArray Function

Creates an array from an array of chars.

C++

```
NResult N_API NArrayCreateFromCharArray(const NChar * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NChar * arValues	[in] Pointer to the char values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.27 NArrayCreateFromDoubleArray Function

Creates an array from an array of double values.

C++

```
NResult N_API NArrayCreateFromDoubleArray(const NDouble * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NDouble * arValues	[in] Pointer to the double values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.1.1.28 NArrayCreateFromInt16Array Function

Creates an array from an array of short integer values.

C++

```
NResult N_API NArrayCreateFromInt16Array(const NShort * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NShort * arValues	[in] Pointer to the short integer values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.29 NArrayCreateFromInt32Array Function

Creates an array from an array of integer values.

C++

```
NResult N_API NArrayCreateFromInt32Array(const NInt * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NInt * arValues	[in] Pointer to the integer values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.30 NArrayCreateFromInt64Array Function

Creates an array from an array of long integer values.

C++

```
NResult N_API NArrayCreateFromInt64Array(const NLong * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NLong * arValues	[in] Pointer to the long integer values that the new array will hold.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.31 NArrayCreateFromObjectArray Function

Creates an array from an array of object values.

C++

```
NResult N_API NArrayCreateFromObjectArray(HNType hType, const HNObject * arhValues, NInt
valuesLength, NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type of objects that the array will hold.
const HNObject * arhValues	[in] Pointer to the objects that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.32 NArrayCreateFromObjectArrayP Function

Creates an array from an array of object values.

C++

```
NResult N_API NArrayCreateFromObjectArrayP(NTypeOfProc pTypeOf, const HNObject * arhValues,
NInt valuesLength, NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
NTypeOfProc pTypeOf	[in] Pointer to the function that retrieves the type of the objects that the array will hold.
const HNObject * arhValues	[in] Pointer to the objects that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.1.1.33 NArrayCreateFromPointerArray Function

Creates an array from an array of pointer values.

C++

```
NResult N_API NArrayCreateFromPointerArray(const void * const * arValues, NInt
valuesLength, NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const void * const * arValues	[in] Pointer to the an array of pointers that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.34 NArrayCreateFromResultArray Function

Creates an array from an array of NResult [1007] values.

C++

```
NResult N_API NArrayCreateFromResultArray(const NResult * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NResult * arValues	[in] Pointer to the NResult [1007] values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.35 NArrayCreateFromSByteArray Function

Creates an array from an array of signed byte values.

C++

```
NResult N_API NArrayCreateFromSByteArray(const NSByte * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NSByte * arValues	[in] Pointer to the signed byte values that the new array will hold.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.36 NArrayCreateFromSingleArray Function

Creates an array from an array of floating point values.

C++

```
NResult N_API NArrayCreateFromSingleArray(const NFloat * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NFloat * arValues	[in] Pointer to the floating point values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.37 NArrayCreateFromSizeTypeArray Function

Creates an array from an array of NSizeType [1008] values.

C++

```
NResult N_API NArrayCreateFromSizeTypeArray(const NSizeType * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NSizeType * arValues	[in] Pointer to the NSizeType [1008] values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.1.1.38 NArrayCreateFromSSizeTypeArray Function

Creates an array from an array of NSSizeType [1009] values.

C++

```
NResult N_API NArrayCreateFromSSizeTypeArray(const NSSizeType * arValues, NInt
valuesLength, NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NSSizeType * arValues	[in] Pointer to the NSSizeType [1009] values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.39 NArrayCreateFromStringArray Function

Creates an array from an array of string values.

C++

```
NResult N_API NArrayCreateFromStringArray(const NChar * const * arszValues, NInt
valuesLength, NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NChar * const * arszValues	[in] Pointer to the string values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.40 NArrayCreateFromUInt16Array Function

Creates an array from an array of unsinged short integer values.

C++

```
NResult N_API NArrayCreateFromUInt16Array(const NUShort * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NUShort * arValues	[in] Pointer to the unsinged short integer values that the new array will hold.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.41 NArrayCreateFromUInt32Array Function

Creates an array from an array of unsinged integer values.

C++

```
NResult N_API NArrayCreateFromUInt32Array(const NUInt * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NUInt * arValues	[in] Pointer to the unsinged integer values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.42 NArrayCreateFromUInt64Array Function

Creates an array from an array of unsinged long integer values.

C++

```
NResult N_API NArrayCreateFromUInt64Array(const NULong * arValues, NInt valuesLength,
NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
const NULong * arValues	[in] Pointer to the unsinged long integer values that the new array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.1.1.43 NArrayCreateP Function

Creates an array of specified type populated by specified values.

C++

```
NResult N_API NArrayCreateP(NTypeOfProc pTypeOf, const void * arValues, NSizeType valuesSize, NInt valuesLength, NAttributes attributes, HNArray * phValue);
```

Parameters

Parameters	Description
NTypeOfProc pTypeOf	[in] Pointer to the function that retrieves the type of the values that the array will hold.
const void * arValues	[in] An array of pointers to the values that the new array will hold.
NSizeType valuesSize	[in] Size of the values that the array will hold.
NInt valuesLength	[in] Length of the array.
NAttributes attributes	[in] Array attributes.
HNArray * phValue	[out] Pointer to a handle to the new array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.44 NArrayGetLength Function

Retrieves the length of the array.

C++

```
NResult N_API NArrayGetLength(HNArray hArray, NInt * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt * pValue	[out] Pointer to the length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.45 NArrayGetValue Function

Retrieves the specified value in the array.

C++

```
NResult N_API NArrayGetValue(HNArray hArray, NInt index, HNValue * phValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
HNValue * phValue	[out] Pointer to a handle to the value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.46 NArrayGetValueAs Function

Retrieves the of the specified value of the specified type in the array.

C++

```
NResult N_API NArrayGetValueAs(HNArray hArray, NInt index, HNType hType, NAttributes
attributes, HNString hFormat, void * pValue, NSizeType elementSize);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
HNType hType	[in] Handle to the type of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
void * pValue	[out] Pointer to the value.
NSizeType elementSize	[in] Size of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.47 NArrayGetValueAsBoolean Function

Retrieves the of the specified Boolean value in the array.

C++

```
NResult N_API NArrayGetValueAsBoolean(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NBoolean * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NBoolean * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.1.1.48 NArrayGetValueAsByte Function

Retrieves the of the specified byte value in the array.

C++

```
NResult N_API NArrayGetValueAsByte(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NByte * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NByte * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.49 NArrayGetValueAsChar Function

Retrieves the of the specified char value in the array.

C++

```
NResult N_API NArrayGetValueAsChar(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NChar * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NChar * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.50 NArrayGetValueAsDouble Function

Retrieves the of the specified double value in the array.

C++

```
NResult N_API NArrayGetValueAsDouble(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NDouble * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NDouble * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.51 NArrayGetValueAsInt16 Function

Retrieves the of the specified short integer value in the array.

C++

```
NResult N_API NArrayGetValueAsInt16(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NShort * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NShort * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.52 NArrayGetValueAsInt32 Function

Retrieves the of the specified integer value in the array.

C++

```
NResult N_API NArrayGetValueAsInt32(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NInt * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NInt * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.1.1.53 NArrayGetValueAsInt64 Function

Retrieves the of the specified long integer value in the array.

C++

```
NResult N_API NArrayGetValueAsInt64(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NLong * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NLong * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.54 NArrayGetValueAsObject Function

Retrieves the of the specified object value in the array.

C++

```
NResult N_API NArrayGetValueAsObject(HNArray hArray, NInt index, HNType hType, NAttributes
attributes, HNString hFormat, HNObject * phValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
HNType hType	[in] Handle to the type of object.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
HNObject * phValue	[out] Pointer to a handle to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.55 NArrayGetValueAsObjectP Function

Retrieves the of the specified object value in the array.

C++

```
NResult N_API NArrayGetValueAsObjectP(HNArray hArray, NInt index, NTypeOfProc pTypeOf,
NAttributes attributes, HNString hFormat, HNObject * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NTypeOfProc pTypeOf	[in] Pointer to the function that retrieves the type of the object.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
HNObject * phValue	[out] Pointer to a handle to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.56 NArrayGetValueAsP Function

Retrieves the of the specified value of the specified type in the array.

C++

```
NResult N_API NArrayGetValueAsP(HNArray hArray, NInt index, NTypeOfProc pTypeOf,
NAttributes attributes, HNString hFormat, void * pValue, NSizeType elementSize);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NTypeOfProc pTypeOf	[in] Pointer to the function that retrieves the type of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
void * pValue	[out] Pointer to the value.
NSizeType elementSize	[in] Size of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.57 NArrayGetValueAsPointer Function

Retrieves the of the specified pointer value in the array.

C++

```
NResult N_API NArrayGetValueAsPointer(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, void * * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

void * * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.58 NArrayGetValueAsResult Function

Retrieves the of the specified NResult [1007] value in the array.

C++

```
NResult N_API NArrayGetValueAsResult(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NResult * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NResult * pValue	[out] Pointer to a handle to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.59 NArrayGetValueAsSByte Function

Retrieves the of the specified signed byte value in the array.

C++

```
NResult N_API NArrayGetValueAsSByte(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NSByte * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NSByte * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.60 NArrayGetValueAsSingle Function

Retrieves the of the specified floating point integer value in the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NArrayGetValueAsSingle(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NFloat * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NFloat * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.61 NArrayGetValueAsSizeType Function

Retrieves the of the specified NSizeType [1008] value in the array.

C++

```
NResult N_API NArrayGetValueAsSizeType(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NSizeType * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NSizeType * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.62 NArrayGetValueAsSSizeType Function

Retrieves the of the specified NSSizeType [1009] value in the array.

C++

```
NResult N_API NArrayGetValueAsSSizeType(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NSSizeType * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSSizeType * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.63 NArrayGetValueAsString Function

Retrieves the of the specified string value in the array.

C++

```
NResult N_API NArrayGetValueAsString(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, HNString * phValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
HNString * phValue	[out] Pointer to a handle to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.64 NArrayGetValueAsUInt16 Function

Retrieves the of the specified unsigned short integer value in the array.

C++

```
NResult N_API NArrayGetValueAsUInt16(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NUShort * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NUShort * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.65 NArrayGetValueAsUInt32 Function

Retrieves the of the specified unsigned integer value in the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NArrayGetValueAsUInt32(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NUInt * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NUInt * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.1.66 NArrayGetValueAsUInt64 Function

Retrieves the of the specified unsigned long integer value in the array.

C++

```
NResult N_API NArrayGetValueAsUInt64(HNArray hArray, NInt index, NAttributes attributes,
HNString hFormat, NULong * pValue);
```

Parameters

Parameters	Description
HNArray hArray	[in] Handle to the array.
NInt index	[in] Index of the value.
NAttributes attributes	[in] Array attributes.
HNString hFormat	[in] Handle to the value format.
NULong * pValue	[out] Pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NArray Unit [259]

7.1.2.1.2 Macros

7.1.2.1.2.1 NArrayCopyToCharArray Macro

Picks a Unicode (W) or ANSCII (A) version of the NArrayCopyToCharArray function.

C++

```
#define NArrayCopyToCharArray N_FUNC_AW(NArrayCopyToCharArray)
```

Module

NArray Unit [259]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.1.2.2 NArrayCreateFromCharArray Macro

Picks a Unicode (W) or ANSCII (A) version of the NArrayCreateFromCharArray function.

C++

```
#define NArrayCreateFromCharArray N_FUNC_AW(NArrayCreateFromCharArray)
```

Module

NArray Unit [259]

7.1.2.1.2.3 NArrayCreateFromStringArray Macro

Picks a Unicode (W) or ANSCII (A) version of the NArrayCreateFromStringArray function.

C++

```
#define NArrayCreateFromStringArray N_FUNC_AW(NArrayCreateFromStringArray)
```

Module

NArray Unit [259]

7.1.2.1.2.4 NArrayGetValueAsChar Macro

Picks a Unicode (W) or ANSCII (A) version of the NArrayGetValueAsChar function.

C++

```
#define NArrayGetValueAsChar N_FUNC_AW(NArrayGetValueAsChar)
```

Module

NArray Unit [259]

7.1.2.1.3 Files

7.1.2.1.3.1 NArray.h

Header file for the NArray [259] unit.

Functions

	Name	Description
	NArrayCopyTo [261]	Copies the specified array into another array.
	NArrayCopyToBooleanArray [262]	Copies the specified Boolean array into another array.
	NArrayCopyToByteArray [262]	Copies the specified byte array into another array.
	NArrayCopyToCharArray [262]	Copies the specified char array into another array.
	NArrayCopyToDoubleArray [263]	Copies the specified double value array into another array.
	NArrayCopyToInt16Array [263]	Copies the specified short integer array into another array.
	NArrayCopyToInt32Array [264]	Copies the specified integer array into another array.
	NArrayCopyToInt64Array [264]	Copies the specified long integer array into another array.
	NArrayCopyToObjectArray [264]	Copies the specified object array into another array.
	NArrayCopyToObjectArrayP [265]	Copies the specified object array into another array.
	NArrayCopyToP [265]	Copies the specified array into another array.
	NArrayCopyToPointerArray [266]	Copies the specified pointer array into another array.
	NArrayCopyToResultArray [266]	Copies the specified NResult [1007] value array into another array.
	NArrayCopyToSByteArray [267]	Copies the specified signed byte array into another array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NArrayCopyToSingleArray [267]	Copies the specified floating point value array into another array.
	NArrayCopyToSizeTypeArray [267]	Copies the specified NSizeType [1008] value array into another array.
	NArrayCopyToSSizeTypeArray [268]	Copies the specified NSSizeType [1009] value array into another array.
	NArrayCopyToStringArray [268]	Copies the specified string array into another array.
	NArrayCopyToUInt16Array [269]	Copies the specified unsigned short integer array into another array.
	NArrayCopyToUInt32Array [269]	Copies the specified unsigned integer array into another array.
	NArrayCopyToUInt64Array [269]	Copies the specified unsigned long integer array into another array.
	NArrayCreate [270]	Creates an array of specified type populated by specified values.
	NArrayCreateCustom [270]	Creates an array of specified type with additional options populated by specified values.
	NArrayCreateFromBooleanArray [271]	Creates an array from an array of Boolean values.
	NArrayCreateFromByteArray [271]	Creates an array from an array of bytes.
	NArrayCreateFromCharArray [272]	Creates an array from an array of chars.
	NArrayCreateFromDoubleArray [272]	Creates an array from an array of double values.
	NArrayCreateFromInt16Array [273]	Creates an array from an array of short integer values.
	NArrayCreateFromInt32Array [273]	Creates an array from an array of integer values.
	NArrayCreateFromInt64Array [273]	Creates an array from an array of long integer values.
	NArrayCreateFromObjectArray [274]	Creates an array from an array of object values.
	NArrayCreateFromObjectArrayP [274]	Creates an array from an array of object values.
	NArrayCreateFromPointerArray [275]	Creates an array from an array of pointer values.
	NArrayCreateFromResultArray [275]	Creates an array from an array of NResult [1007] values.
	NArrayCreateFromSByteArray [275]	Creates an array from an array of signed byte values.
	NArrayCreateFromSingleArray [276]	Creates an array from an array of floating point values.
	NArrayCreateFromSizeTypeArray [276]	Creates an array from an array of NSizeType [1008] values.
	NArrayCreateFromSSizeTypeArray [277]	Creates an array from an array of NSSizeType [1009] values.
	NArrayCreateFromStringArray [277]	Creates an array from an array of string values.
	NArrayCreateFromUInt16Array [277]	Creates an array from an array of unsinged short integer values.
	NArrayCreateFromUInt32Array [278]	Creates an array from an array of unsinged integer values.
	NArrayCreateFromUInt64Array [278]	Creates an array from an array of unsinged long integer values.
	NArrayCreateP [279]	Creates an array of specified type populated by specified values.
	NArrayGetLength [279]	Retrieves the length of the array.
	NArrayGetValue [279]	Retrieves the specified value in the array.
	NArrayGetValueAs [280]	Retrieves the of the specified value of the specified type in the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NArrayGetValueAsBoolean [280]	Retrieves the of the specified Boolean value in the array.
	NArrayGetValueAsByte [281]	Retrieves the of the specified byte value in the array.
	NArrayGetValueAsChar [281]	Retrieves the of the specified char value in the array.
	NArrayGetValueAsDouble [281]	Retrieves the of the specified double value in the array.
	NArrayGetValueAsInt16 [282]	Retrieves the of the specified short integer value in the array.
	NArrayGetValueAsInt32 [282]	Retrieves the of the specified integer value in the array.
	NArrayGetValueAsInt64 [283]	Retrieves the of the specified long integer value in the array.
	NArrayGetValueAsObject [283]	Retrieves the of the specified object value in the array.
	NArrayGetValueAsObjectP [283]	Retrieves the of the specified object value in the array.
	NArrayGetValueAsP [284]	Retrieves the of the specified value of the specified type in the array.
	NArrayGetValueAsPointer [284]	Retrieves the of the specified pointer value in the array.
	NArrayGetValueAsResult [285]	Retrieves the of the specified NResult [1007] value in the array.
	NArrayGetValueAsSByte [285]	Retrieves the of the specified signed byte value in the array.
	NArrayGetValueAsSingle [285]	Retrieves the of the specified floating point integer value in the array.
	NArrayGetValueAsSizeType [286]	Retrieves the of the specified NSizeType [1008] value in the array.
	NArrayGetValueAsSSizeType [286]	Retrieves the of the specified NSSizeType [1009] value in the array.
	NArrayGetValueAsString [287]	Retrieves the of the specified string value in the array.
	NArrayGetValueAsUInt16 [287]	Retrieves the of the specified unsigned short integer value in the array.
	NArrayGetValueAsUInt32 [287]	Retrieves the of the specified unsigned integer value in the array.
	NArrayGetValueAsUInt64 [288]	Retrieves the of the specified unsigned long integer value in the array.

Macros

Name	Description
NArrayCopyToCharArray [288]	Picks a Unicode (W) or ANSCII (A) version of the NArrayCopyToCharArray function.
NArrayCreateFromCharArray [289]	Picks a Unicode (W) or ANSCII (A) version of the NArrayCreateFromCharArray function.
NArrayCreateFromStringArray [289]	Picks a Unicode (W) or ANSCII (A) version of the NArrayCreateFromStringArray function.
NArrayGetValueAsChar [289]	Picks a Unicode (W) or ANSCII (A) version of the NArrayGetValueAsChar function.

Module

NArray Unit [259]

7.1.2.2 NBinaryReader Unit

Provides functions to read primitive data types in specific encoding.

Functions

	Name	Description
	NBinaryReaderCreate [293]	Creates binary reader based on the specified stream and byte order.
	NBinaryReaderCreateWithEncoding [293]	Creates binary reader based on the specified stream, byte order and encoding.
	NBinaryReaderGetBaseStream [293]	Gets base stream (the one which was used to create NBinaryReader).
	NBinaryReaderGetByteOrder [294]	Gets byte order in specified NBinaryReader.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NBinaryReaderGetNull [294]	Returns empty (null) NBinaryReader which can not be used for stream reading.
	NBinaryReaderGetPosition [295]	Gets position in base stream.
	NBinaryReaderPeekChar [295]	Returns the next available character in the NBinaryReader.
	NBinaryReaderRead [295]	Reads characters from base stream specified by NBinaryReader
	NBinaryReaderReadBoolean [296]	Reads NBoolean [1004] value from the stream and advances the current position of the stream by one byte.
	NBinaryReaderReadByte [296]	Reads one byte from the stream starting from the current position.
	NBinaryReaderReadBytesDst [296]	Reads bytes from the stream to memory buffer.
	NBinaryReaderReadBytesDstN [297]	Reads bytes from the stream to specified NBuffer [313].
	NBinaryReaderReadBytesN [297]	Reads bytes from the stream to specified NBuffer [313].
	NBinaryReaderReadChar [298]	Reads the next character from the stream and advances the current position of the stream.
	NBinaryReaderReadChars [298]	Reads the specified number of characters from the stream, and advances the current position in accordance with specific character being read from the stream.
	NBinaryReaderReadCharsDst [298]	Reads the specified number of characters from the stream, and advances the current position in accordance with specific character being read from the stream.
	NBinaryReaderReadDouble [299]	Reads an 8 byte floating point value (NDouble [1005]) from the stream and advances the current position of the stream.
	NBinaryReaderReadInt16 [299]	Reads a 2-byte signed integer from the stream and advances the current position of the stream.
	NBinaryReaderReadInt32 [300]	Reads a 4-byte signed integer from the stream and advances the current position of the stream by four bytes.
	NBinaryReaderReadInt64 [300]	Reads an 8-byte signed integer from the stream and advances the current position of the stream by eight bytes.
	NBinaryReaderReadSByte [300]	Reads a signed byte from the stream and advances the current position of the stream by one byte.
	NBinaryReaderReadSingle [301]	Reads a 4-byte floating point value from the stream and advances the current position of the stream by four bytes.
	NBinaryReaderReadUInt16 [301]	Reads a 2-byte unsigned integer from the stream and advances the position of the stream by two bytes.
	NBinaryReaderReadUInt32 [301]	Reads a 4-byte unsigned integer from the stream and advances the position of the stream by four bytes.
	NBinaryReaderReadUInt64 [302]	Reads an 8-byte unsigned integer from the stream and advances the position of the stream by eight bytes.
	NBinaryReaderSeek [302]	Sets the position in the stream.
	NBinaryReaderSetPosition [302]	Sets position in the base stream.

Macros

Name	Description
NBinaryReaderPeekChar [303]	Picks a Unicode (W) or ASCII (A) version of the NBinaryReaderPeekChar function.
NBinaryReaderRead [303]	Picks a Unicode (W) or ASCII (A) version of the NBinaryReaderRead function.
NBinaryReaderReadChar [303]	Picks a Unicode (W) or ASCII (A) version of the NBinaryReaderReadChar function.
NBinaryReaderReadChars [303]	Picks a Unicode (W) or ASCII (A) version of the NBinaryReaderReadChars function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBinaryReaderReadCharsDst [303]	Picks a Unicode (W) or ASCII (A) version of the NBinaryReaderReadCharsDst function.

7.1.2.2.1 Functions

7.1.2.2.1.1 NBinaryReaderCreate Function

Creates binary reader based on the specified stream and byte order.

C++

```
NResult N_API NBinaryReaderCreate(HNStream hStream, NByteOrder byteOrder, HNBinaryReader *
phReader);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to input stream.
NByteOrder byteOrder	[in] Specifies byte order in the stream.
HNBinaryReader * phReader	[out] Pointer to created NBinaryReader [291].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.2 NBinaryReaderCreateWithEncoding Function

Creates binary reader based on the specified stream, byte order and encoding.

C++

```
NResult N_API NBinaryReaderCreateWithEncoding(HNStream hStream, NByteOrder byteOrder,
NEncoding encoding, HNBinaryReader * phReader);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to input stream.
NByteOrder byteOrder	[in] Specifies byte order in the stream.
NEncoding encoding	[in] The character encoding to use.
HNBinaryReader * phReader	[out] Pointer to created NBinaryReader [291].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.3 NBinaryReaderGetBaseStream Function

Gets base stream (the one which was used to create NBinaryReader [291]).

C++

```
NResult N_API NBinaryReaderGetBaseStream(HNBinaryReader hReader, HNStream * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
HNStream * phValue	[out] Base stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

Using base stream while reading or while using the NBinaryReader [291] can cause data loss and corruption. For example, the same bytes might be read more than once, bytes might be skipped, or character reading might become unpredictable.

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NBinaryReader Unit [291]

7.1.2.2.1.4 NBinaryReaderGetByteOrder Function

Gets byte order in specified NBinaryReader [291].

C++

```
NResult N_API NBinaryReaderGetByteOrder(HNBinaryReader hReader, NByteOrder * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NByteOrder * pValue	[out] Byte order.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.5 NBinaryReaderGetNull Function

Returns empty (null) NBinaryReader [291] which can not be used for stream reading.

C++

```
NResult N_API NBinaryReaderGetNull(HNBinaryReader * phValue);
```

Parameters

Parameters	Description
HNBinaryReader * phValue	[out] Null NBinaryReader [291].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NBinaryReader Unit [291]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.2.1.6 NBinaryReaderGetPosition Function

Gets position in base stream.

C++

```
NResult N_API NBinaryReaderGetPosition(HNBinaryReader hReader, NLong * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NLong * pValue	[out] The current position with base stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.7 NBinaryReaderPeekChar Function

Returns the next available character in the NBinaryReader [291].

C++

```
NResult N_API NBinaryReaderPeekChar(HNBinaryReader hReader, NInt * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NInt * pValue	[out] The next available character or -1 if NBinaryReader [291] (the stream) does not support seeking.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.8 NBinaryReaderRead Function

Reads characters from base stream specified by NBinaryReader [291]

C++

```
NResult N_API NBinaryReaderRead(HNBinaryReader hReader, NInt * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NInt * pValue	[out] The next character from the base stream or -1 if no characters are available.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.2.1.9 NBinaryReaderReadBoolean Function

Reads NBoolean [1004] value from the stream and advances the current position of the stream by one byte.

C++

```
NResult N_API NBinaryReaderReadBoolean(HNBinaryReader hReader, NBoolean * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NBoolean * pValue	[out] NTrue [1064] if the byte is nonzero; otherwise, NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.10 NBinaryReaderReadByte Function

Reads one byte from the stream starting from the current position.

C++

```
NResult N_API NBinaryReaderReadByte(HNBinaryReader hReader, NByte * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NByte * pValue	[out] Byte read from the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.11 NBinaryReaderReadBytesDst Function

Reads bytes from the stream to memory buffer.

C++

```
NResult N_API NBinaryReaderReadBytesDst(HNBinaryReader hReader, void * pBuffer, NSizeType
bufferSize, NBool readAll, NSizeType * pSizeRead);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
void * pBuffer	[out] Pointer to memory buffer to read characters into.
NSizeType bufferSize	[in] Size of memory buffer *pBuffer*.
NBool readAll	[in] If NTrue [1064] N_E_END_OF_STREAM [429] is returned when it is not possible to read specified number of bytes. If NFalse [1041] number of read bytes is returned through *pSizeRead*.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSizeType * pSizeRead	[out] Number of read bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.12 NBinaryReaderReadBytesDstN Function

Reads bytes from the stream to specified NBuffer [313].

C++

```
NResult N_API NBinaryReaderReadBytesDstN(HNBinaryReader hReader, HNBuffer hBuffer, NBool
readAll, NSizeType * pSizeRead);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
HNBuffer hBuffer	[in] NBuffer [313] to read data into.
NBool readAll	[in] If NTrue [1064] N_E_END_OF_STREAM [429] is returned when it is not possible to read specified number of bytes. If NFalse [1041] number of read bytes is returned through *pSizeRead*.
NSizeType * pSizeRead	[out] Number of read bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.13 NBinaryReaderReadBytesN Function

Reads bytes from the stream to specified NBuffer [313].

C++

```
NResult N_API NBinaryReaderReadBytesN(HNBinaryReader hReader, NSizeType size, NBool
readAll, HNBuffer * phBuffer);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NSizeType size	[in] Platform dependent size type of byte.
NBool readAll	[in] If NTrue [1064] N_E_END_OF_STREAM [429] is returned when it is not possible to read specified number of bytes. If NFalse [1041] read bytes is returned through *phBuffer*.
HNBuffer * phBuffer	[out] Pointer to NBuffer [313] containing read bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.2.1.14 NBinaryReaderReadChar Function

Reads the next character from the stream and advances the current position of the stream.

C++

```
NResult N_API NBinaryReaderReadChar(HNBinaryReader hReader, NChar * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NChar * pValue	[out] Character read from the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.15 NBinaryReaderReadChars Function

Reads the specified number of characters from the stream, and advances the current position in accordance with specific character being read from the stream.

C++

```
NResult N_API NBinaryReaderReadChars(HNBinaryReader hReader, NInt count, NBool readAll,
NInt * pCharCount, NChar * * parChars);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NInt count	[in] Number of characters to read.
NBool readAll	[in] If NTrue [1064] N_E_END_OF_STREAM [429] is returned when it is not possible to read specified number of bytes. If NFalse [1041] number of read bytes is returned through *pCharCount*.
NInt * pCharCount	[out] Actually read characters count.
NChar * * parChars	[out] Characters array containing data read from the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.16 NBinaryReaderReadCharsDst Function

Reads the specified number of characters from the stream, and advances the current position in accordance with specific character being read from the stream.

C++

```
NResult N_API NBinaryReaderReadCharsDst(HNBinaryReader hReader, NChar * arChars, NInt
count, NBool readAll, NInt * pCountRead);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NChar * arChars	[out] Characters array containing data read from the stream.
NInt count	[in] Number of characters to read.
NBool readAll	[in] If NTrue [1064] N_E_END_OF_STREAM [429] is returned when it is not possible to read specified number of bytes. If NFalse [1041] number of read bytes is returned through *pCharCount*.
NInt * pCountRead	[out] Actually read characters count.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.17 NBinaryReaderReadDouble Function

Reads an 8 byte floating point value (NDouble [1005]) from the stream and advances the current position of the stream.

C++

```
NResult N_API NBinaryReaderReadDouble(HNBinaryReader hReader, NDouble * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NDouble * pValue	[out] NDouble [1005] value read from the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.18 NBinaryReaderReadInt16 Function

Reads a 2-byte signed integer from the stream and advances the current position of the stream.

C++

```
NResult N_API NBinaryReaderReadInt16(HNBinaryReader hReader, NInt16 * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NInt16 * pValue	[out] NInt16 [1006] value read from the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.2.1.19 NBinaryReaderReadInt32 Function

Reads a 4-byte signed integer from the stream and advances the current position of the stream by four bytes.

C++

```
NResult N_API NBinaryReaderReadInt32(HNBinaryReader hReader, NInt32 * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NInt32 * pValue	[out] NInt32 [1006] value read from the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.20 NBinaryReaderReadInt64 Function

Reads an 8-byte signed integer from the stream and advances the current position of the stream by eight bytes.

C++

```
NResult N_API NBinaryReaderReadInt64(HNBinaryReader hReader, NInt64 * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NInt64 * pValue	[out] NInt64 [1006] value read from the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.21 NBinaryReaderReadSByte Function

Reads a signed byte from the stream and advances the current position of the stream by one byte.

C++

```
NResult N_API NBinaryReaderReadSByte(HNBinaryReader hReader, NSByte * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NSByte * pValue	[out] NSByte [1008] value read from the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.2.1.22 NBinaryReaderReadSingle Function

Reads a 4-byte floating point value from the stream and advances the current position of the stream by four bytes.

C++

```
NResult N_API NBinaryReaderReadSingle(HNBinaryReader hReader, NSingle * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NSingle * pValue	[out] NSingle [1008] value read from the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.23 NBinaryReaderReadUInt16 Function

Reads a 2-byte unsigned integer from the stream and advances the position of the stream by two bytes.

C++

```
NResult N_API NBinaryReaderReadUInt16(HNBinaryReader hReader, NUInt16 * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NUInt16 * pValue	[out] NUInt16 [1009] value read from the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.24 NBinaryReaderReadUInt32 Function

Reads a 4-byte unsigned integer from the stream and advances the position of the stream by four bytes.

C++

```
NResult N_API NBinaryReaderReadUInt32(HNBinaryReader hReader, NUInt32 * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NUInt32 * pValue	[out] NUInt32 [1009] value read from the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.2.1.25 NBinaryReaderReadUInt64 Function

Reads an 8-byte unsigned integer from the stream and advances the position of the stream by eight bytes.

C++

```
NResult N_API NBinaryReaderReadUInt64(HNBinaryReader hReader, NUInt64 * pValue);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NUInt64 * pValue	[out] NUInt64 [1009] value read from the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.26 NBinaryReaderSeek Function

Sets the position in the stream.

C++

```
NResult N_API NBinaryReaderSeek(HNBinaryReader hReader, NLong offset, NSeekOrigin origin);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NLong offset	[in] A byte offset.
NSeekOrigin origin	[in] NSeekOrigin [497] indicating the reference point used to obtain the new position.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

7.1.2.2.1.27 NBinaryReaderSetPosition Function

Sets position in the base stream.

C++

```
NResult N_API NBinaryReaderSetPosition(HNBinaryReader hReader, NLong value);
```

Parameters

Parameters	Description
HNBinaryReader hReader	[in] Handle to NBinaryReader [291].
NLong value	[in] Position to set.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryReader Unit [291]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.2.2 Macros

7.1.2.2.2.1 NBinaryReaderPeekChar Macro

Picks a Unicode (W) or ASCII (A) version of the NBinaryReaderPeekChar function.

C++

```
#define NBinaryReaderPeekChar N_FUNC_AW(NBinaryReaderPeekChar)
```

Module

NBinaryReader Unit [291]

7.1.2.2.2.2 NBinaryReaderRead Macro

Picks a Unicode (W) or ASCII (A) version of the NBinaryReaderRead function.

C++

```
#define NBinaryReaderRead N_FUNC_AW(NBinaryReaderRead)
```

Module

NBinaryReader Unit [291]

7.1.2.2.2.3 NBinaryReaderReadChar Macro

Picks a Unicode (W) or ASCII (A) version of the NBinaryReaderReadChar function.

C++

```
#define NBinaryReaderReadChar N_FUNC_AW(NBinaryReaderReadChar)
```

Module

NBinaryReader Unit [291]

7.1.2.2.2.4 NBinaryReaderReadChars Macro

Picks a Unicode (W) or ASCII (A) version of the NBinaryReaderReadChars function.

C++

```
#define NBinaryReaderReadChars N_FUNC_AW(NBinaryReaderReadChars)
```

Module

NBinaryReader Unit [291]

7.1.2.2.2.5 NBinaryReaderReadCharsDst Macro

Picks a Unicode (W) or ASCII (A) version of the NBinaryReaderReadCharsDst function.

C++

```
#define NBinaryReaderReadCharsDst N_FUNC_AW(NBinaryReaderReadCharsDst)
```

Module

NBinaryReader Unit [291]

7.1.2.3 NBinaryWriter Unit

Provides functions to write primitive types in binary to a stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NBinaryWriterCreate [305]	Creates binary writer based on the specified stream and byte order.
	NBinaryWriterCreateWithEncoding [305]	Creates binary writer based on the specified stream, byte order and encoding.
	NBinaryWriterFlush [305]	Clears all buffers for the current writer.
	NBinaryWriterGetBaseStream [306]	Gets the base stream of the NBinaryWriter.
	NBinaryWriterGetByteOrder [306]	Gets byte order in specified NBinaryWriter.
	NBinaryWriterGetNull [306]	Returns empty (null) NBinaryWriter which can not be used for stream reading.
	NBinaryWriterGetPosition [307]	Gets position in base stream.
	NBinaryWriterSeek [307]	Sets the position in the stream.
	NBinaryWriterSetPosition [307]	Sets position in the base stream.
	NBinaryWriterWriteBoolean [308]	Writes NBoolean [1004] value into the stream.
	NBinaryWriterWriteByte [308]	Writes one byte into the stream starting from the current position.
	NBinaryWriterWriteBytes [308]	Writes bytes into the stream from specified memory buffer.
	NBinaryWriterWriteBytesN [309]	Writes bytes into the stream from specified NBuffer [313].
	NBinaryWriterWriteChar [309]	Writes specified character into the stream and advances the current position of the stream.
	NBinaryWriterWriteChars [309]	Writes the specified number of characters into the stream, and advances the current position in accordance with specific character being written into the stream.
	NBinaryWriterWriteDouble [310]	Writes a 8-byte floating point value (NDouble [1005]) into the stream and advances the current position of the stream by eight bytes.
	NBinaryWriterWriteInt16 [310]	Writes 2-byte signed integer to the stream and advances the current position of the stream by two bytes.
	NBinaryWriterWriteInt32 [310]	Writes a 4-byte signed integer to the stream and advances the position of the stream by four bytes.
	NBinaryWriterWriteInt64 [311]	Writes a 8-byte signed integer to the stream and advances the position of the stream by eight bytes.
	NBinaryWriterWriteSByte [311]	Writes a signed byte into the stream and advances the position of the stream by one byte.
	NBinaryWriterWriteSingle [311]	Writes a 4-byte floating point value into the stream and advances the current position of the stream by four bytes.
	NBinaryWriterWriteUInt16 [312]	Writes a 2-byte unsigned integer to the stream and advances the position of the stream by two bytes.
	NBinaryWriterWriteUInt32 [312]	Writes a 4-byte unsigned integer to the stream and advances the position of the stream by four bytes.
	NBinaryWriterWriteUInt64 [312]	Writes a 8-byte unsigned integer to the stream and advances the position of the stream by eight bytes.

Macros

Name	Description
NBinaryWriterWriteChar [313]	Picks a Unicode (W) or ASCII (A) version of the NBinaryWriterWriteChar function.
NBinaryWriterWriteChars [313]	Picks a Unicode (W) or ASCII (A) version of the NBinaryWriterWriteChars function.

7.1.2.3.1 Functions

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.3.1.1 NBinaryWriterCreate Function

Creates binary writer based on the specified stream and byte order.

C++

```
NResult N_API NBinaryWriterCreate(HNStream hStream, NByteOrder byteOrder, HNBinaryWriter *
phWriter);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to input stream.
NByteOrder byteOrder	[in] Specifies byte order in the stream.
HNBinaryWriter * phWriter	[out] Pointer to created NBinaryWriter [303].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.2 NBinaryWriterCreateWithEncoding Function

Creates binary writer based on the specified stream, byte order and encoding.

C++

```
NResult N_API NBinaryWriterCreateWithEncoding(HNStream hStream, NByteOrder byteOrder,
NEncoding encoding, HNBinaryWriter * phWriter);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to input stream.
NByteOrder byteOrder	[in] Specifies byte order in the stream.
NEncoding encoding	[in] The character encoding to use.
phReader	[out] Pointer to created NBinaryWriter [303].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.3 NBinaryWriterFlush Function

Clears all buffers for the current writer.

C++

```
NResult N_API NBinaryWriterFlush(HNBinaryWriter hWriter);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	_nt_

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.4 NBinaryWriterGetBaseStream Function

Gets the base stream of the NBinaryWriter [303].

C++

```
NResult N_API NBinaryWriterGetBaseStream(HNBinaryWriter hWriter, HNStream * phValue);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryReader [291].
HNStream * phValue	[out] Base stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.5 NBinaryWriterGetByteOrder Function

Gets byte order in specified NBinaryWriter [303].

C++

```
NResult N_API NBinaryWriterGetByteOrder(HNBinaryWriter hWriter, NByteOrder * pValue);
```

Parameters

Parameters	Description
NByteOrder * pValue	[out] Byte order.
hReader	[in] Handle to NBinaryWriter [303].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.6 NBinaryWriterGetNull Function

Returns empty (null) NBinaryWriter [303] which can not be used for stream reading.

C++

```
NResult N_API NBinaryWriterGetNull(HNBinaryWriter * phValue);
```

Parameters

Parameters	Description
HNBinaryWriter * phValue	[out] Null NBinaryWriter [303].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.7 NBinaryWriterGetPosition Function

Gets position in base stream.

C++

```
NResult N_API NBinaryWriterGetPosition(HNBinaryWriter hWriter, NLong * pValue);
```

Parameters

Parameters	Description
NLong * pValue	[out] The current position with base stream.
hReader	[in] Handle to NBinaryWriter [303].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.8 NBinaryWriterSeek Function

Sets the position in the stream.

C++

```
NResult N_API NBinaryWriterSeek(HNBinaryWriter hWriter, NLong offset, NSeekOrigin origin);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NLong offset	[in] A byte offset.
NSeekOrigin origin	[in] NSeekOrigin [497] indicating the reference point used to obtain the new position.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.9 NBinaryWriterSetPosition Function

Sets position in the base stream.

C++

```
NResult N_API NBinaryWriterSetPosition(HNBinaryWriter hWriter, NLong value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NLong value	[in] Position to set.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.10 NBinaryWriterWriteBoolean Function

Writes NBoolean [1004] value into the stream.

C++

```
NResult N_API NBinaryWriterWriteBoolean(HNBinaryWriter hWriter, NBoolean value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NBoolean value	[in] NBoolean [1004] value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.11 NBinaryWriterWriteByte Function

Writes one byte into the stream starting from the current position.

C++

```
NResult N_API NBinaryWriterWriteByte(HNBinaryWriter hWriter, NByte value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NByte value	[out] Byte written into the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.12 NBinaryWriterWriteBytes Function

Writes bytes into the stream from specified memory buffer.

C++

```
NResult N_API NBinaryWriterWriteBytes(HNBinaryWriter hWriter, const void * pBuffer,
NSizeType bufferSize);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
const void * pBuffer	[out] Pointer to memory buffer to write characters from.
NSizeType bufferSize	[in] Size of memory buffer *pBuffer*.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.13 NBinaryWriterWriteBytesN Function

Writes bytes into the stream from specified NBuffer [313].

C++

```
NResult N_API NBinaryWriterWriteBytesN(HNBinaryWriter hWriter, HNBuffer hBuffer);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
HNBuffer hBuffer	[in] NBuffer [313] to write data from.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.14 NBinaryWriterWriteChar Function

Writes specified character into the stream and advances the current position of the stream.

C++

```
NResult N_API NBinaryWriterWriteChar(HNBinaryWriter hWriter, NChar value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NChar value	[in] Character to write into the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.15 NBinaryWriterWriteChars Function

Writes the specified number of characters into the stream, and advances the current position in accordance with specific character being written into the stream.

C++

```
NResult N_API NBinaryWriterWriteChars(HNBinaryWriter hWriter, const NChar * arChars, NInt
count);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
const NChar * arChars	[in] Characters array containing data to write to the stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt count	[in] Number of characters to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.16 NBinaryWriterWriteDouble Function

Writes a 8-byte floating point value (NDouble [1005]) into the stream and advances the current position of the stream by eight bytes.

C++

```
NResult N_API NBinaryWriterWriteDouble(HNBinaryWriter hWriter, NDouble value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NDouble value	[out] NDouble [1005] written into the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.17 NBinaryWriterWriteInt16 Function

Writes 2-byte signed integer to the stream and advances the current position of the stream by two bytes.

C++

```
NResult N_API NBinaryWriterWriteInt16(HNBinaryWriter hWriter, NInt16 value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NInt16 value	[in] NInt16 [1006] written into the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.18 NBinaryWriterWriteInt32 Function

Writes a 4-byte signed integer to the stream and advances the position of the stream by four bytes.

C++

```
NResult N_API NBinaryWriterWriteInt32(HNBinaryWriter hWriter, NInt32 value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt32 value	[in] NInt32 [1006] to write to the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.19 NBinaryWriterWriteInt64 Function

Writes a 8-byte signed integer to the stream and advances the position of the stream by eight bytes.

C++

```
NResult N_API NBinaryWriterWriteInt64(HNBinaryWriter hWriter, NInt64 value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NInt64 value	[in] NInt64 [1006] to write to the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.20 NBinaryWriterWriteSByte Function

Writes a signed byte into the stream and advances the position of the stream by one byte.

C++

```
NResult N_API NBinaryWriterWriteSByte(HNBinaryWriter hWriter, NSByte value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NSByte value	[in] NSByte [1008] to write to the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.21 NBinaryWriterWriteSingle Function

Writes a 4-byte floating point value into the stream and advances the current position of the stream by four bytes.

C++

```
NResult N_API NBinaryWriterWriteSingle(HNBinaryWriter hWriter, NSingle value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NSingle value	[in] NSingle [1008] to write to the stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.22 NBinaryWriterWriteUInt16 Function

Writes a 2-byte unsigned integer to the stream and advances the position of the stream by two bytes.

C++

```
NResult N_API NBinaryWriterWriteUInt16(HNBinaryWriter hWriter, NUInt16 value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NUInt16 value	[in] NUInt16 [1009] to write to the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.23 NBinaryWriterWriteUInt32 Function

Writes a 4-byte unsigned integer to the stream and advances the position of the stream by four bytes.

C++

```
NResult N_API NBinaryWriterWriteUInt32(HNBinaryWriter hWriter, NUInt32 value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NUInt32 value	[in] NUInt32 [1009] to write to the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.1.24 NBinaryWriterWriteUInt64 Function

Writes a 8-byte unsigned integer to the stream and advances the position of the stream by eight bytes.

C++

```
NResult N_API NBinaryWriterWriteUInt64(HNBinaryWriter hWriter, NUInt64 value);
```

Parameters

Parameters	Description
HNBinaryWriter hWriter	[in] Handle to NBinaryWriter [303].
NUInt64 value	[in] NUInt64 [1009] to write to the stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBinaryWriter Unit [303]

7.1.2.3.2 Macros

7.1.2.3.2.1 NBinaryWriterWriteChar Macro

Picks a Unicode (W) or ASCII (A) version of the NBinaryWriterWriteChar function.

C++

```
#define NBinaryWriterWriteChar N_FUNC_AW(NBinaryWriterWriteChar)
```

Module

NBinaryWriter Unit [303]

7.1.2.3.2.2 NBinaryWriterWriteChars Macro

Picks a Unicode (W) or ASCII (A) version of the NBinaryWriterWriteChars function.

C++

```
#define NBinaryWriterWriteChars N_FUNC_AW(NBinaryWriterWriteChars)
```

Module

NBinaryWriter Unit [303]

7.1.2.4 NBuffer Unit

Provides access to unmanaged blocks of memory from managed code.

Functions

	Name	Description
	NBufferCopy [314]	Copies data into current NBuffer from a different NBuffer with specified offsets into both buffers and specified size.
	NBufferCopyFromPtr [314]	Copies data into current NBuffer from a different NBuffer with specified offsets into current NBuffer and specified size of the source NBuffer.
	NBufferCopyTo [315]	Reads the bytes from the current NBuffer and writes them to another buffer, using a specified offset.
	NBufferCopyToPtr [315]	Reads the bytes from the current NBuffer and writes them to another buffer, indicated by a pointer, using a specified buffer size.
	NBufferCopyToPtrRange [315]	Reads the bytes from the current NBuffer and writes them to another buffer, using a specified offset and buffer size.
	NBufferCreate [316]	Creates a new NBuffer using the specified memory location and size.
	NBufferCreateFromBuffer [316]	Creates a new NBuffer based on the specified byte array, with specified offset and size.
	NBufferCreateFromPtr [317]	Creates a new NBuffer using the specified memory location and size.
	NBufferCreateFromPtrEx [317]	Creates a new NBuffer using the specified size and a specified location of specified type.
	NBufferCreateFromPtrWithFree [317]	Creates a new NBuffer using the specified memory location and size; frees the memory in the source location.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NBufferCreateFromPtrWithFreeProc [318]	Creates a new NBuffer using the specified memory location and size; frees the memory in the source location.
	NBufferGetEmpty [318]	Returns an empty buffer.
	NBufferGetPtr [319]	Reads the bytes from the NBuffer and returns them as a pointer.
	NBufferGetSize [319]	Returns the size of the NBuffer in bytes.
	NBufferToPtr [319]	Copies the memory contained in NBuffer and returns it as a pointer.

Macros

Name	Description
NBufferCreateFromConstPtr [320]	Creates a new NBuffer using the specified memory location and size.

7.1.2.4.1 Functions

7.1.2.4.1.1 NBufferCopy Function

Copies data into current NBuffer [313] from a different NBuffer [313] with specified offsets into both buffers and specified size.

C++

```
NResult N_API NBufferCopy(HNBuffer hSrcBuffer, NSizeType srcOffset, HNBuffer hDstBuffer,
NSizeType dstOffset, NSizeType size);
```

Parameters

Parameters	Description
HNBuffer hSrcBuffer	[in] A handle to the source NBuffer [313].
NSizeType srcOffset	[in] The zero-based byte offset into the source NBuffer [313].
HNBuffer hDstBuffer	[in] A handle to the destination NBuffer [313].
NSizeType dstOffset	[in] The zero-based byte offset into the destination NBuffer [313].
NSizeType size	[in] The positive NInt32 [1006] value greater than 0 indicating the source NBuffer [313] size in bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.1.2 NBufferCopyFromPtr Function

Copies data into current NBuffer [313] from a different NBuffer [313] with specified offsets into current NBuffer [313] and specified size of the source NBuffer [313].

C++

```
NResult N_API NBufferCopyFromPtr(HNBuffer hBuffer, const void * pSrcBuffer, NSizeType
srcBufferSize, NSizeType offset);
```

Parameters

Parameters	Description
HNBuffer hBuffer	[in] A handle to NBuffer [313].
const void * pSrcBuffer	[in] A pointer to the source buffer.
NSizeType srcBufferSize	[in] The size of the source buffer in bytes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSizeType offset	[in] The zero-based byte offset into current NBuffer [313].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.1.3 NBufferCopyTo Function

Reads the bytes from the current NBuffer [313] and writes them to another buffer, using a specified offset.

C++

```
NResult N_API NBufferCopyTo(HNBuffer hBuffer, HNBuffer hDstBuffer, NSizeType dstOffset);
```

Parameters

Parameters	Description
HNBuffer hBuffer	[in] A handle to NBuffer [313].
HNBuffer hDstBuffer	[in] A handle to the destination buffer.
NSizeType dstOffset	[in] The zero-based byte offset into the destination NBuffer [313].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.1.4 NBufferCopyToPtr Function

Reads the bytes from the current NBuffer [313] and writes them to another buffer, indicated by a pointer, using a specified buffer size.

C++

```
NResult N_API NBufferCopyToPtr(HNBuffer hBuffer, void * pDstBuffer, NSizeType
dstBufferSize);
```

Parameters

Parameters	Description
HNBuffer hBuffer	[in] A handle to NBuffer [313].
void * pDstBuffer	[in] A pointer to the destination buffer.
NSizeType dstBufferSize	[in] The positive value greater than 0 indicating the destination buffer size in bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.1.5 NBufferCopyToPtrRange Function

Reads the bytes from the current NBuffer [313] and writes them to another buffer, using a specified offset and buffer size.

C++

```
NResult N_API NBufferCopyToPtrRange(HNBuffer hBuffer, NSizeType offset, void * pDstBuffer,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
NSizeType dstBufferSize, NSizeType size);
```

Parameters

Parameters	Description
HNBuffer hBuffer	[in] A handle to NBuffer [313].
NSizeType offset	[in] The zero-based byte offset into the NBuffer [313].
void * pDstBuffer	[in] A pointer to the destination buffer.
NSizeType dstBufferSize	[in] The destination buffer's size in bytes.
NSizeType size	[in] NBuffer [313]'s size in bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.1.6 NBufferCreate Function

Creates a new NBuffer [313] using the specified memory location and size.

C++

```
NResult N_API NBufferCreate(NSizeType size, HNBuffer * phBuffer);
```

Parameters

Parameters	Description
NSizeType size	[in] The size of the NBuffer [313]. This value must be greater than zero. The default size is 4096.
HNBuffer * phBuffer	[out] A pointer to the created NBuffer [313].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.1.7 NBufferCreateFromBuffer Function

Creates a new NBuffer [313] based on the specified byte array, with specified offset and size.

C++

```
NResult N_API NBufferCreateFromBuffer(HNBuffer hSrcBuffer, NSizeType offset, NSizeType
size, HNBuffer * phBuffer);
```

Parameters

Parameters	Description
HNBuffer hSrcBuffer	[in] A handle to the source NBuffer [313].
NSizeType offset	[in] The byte position in the buffer at which to start the unmanaged memory stream.
NSizeType size	[in] The size of the NBuffer [313]. This value must be greater than zero. The default size is 4096.
HNBuffer * phBuffer	[out] A pointer to the created NBuffer [313].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NBuffer Unit [313]

7.1.2.4.1.8 NBufferCreateFromPtr Function

Creates a new NBuffer [313] using the specified memory location and size.

C++

```
NResult N_API NBufferCreateFromPtr(void * ptr, NSizeType size, NBool ownsPtr, HNBuffer *
phBuffer);
```

Parameters

Parameters	Description
void * ptr	[in] A pointer to the source memory location.
NSizeType size	[in] The size of the NBuffer [313]. This value must be greater than zero. The default size is 4096.
NBool ownsPtr	[in] NTrue [1064] to reliably release the pointer during the finalization phase; NFalse [1041] to prevent reliable release (not recommended).
HNBuffer * phBuffer	[out] A pointer to the created NBuffer [313].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.1.9 NBufferCreateFromPtrEx Function

Creates a new NBuffer [313] using the specified size and a specified location of specified type.

C++

```
NResult N_API NBufferCreateFromPtrEx(void * ptr, NSizeType size, NMemoryType ptrType,
HNBuffer * phBuffer);
```

Parameters

Parameters	Description
void * ptr	[in] A pointer to the source memory location.
NSizeType size	[in] The size of the NBuffer [313]. This value must be greater than zero. The default size is 4096.
NMemoryType ptrType	[in] The type of source memory location.
HNBuffer * phBuffer	[out] A pointer to the created NBuffer [313].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.1.10 NBufferCreateFromPtrWithFree Function

Creates a new NBuffer [313] using the specified memory location and size; frees the memory in the source location.

C++

```
NResult N_API NBufferCreateFromPtrWithFree(void * ptr, NSizeType size, HNCallback hFree,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
HNBuffer * phBuffer);
```

Parameters

Parameters	Description
void * ptr	[in] A pointer to the source memory location.
NSizeType size	[in] The size of the NBuffer [313]. This value must be greater than zero. The default size is 4096.
HNCallback hFree	[in] A callback called when NBuffer [313] is freed.
HNBuffer * phBuffer	[out] A pointer to the created NBuffer [313].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.1.11 NBufferCreateFromPtrWithFreeProc Function

Creates a new NBuffer [313] using the specified memory location and size; frees the memory in the source location.

C++

```
NResult N_API NBufferCreateFromPtrWithFreeProc(void * ptr, NSizeType size, NPointerFreeProc
pFree, void * pFreeParam, HNBuffer * phBuffer);
```

Parameters

Parameters	Description
void * ptr	[in] A pointer to the source memory location.
NSizeType size	[in] The size of the NBuffer [313]. This value must be greater than zero. The default size is 4096.
NPointerFreeProc pFree	[in] A callback called when NBuffer [313] is freed.
void * pFreeParam	[in] A pointer to user data provided to NPointerFreeProc.
HNBuffer * phBuffer	[out] A pointer to the created NBuffer [313].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.1.12 NBufferGetEmpty Function

Returns an empty buffer.

C++

```
NResult N_API NBufferGetEmpty(HNBuffer * phValue);
```

Parameters

Parameters	Description
HNBuffer * phValue	[out] A pointer to empty NBuffer [313].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NBuffer Unit [313]

7.1.2.4.1.13 NBufferGetPtr Function

Reads the bytes from the NBuffer [313] and returns them as a pointer.

C++

```
NResult N_API NBufferGetPtr(HNBuffer hBuffer, void * * pValue);
```

Parameters

Parameters	Description
HNBuffer hBuffer	[in] A handle to NBuffer [313].
void * * pValue	[out] A pointer to the NBuffer [313].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.1.14 NBufferGetSize Function

Returns the size of the NBuffer [313] in bytes.

C++

```
NResult N_API NBufferGetSize(HNBuffer hBuffer, NSizeType * pValue);
```

Parameters

Parameters	Description
HNBuffer hBuffer	[in] A handle to NBuffer [313].
NSizeType * pValue	[out] A pointer to a positive NInt32 [1006] value greater than 0 indicating the buffer size.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.1.15 NBufferToPtr Function

Copies the memory contained in NBuffer [313] and returns it as a pointer.

C++

```
NResult N_API NBufferToPtr(HNBuffer hBuffer, NSizeType * pBufferSize, void * * ppBuffer);
```

Parameters

Parameters	Description
HNBuffer hBuffer	[in] A handle to NBuffer [313].
NSizeType * pBufferSize	[out] A pointer to a positive NInt32 [1006] value greater than 0 indicating the buffer size.
void * * ppBuffer	[out] A pointer to the returned memory.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.4.2 Macros

7.1.2.4.2.1 NBufferCreateFromConstPtr Macro

Creates a new NBuffer [313] using the specified memory location and size.

C++

```
#define NBufferCreateFromConstPtr(ptr, size, phBuffer) NBufferCreateFromPtr((void *)ptr,
size, NFalse, phBuffer)
```

Parameters

Parameters	Description
ptr	[in] A pointer to the source memory location.
size	[in] The size of the NBuffer [313]. This value must be greater than zero. The default size is 4096.
phBuffer	[out] A pointer to the created NBuffer [313].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBuffer Unit [313]

7.1.2.5 NBufferedStream Unit

Provides functions for adding a buffering layer to read and write operations on another stream.

Functions

	Name	Description
	NBufferedStreamCreate [320]	Creates a new NBufferedStream with a default buffer size of 4096 bytes.
	NBufferedStreamCreateWithBufferSize [321]	Creates a new NBufferedStream with the specified buffer size.

7.1.2.5.1 Functions

7.1.2.5.1.1 NBufferedStreamCreate Function

Creates a new NBufferedStream [320] with a default buffer size of 4096 bytes.

C++

```
NResult N_API NBufferedStreamCreate(HNStream hInnerStream, HNBufferedStream * phStream);
```

Parameters

Parameters	Description
HNStream hInnerStream	[in] A handle to the current stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNBufferedStream * phStream	[out] A pointer to the created NBufferedStream [320].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBufferedStream Unit [320]

7.1.2.5.1.2 NBufferedStreamCreateWithBufferSize Function

Creates a new NBufferedStream [320] with the specified buffer size.

C++

```
NResult N_API NBufferedStreamCreateWithBufferSize(HNStream hInnerStream, NSizeType
bufferSize, HNBufferedStream * phStream);
```

Parameters

Parameters	Description
HNStream hInnerStream	[in] A handle to the current stream.
NSizeType bufferSize	[in] A buffer size in bytes.
HNBufferedStream * phStream	[out] A pointer to the created NBufferedStream [320].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBufferedStream Unit [320]

7.1.2.6 NCallback Unit

Provides function callback functionality.

Files

Name	Description
NCallback.h [332]	Header file for the NCallback unit.

Functions

	Name	Description
	NCallbackClone [322]	Creates a copy of the specified callback.
	NCallbackCreateCustomRaw [322]	Creates a callback for the specified process with additional parameters.
	NCallbackCreateRaw [323]	Creates a callback for the specified process.
	NCallbackEquals [323]	Checks whether the specified callback equals the other specified callback.
	NCallbackFree [324]	Frees the specified callback.
	NCallbackGetHashCode [324]	Retrieves the hash code of the specified callback.
	NCallbackGetParam [324]	Retrieves the parameters of the specified callback.
	NCallbackGetProcRaw [325]	Retrieves the process that invokes the specified callback.
	NCallbackSet [325]	Sets the value of the specified variable to the value of the specified callback.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Macros

Name	Description
NCallbackCreate [325]	Creates a callback for the specified process.
NCallbackCreateCustom [326]	Creates a callback for the specified process with additional parameters.
NCallbackGetProc [327]	Retrieves the process that invokes the specified callback.
NCallbackInvoke0 [327]	Invokes the specified callback.
NCallbackInvoke1 [327]	Invokes the specified callback with specified arguments.
NCallbackInvoke2 [328]	Invokes the specified callback with specified arguments.
NCallbackInvoke3 [328]	Invokes the specified callback with specified arguments.
NCallbackInvoke4 [328]	Invokes the specified callback with specified arguments.
NCallbackInvoke5 [329]	Invokes the specified callback with specified arguments.
NCallbackInvoke6 [329]	Invokes the specified callback with specified arguments.
NCallbackInvoke7 [330]	Invokes the specified callback with specified arguments.
NCallbackInvoke8 [330]	Invokes the specified callback with specified arguments.
NCallbackInvoke9 [331]	Invokes the specified callback with specified arguments.
NCallbackInvokeRaw [331]	Invokes the specified callback with specified arguments.

7.1.2.6.1 Functions

7.1.2.6.1.1 NCallbackClone Function

Creates a copy of the specified callback.

C++

```
NResult N_API NCallbackClone(HNCallback hCallback, HNCallback * phClonedCallback);
```

Parameters

Parameters	Description
HNCallback hCallback	[in] Handle to the callback to be cloned.
HNCallback * phClonedCallback	[out] Pointer to handle to the copy of the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCallback Unit [321]

7.1.2.6.1.2 NCallbackCreateCustomRaw Function

Creates a callback for the specified process with additional parameters.

C++

```
NResult N_API NCallbackCreateCustomRaw(void * pProc, void * pParam, NPointerFreeProc pFree,
NPointerGetHashCodeProc pGetHashCode, NPointerEqualsProc pEquals, HNCallback * phCallback);
```

Parameters

Parameters	Description
void * pProc	[in/out] Pointer to the process.
void * pParam	[in/out] Pointer to the callback parameters.
NPointerFreeProc pFree	[in] Pointer to the process that frees the callback.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPointerGetHashCodeProc pGetHashCode	[in] Pointer to the process that retrieves the hash code of the callback.
NPointerEqualsProc pEquals	[in] Pointer to the process that retrieves the compares two callbacks.
HNCallback * phCallback	[out] Pointer to a handle to the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

Raw functions (i.e. NCallbackCreateRaw [323]) represent the functions themselves.

Non-raw functions (i.e. NCallbackCreate [325]) represent macros that call those functions for the convenience of using them in C++.

Module

NCallback Unit [321]

7.1.2.6.1.3 NCallbackCreateRaw Function

Creates a callback for the specified process.

C++

```
NResult N_API NCallbackCreateRaw(void * pProc, void * pParam, HNCallback * phCallback);
```

Parameters

Parameters	Description
void * pProc	[in/out] Pointer to the process.
void * pParam	[in/out] Pointer to the callback parameters.
HNCallback * phCallback	[out] Pointer to a handle to the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

Raw functions (i.e. NCallbackCreateRaw) represent the functions themselves.

Non-raw functions (i.e. NCallbackCreate [325]) represent macros that call those functions for the convenience of using them in C++.

Module

NCallback Unit [321]

7.1.2.6.1.4 NCallbackEquals Function

Checks whether the specified callback equals the other specified callback.

C++

```
NResult N_API NCallbackEquals(HNCallback hCallback, HNCallback hOtherCallback, NBool *
pResult);
```

Parameters

Parameters	Description
HNCallback hCallback	[in] Handle to the first callback.
HNCallback hOtherCallback	[in] Handle to the second callback.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBool * pResult	[out] Pointer to NTrue [1064] if the callbacks are equal, else pointer to NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCallback Unit [321]

7.1.2.6.1.5 NCallbackFree Function

Frees the specified callback.

C++

```
NResult N_API NCallbackFree(HNCallback hCallback);
```

Parameters

Parameters	Description
HNCallback hCallback	[in] Handle to the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCallback Unit [321]

7.1.2.6.1.6 NCallbackGetHashCode Function

Retrieves the hash code of the specified callback.

C++

```
NResult N_API NCallbackGetHashCode(HNCallback hCallback, NInt * pValue);
```

Parameters

Parameters	Description
HNCallback hCallback	[in] Handle to the callback.
NInt * pValue	[out] Pointer to the hash code of the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCallback Unit [321]

7.1.2.6.1.7 NCallbackGetParam Function

Retrieves the parameters of the specified callback.

C++

```
void * N_API_PTR_RET NCallbackGetParam(HNCallback hCallback);
```

Parameters

Parameters	Description
HNCallback hCallback	[in] Handle to the callback.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

An array containing the parameters of the callback.

Module

NCallback Unit [321]

7.1.2.6.1.8 NCallbackGetProcRaw Function

Retrieves the process that invokes the specified callback.

C++

```
void * N_API_PTR_RET NCallbackGetProcRaw(HNCallback hCallback);
```

Parameters

Parameters	Description
HNCallback hCallback	[in] Handle to the callback.

Returns

The process that invokes the callback.

Notes

Raw functions (i.e. NCallbackCreateRaw [323]) represent the functions themselves.

Non-raw functions (i.e. NCallbackCreate [325]) represent macros that call those functions for the convenience of using them in C++.

Module

NCallback Unit [321]

7.1.2.6.1.9 NCallbackSet Function

Sets the value of the specified variable to the value of the specified callback.

C++

```
NResult N_API NCallbackSet(HNCallback hCallback, HNCallback * phVariable);
```

Parameters

Parameters	Description
HNCallback hCallback	[in] Handle to the callback.
HNCallback * phVariable	[out] Pointer to a handle to a variable that will hold the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCallback Unit [321]

7.1.2.6.2 Macros

7.1.2.6.2.1 NCallbackCreate Macro

Creates a callback for the specified process.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NCallbackCreate(pProc, pParam, phCallback) NCallbackCreateRaw((void
*)(NSizeType)pProc, pParam, phCallback)
```

Parameters

Parameters	Description
pProc	[in/out] Pointer to the process.
pParam	[in/out] Pointer to the callback parameters.
phCallback	[out] Pointer to a handle to the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

Raw functions (i.e. NCallbackCreateRaw [323]) represent the functions themselves.

Non-raw functions (i.e. NCallbackCreate) represent macros that call those functions for the convenience of using them in C++.

Module

NCallback Unit [321]

7.1.2.6.2.2 NCallbackCreateCustom Macro

Creates a callback for the specified process with additional parameters.

C++

```
#define NCallbackCreateCustom(pProc, pParam, pFree, pGetHashCode, pEquals, phCallback)
NCallbackCreateCustomRaw((void *)(NSizeType)pProc, pParam, pFree, pGetHashCode, pEquals,
phCallback)
```

Parameters

Parameters	Description
pProc	[in/out] Pointer to the process.
pParam	[in/out] Pointer to the callback parameters.
pFree	[in] Pointer to the process that frees the callback.
pGetHashCode	[in] Pointer to the process that retrieves the hash code of the callback.
pEquals	[in] Pointer to the process that retrieves the compares two callbacks.
phCallback	[out] Pointer to a handle to the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

Raw functions (i.e. NCallbackCreateRaw [323]) represent the functions themselves.

Non-raw functions (i.e. NCallbackCreate [325]) represent macros that call those functions for the convenience of using them in C++.

Module

NCallback Unit [321]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.6.2.3 NCallbackGetProc Macro

Retrieves the process that invokes the specified callback.

C++

```
#define NCallbackGetProc(callbackType, hCallback)
((callbackType)(NSizeType)NCallbackGetProcRaw(hCallback))
```

Parameters

Parameters	Description
hCallback	[in] Handle to the callback.

Returns

The process that invokes the callback.

Module

NCallback Unit [321]

7.1.2.6.2.4 NCallbackInvoke0 Macro

Invokes the specified callback.

C++

```
#define NCallbackInvoke0(callbackType, hCallback) NCallbackInvokeRaw(callbackType,
hCallback, (NCallbackGetParam(hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
hCallback	[in] Handle to the callback.

Returns

The process that invokes the callback.

Module

NCallback Unit [321]

7.1.2.6.2.5 NCallbackInvoke1 Macro

Invokes the specified callback with specified arguments.

C++

```
#define NCallbackInvoke1(callbackType, hCallback, arg1) NCallbackInvokeRaw(callbackType,
hCallback, ((arg1), NCallbackGetParam(hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
hCallback	[in] Handle to the callback.
arg1	[in] Callback argument.

Returns

The process that invokes the callback.

Module

NCallback Unit [321]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.6.2.6 NCallbackInvoke2 Macro

Invokes the specified callback with specified arguments.

C++

```
#define NCallbackInvoke2(callbackType, hCallback, arg1, arg2)
NCallbackInvokeRaw(callbackType, hCallback, ((arg1), (arg2), NCallbackGetParam(hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
hCallback	[in] Handle to the callback.
arg1	[in] Callback argument.
arg2	[in] Callback argument.

Returns

The process that invokes the callback.

Module

NCallback Unit [321]

7.1.2.6.2.7 NCallbackInvoke3 Macro

Invokes the specified callback with specified arguments.

C++

```
#define NCallbackInvoke3(callbackType, hCallback, arg1, arg2, arg3)
NCallbackInvokeRaw(callbackType, hCallback, ((arg1), (arg2), (arg3),
NCallbackGetParam(hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
hCallback	[in] Handle to the callback.
arg1	[in] Callback argument.
arg2	[in] Callback argument.
arg3	[in] Callback argument.

Returns

The process that invokes the callback.

Module

NCallback Unit [321]

7.1.2.6.2.8 NCallbackInvoke4 Macro

Invokes the specified callback with specified arguments.

C++

```
#define NCallbackInvoke4(callbackType, hCallback, arg1, arg2, arg3, arg4)
NCallbackInvokeRaw(callbackType, hCallback, ((arg1), (arg2), (arg3), (arg4),
NCallbackGetParam(hCallback)))
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
hCallback	[in] Handle to the callback.
arg1	[in] Callback argument.
arg2	[in] Callback argument.
arg3	[in] Callback argument.
arg4	[in] Callback argument.

Returns

The process that invokes the callback.

Module

NCallback Unit [321]

7.1.2.6.2.9 NCallbackInvoke5 Macro

Invokes the specified callback with specified arguments.

C++

```
#define NCallbackInvoke5(callbackType, hCallback, arg1, arg2, arg3, arg4, arg5)
NCallbackInvokeRaw(callbackType, hCallback, ((arg1), (arg2), (arg3), (arg4), (arg5),
NCallbackGetParam(hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
hCallback	[in] Handle to the callback.
arg1	[in] Callback argument.
arg2	[in] Callback argument.
arg3	[in] Callback argument.
arg4	[in] Callback argument.
arg5	[in] Callback argument.

Returns

The process that invokes the callback.

Module

NCallback Unit [321]

7.1.2.6.2.10 NCallbackInvoke6 Macro

Invokes the specified callback with specified arguments.

C++

```
#define NCallbackInvoke6(callbackType, hCallback, arg1, arg2, arg3, arg4, arg5, arg6)
NCallbackInvokeRaw(callbackType, hCallback, ((arg1), (arg2), (arg3), (arg4), (arg5),
(arg6), NCallbackGetParam(hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
hCallback	[in] Handle to the callback.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

arg1	[in] Callback argument.
arg2	[in] Callback argument.
arg3	[in] Callback argument.
arg4	[in] Callback argument.
arg5	[in] Callback argument.
arg6	[in] Callback argument.

Returns

The process that invokes the callback.

Module

NCallback Unit [321]

7.1.2.6.2.11 NCallbackInvoke7 Macro

Invokes the specified callback with specified arguments.

C++

```
#define NCallbackInvoke7(callbackType, hCallback, arg1, arg2, arg3, arg4, arg5, arg6, arg7)
NCallbackInvokeRaw(callbackType, hCallback, ((arg1), (arg2), (arg3), (arg4), (arg5),
(arg6), (arg7), NCallbackGetParam(hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
hCallback	[in] Handle to the callback.
arg1	[in] Callback argument.
arg2	[in] Callback argument.
arg3	[in] Callback argument.
arg4	[in] Callback argument.
arg5	[in] Callback argument.
arg6	[in] Callback argument.
arg7	[in] Callback argument.

Returns

The process that invokes the callback.

Module

NCallback Unit [321]

7.1.2.6.2.12 NCallbackInvoke8 Macro

Invokes the specified callback with specified arguments.

C++

```
#define NCallbackInvoke8(callbackType, hCallback, arg1, arg2, arg3, arg4, arg5, arg6, arg7,
arg8) NCallbackInvokeRaw(callbackType, hCallback, ((arg1), (arg2), (arg3), (arg4), (arg5),
(arg6), (arg7), (arg8), NCallbackGetParam(hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
hCallback	[in] Handle to the callback.
arg1	[in] Callback argument.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

arg2	[in] Callback argument.
arg3	[in] Callback argument.
arg4	[in] Callback argument.
arg5	[in] Callback argument.
arg6	[in] Callback argument.
arg7	[in] Callback argument.
arg8	[in] Callback argument.

Returns

The process that invokes the callback.

Module

NCallback Unit [321]

7.1.2.6.2.13 NCallbackInvoke9 Macro

Invokes the specified callback with specified arguments.

C++

```
#define NCallbackInvoke9(callbackType, hCallback, arg1, arg2, arg3, arg4, arg5, arg6, arg7,
arg8, arg9) NCallbackInvokeRaw(callbackType, hCallback, ((arg1), (arg2), (arg3), (arg4),
(arg5), (arg6), (arg7), (arg8), (arg9), NCallbackGetParam(hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
hCallback	[in] Handle to the callback.
arg1	[in] Callback argument.
arg2	[in] Callback argument.
arg3	[in] Callback argument.
arg4	[in] Callback argument.
arg5	[in] Callback argument.
arg6	[in] Callback argument.
arg7	[in] Callback argument.
arg8	[in] Callback argument.
arg9	[in] Callback argument.

Returns

The process that invokes the callback.

Module

NCallback Unit [321]

7.1.2.6.2.14 NCallbackInvokeRaw Macro

Invokes the specified callback with specified arguments.

C++

```
#define NCallbackInvokeRaw(callbackType, hCallback, args) (NCallbackGetProc(callbackType,
hCallback)args)
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

hCallback	[in] Handle to the callback.
args	[in] Callback arguments.

Returns

The process that invokes the callback.

Notes

Raw functions (i.e. NCallbackCreateRaw [323]) represent the functions themselves.

Non-raw functions (i.e. NCallbackCreate [325]) represent macros that call those functions for the convenience of using them in C++.

Module

NCallback Unit [321]

7.1.2.6.3 Files

7.1.2.6.3.1 NCallback.h

Header file for the NCallback [321] unit.

Functions

	Name	Description
	NCallbackClone [322]	Creates a copy of the specified callback.
	NCallbackCreateCustomRaw [322]	Creates a callback for the specified process with additional parameters.
	NCallbackCreateRaw [323]	Creates a callback for the specified process.
	NCallbackEquals [323]	Checks whether the specified callback equals the other specified callback.
	NCallbackFree [324]	Frees the specified callback.
	NCallbackGetHashCode [324]	Retrieves the hash code of the specified callback.
	NCallbackGetParam [324]	Retrieves the parameters of the specified callback.
	NCallbackGetProcRaw [325]	Retrieves the process that invokes the specified callback.
	NCallbackSet [325]	Sets the value of the specified variable to the value of the specified callback.

Macros

Name	Description
NCallbackCreate [325]	Creates a callback for the specified process.
NCallbackCreateCustom [326]	Creates a callback for the specified process with additional parameters.
NCallbackGetProc [327]	Retrieves the process that invokes the specified callback.
NCallbackInvoke0 [327]	Invokes the specified callback.
NCallbackInvoke1 [327]	Invokes the specified callback with specified arguments.
NCallbackInvoke2 [328]	Invokes the specified callback with specified arguments.
NCallbackInvoke3 [328]	Invokes the specified callback with specified arguments.
NCallbackInvoke4 [328]	Invokes the specified callback with specified arguments.
NCallbackInvoke5 [329]	Invokes the specified callback with specified arguments.
NCallbackInvoke6 [329]	Invokes the specified callback with specified arguments.
NCallbackInvoke7 [330]	Invokes the specified callback with specified arguments.
NCallbackInvoke8 [330]	Invokes the specified callback with specified arguments.
NCallbackInvoke9 [331]	Invokes the specified callback with specified arguments.
NCallbackInvokeRaw [331]	Invokes the specified callback with specified arguments.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NCallback Unit [321]

7.1.2.7 NConsole Unit

Represents the standard input, output, and error streams for console applications.

Functions

	Name	Description
	NConsoleGetError [335]	Gets the standard error output stream.
	NConsoleGetIn [335]	Gets the standard input stream.
	NConsoleGetOut [335]	Gets the standard output stream.
	NConsoleReadChar [336]	Reads the next character from the standard input stream.
	NConsoleReadLine [336]	Reads the next line from the standard input stream.
	NConsoleSetError [336]	Sets the standard error output stream.
	NConsoleSetIn [336]	Sets the standard input stream.
	NConsoleSetOut [337]	Sets the standard output stream.
	NConsoleWrite [337]	Writes the specified character to the standard output stream.
	NConsoleWriteBoolean [337]	Writes the specified boolean value to the standard output stream.
	NConsoleWriteBooleanLine [338]	Writes the specified boolean value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteChar [338]	Writes the specified char value to the standard output stream.
	NConsoleWriteCharLine [338]	Writes the specified char value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteChars [339]	Writes the specified char array value to the standard output stream.
	NConsoleWriteCharsLine [339]	Writes the specified char array value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteDouble [339]	Writes the specified double value to the standard output stream.
	NConsoleWriteDoubleLine [340]	Writes the specified double value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteEmptyLine [340]	Writes an empty line to the standard output stream.
	NConsoleWriteFormat [340]	Writes the specified format string to the standard output stream.
	NConsoleWriteFormatLine [341]	Writes the specified format string, followed by the current line terminator, to the standard output stream.
	NConsoleWriteFormatLineVA [341]	Writes the specified format string, followed by the current line terminator, to the standard output stream.
	NConsoleWriteFormatVA [341]	Writes the specified format string to the standard output stream.
	NConsoleWriteInt32 [342]	Writes the specified Int32 value to the standard output stream.
	NConsoleWriteInt32Line [342]	Writes the specified Int32 value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteInt64 [342]	Writes the specified Int64 value to the standard output stream.
	NConsoleWriteInt64Line [343]	Writes the specified Int64 value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteLine [343]	Writes the specified character to the standard output stream.
	NConsoleWriteObject [343]	Writes the specified object to the standard output stream.
	NConsoleWriteObjectLine [343]	Writes the specified object, followed by the current line terminator, to the standard output stream.
	NConsoleWritePointer [344]	Writes the specified pointer value to the standard output stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NConsoleWritePointerLine [344]	Writes the specified pointer value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteResult [344]	Writes the specified NResult [1007] value to the standard output stream.
	NConsoleWriteResultLine [345]	Writes the specified NResult [1007] value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteSingle [345]	Writes the specified single value to the standard output stream.
	NConsoleWriteSingleLine [345]	Writes the specified single value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteSizeType [346]	Writes the specified SizeType value to the standard output stream.
	NConsoleWriteSizeTypeLine [346]	Writes the specified SizeType value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteSSizeType [346]	Writes the specified SSizeType value to the standard output stream.
	NConsoleWriteSSizeTypeLine [347]	Writes the specified SSizeType value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteUInt32 [347]	Writes the specified UInt32 value to the standard output stream.
	NConsoleWriteUInt32Line [347]	Writes the specified UInt32 value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteUInt64 [347]	Writes the specified UInt64 value to the standard output stream.
	NConsoleWriteUInt64Line [348]	Writes the specified UInt64 value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteValue [348]	Writes the specified value to the standard output stream.
	NConsoleWriteValueLine [348]	Writes the specified value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteValueLineP [349]	Writes the specified value, followed by the current line terminator, to the standard output stream.
	NConsoleWriteValueP [349]	Writes the specified value to the standard output stream.

Macros

Name	Description
NConsoleReadChar [350]	Picks a Unicode (W) or ASCII (A) version of the NConsoleReadChar function.
NConsoleWrite [350]	Picks a Unicode (W) or ASCII (A) version of the NConsoleWrite function.
NConsoleWriteChar [350]	Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteChar function.
NConsoleWriteCharLine [350]	Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteCharLine function.
NConsoleWriteChars [350]	Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteChars function.
NConsoleWriteCharsLine [350]	Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteCharsLine function.
NConsoleWriteFormat [351]	Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteFormat function.
NConsoleWriteFormatLine [351]	Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteFormatLine function.
NConsoleWriteFormatLineVA [351]	Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteFormatLineVA function.
NConsoleWriteFormatVA [351]	Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteFormatVA function.
NConsoleWriteLine [351]	Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteLine function.

7.1.2.7.1 Functions

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.7.1.1 NConsoleGetError Function

Gets the standard error output stream.

C++

```
NResult N_API NConsoleGetError(HNTextWriter * phValue);
```

Parameters

Parameters	Description
HNTextWriter * phValue	[out] Pointer to a handle to NTextWriter [833] that represents the standard error output stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.2 NConsoleGetIn Function

Gets the standard input stream.

C++

```
NResult N_API NConsoleGetIn(HNTextReader * phValue);
```

Parameters

Parameters	Description
HNTextReader * phValue	[out] Pointer to a handle to NTextWriter [833] that represents the standard input stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.3 NConsoleGetOut Function

Gets the standard output stream.

C++

```
NResult N_API NConsoleGetOut(HNTextWriter * phValue);
```

Parameters

Parameters	Description
HNTextWriter * phValue	[out] Pointer to a handle to NTextWriter [833] that represents the standard output stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.7.1.4 NConsoleReadChar Function

Reads the next character from the standard input stream.

C++

```
NResult N_API NConsoleReadChar(NInt * pValue);
```

Parameters

Parameters	Description
NInt * pValue	[out] Pointer to the next character from the input stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.5 NConsoleReadLine Function

Reads the next line from the standard input stream.

C++

```
NResult N_API NConsoleReadLine(HNString * phValue);
```

Parameters

Parameters	Description
HNString * phValue	[out] Pointer to a handle to the next line from the standard input stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.6 NConsoleSetError Function

Sets the standard error output stream.

C++

```
NResult N_API NConsoleSetError(HNTextWriter hValue);
```

Parameters

Parameters	Description
HNTextWriter hValue	[in] A handle to NTextWriter [833] that represents the standard error output stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.7 NConsoleSetIn Function

Sets the standard input stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NConsoleSetIn(HNTextReader hValue);
```

Parameters

Parameters	Description
HNTextReader hValue	[in] A handle to NTextWriter [833] that represents the standard input stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.8 NConsoleSetOut Function

Sets the standard output stream.

C++

```
NResult N_API NConsoleSetOut(HNTextWriter hValue);
```

Parameters

Parameters	Description
HNTextWriter hValue	[in] A handle to NTextWriter [833] that represents the standard output stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.9 NConsoleWrite Function

Writes the specified character to the standard output stream.

C++

```
NResult N_API NConsoleWrite(const NChar * szValue);
```

Parameters

Parameters	Description
const NChar * szValue	[in] Pointer to the value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.10 NConsoleWriteBoolean Function

Writes the specified boolean value to the standard output stream.

C++

```
NResult N_API NConsoleWriteBoolean(NBool value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NBool value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.11 NConsoleWriteBooleanLine Function

Writes the specified boolean value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteBooleanLine(NBool value);
```

Parameters

Parameters	Description
NBool value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.12 NConsoleWriteChar Function

Writes the specified char value to the standard output stream.

C++

```
NResult N_API NConsoleWriteChar(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.13 NConsoleWriteCharLine Function

Writes the specified char value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteCharLine(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] The value to write.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.14 NConsoleWriteChars Function

Writes the specified char array value to the standard output stream.

C++

```
NResult N_API NConsoleWriteChars(const NChar * arValue, NInt valueLength);
```

Parameters

Parameters	Description
const NChar * arValue	[in] The value to write.
NInt valueLength	[in] Length of char array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.15 NConsoleWriteCharsLine Function

Writes the specified char array value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteCharsLine(const NChar * arValue, NInt valueLength);
```

Parameters

Parameters	Description
const NChar * arValue	[in] The value to write.
NInt valueLength	[in] Length of char array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.16 NConsoleWriteDouble Function

Writes the specified double value to the standard output stream.

C++

```
NResult N_API NConsoleWriteDouble(NDouble value);
```

Parameters

Parameters	Description
NDouble value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NConsole Unit [333]

7.1.2.7.1.17 NConsoleWriteDoubleLine Function

Writes the specified double value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteDoubleLine(NDouble value);
```

Parameters

Parameters	Description
NDouble value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.18 NConsoleWriteEmptyLine Function

Writes an empty line to the standard output stream.

C++

```
NResult N_API NConsoleWriteEmptyLine();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.19 NConsoleWriteFormat Function

Writes the specified format string to the standard output stream.

C++

```
NResult N_API NConsoleWriteFormat(const NChar * szFormat, ... multipleParameters);
```

Parameters

Parameters	Description
const NChar * szFormat	[in] Pointer to the format string to write.
... multipleParameters	[in] Format parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NConsole Unit [333]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.7.1.20 NConsoleWriteFormatLine Function

Writes the specified format string, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteFormatLine(const NChar * szFormat, ... multipleParameters);
```

Parameters

Parameters	Description
const NChar * szFormat	[in] Pointer to the format string to write.
... multipleParameters	[in] Format parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NConsole Unit [333]

7.1.2.7.1.21 NConsoleWriteFormatLineVA Function

Writes the specified format string, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteFormatLineVA(const NChar * szFormat, va_list args);
```

Parameters

Parameters	Description
const NChar * szFormat	[in] Pointer to the format string to write.
va_list args	[in] Format arguments.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NConsole Unit [333]

7.1.2.7.1.22 NConsoleWriteFormatVA Function

Writes the specified format string to the standard output stream.

C++

```
NResult N_API NConsoleWriteFormatVA(const NChar * szFormat, va_list args);
```

Parameters

Parameters	Description
const NChar * szFormat	[in] Pointer to the format string to write.
va_list args	[in] Format arguments.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NConsole Unit [333]

7.1.2.7.1.23 NConsoleWriteInt32 Function

Writes the specified Int32 value to the standard output stream.

C++

```
NResult N_API NConsoleWriteInt32(NInt value);
```

Parameters

Parameters	Description
NInt value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.24 NConsoleWriteInt32Line Function

Writes the specified Int32 value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteInt32Line(NInt value);
```

Parameters

Parameters	Description
NInt value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.25 NConsoleWriteInt64 Function

Writes the specified Int64 value to the standard output stream.

C++

```
NResult N_API NConsoleWriteInt64(NLong value);
```

Parameters

Parameters	Description
NLong value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.7.1.26 NConsoleWriteInt64Line Function

Writes the specified Int64 value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteInt64Line(NLong value);
```

Parameters

Parameters	Description
NLong value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.27 NConsoleWriteLine Function

Writes the specified character to the standard output stream.

C++

```
NResult N_API NConsoleWriteLine(const NChar * szValue);
```

Parameters

Parameters	Description
const NChar * szValue	[in] Pointer to the value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.28 NConsoleWriteObject Function

Writes the specified object to the standard output stream.

C++

```
NResult N_API NConsoleWriteObject(HNObject hObject);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the object to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.29 NConsoleWriteObjectLine Function

Writes the specified object, followed by the current line terminator, to the standard output stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NConsoleWriteObjectLine(HNObject hObject);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the object to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.30 NConsoleWritePointer Function

Writes the specified pointer value to the standard output stream.

C++

```
NResult N_API NConsoleWritePointer(const void * value);
```

Parameters

Parameters	Description
const void * value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.31 NConsoleWritePointerLine Function

Writes the specified pointer value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWritePointerLine(const void * value);
```

Parameters

Parameters	Description
const void * value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.32 NConsoleWriteResult Function

Writes the specified NResult [1007] value to the standard output stream.

C++

```
NResult N_API NConsoleWriteResult(NResult value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NResult value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.33 NConsoleWriteResultLine Function

Writes the specified NResult [1007] value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteResultLine(NResult value);
```

Parameters

Parameters	Description
NResult value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.34 NConsoleWriteSingle Function

Writes the specified single value to the standard output stream.

C++

```
NResult N_API NConsoleWriteSingle(NFloat value);
```

Parameters

Parameters	Description
NFloat value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.35 NConsoleWriteSingleLine Function

Writes the specified single value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteSingleLine(NFloat value);
```

Parameters

Parameters	Description
NFloat value	[in] The value to write.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.36 NConsoleWriteSizeType Function

Writes the specified SizeType value to the standard output stream.

C++

```
NResult N_API NConsoleWriteSizeType(NSizeType value);
```

Parameters

Parameters	Description
NSizeType value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.37 NConsoleWriteSizeTypeLine Function

Writes the specified SizeType value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteSizeTypeLine(NSizeType value);
```

Parameters

Parameters	Description
NSizeType value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.38 NConsoleWriteSSizeType Function

Writes the specified SSizeType value to the standard output stream.

C++

```
NResult N_API NConsoleWriteSSizeType(NSSizeType value);
```

Parameters

Parameters	Description
NSSizeType value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.7.1.39 NConsoleWriteSSizeTypeLine Function

Writes the specified SSizeType value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteSSizeTypeLine(NSSizeType value);
```

Parameters

Parameters	Description
NSSizeType value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.40 NConsoleWriteUInt32 Function

Writes the specified UInt32 value to the standard output stream.

C++

```
NResult N_API NConsoleWriteUInt32(NUInt value);
```

Parameters

Parameters	Description
NUInt value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.41 NConsoleWriteUInt32Line Function

Writes the specified UInt32 value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteUInt32Line(NUInt value);
```

Parameters

Parameters	Description
NUInt value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.42 NConsoleWriteUInt64 Function

Writes the specified UInt64 value to the standard output stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NConsoleWriteUInt64(NULong value);
```

Parameters

Parameters	Description
NULong value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.43 NConsoleWriteUInt64Line Function

Writes the specified UInt64 value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteUInt64Line(NULong value);
```

Parameters

Parameters	Description
NULong value	[in] The value to write.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.44 NConsoleWriteValue Function

Writes the specified value to the standard output stream.

C++

```
NResult N_API NConsoleWriteValue(HNType hType, const void * pValue, NSizeType valueSize);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to a function that retrieves the type of the value.
const void * pValue	[in] Pointer to the value.
NSizeType valueSize	[in] Size of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.45 NConsoleWriteValueLine Function

Writes the specified value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteValueLine(HNType hType, const void * pValue, NSizeType
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
valueSize);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to a function that retrieves the type of the value.
const void * pValue	[in] Pointer to the value.
NSizeType valueSize	[in] Size of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.46 NConsoleWriteValueLineP Function

Writes the specified value, followed by the current line terminator, to the standard output stream.

C++

```
NResult N_API NConsoleWriteValueLineP(NTypeOfProc pTypeOf, const void * pValue, NSizeType
valueSize);
```

Parameters

Parameters	Description
NTypeOfProc pTypeOf	[in] Pointer to a function that retrieves the type of the value.
const void * pValue	[in] Pointer to the value.
NSizeType valueSize	[in] Size of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.1.47 NConsoleWriteValueP Function

Writes the specified value to the standard output stream.

C++

```
NResult N_API NConsoleWriteValueP(NTypeOfProc pTypeOf, const void * pValue, NSizeType
valueSize);
```

Parameters

Parameters	Description
NTypeOfProc pTypeOf	[in] Pointer to a function that retrieves the type of the value.
const void * pValue	[in] Pointer to the value.
NSizeType valueSize	[in] Size of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NConsole Unit [333]

7.1.2.7.2 Macros

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.7.2.1 NConsoleReadChar Macro

Picks a Unicode (W) or ASCII (A) version of the NConsoleReadChar function.

C++

```
#define NConsoleReadChar N_FUNC_AW(NConsoleReadChar)
```

Module

NConsole Unit [333]

7.1.2.7.2.2 NConsoleWrite Macro

Picks a Unicode (W) or ASCII (A) version of the NConsoleWrite function.

C++

```
#define NConsoleWrite N_FUNC_AW(NConsoleWrite)
```

Module

NConsole Unit [333]

7.1.2.7.2.3 NConsoleWriteChar Macro

Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteChar function.

C++

```
#define NConsoleWriteChar N_FUNC_AW(NConsoleWriteChar)
```

Module

NConsole Unit [333]

7.1.2.7.2.4 NConsoleWriteCharLine Macro

Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteCharLine function.

C++

```
#define NConsoleWriteCharLine N_FUNC_AW(NConsoleWriteCharLine)
```

Module

NConsole Unit [333]

7.1.2.7.2.5 NConsoleWriteChars Macro

Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteChars function.

C++

```
#define NConsoleWriteChars N_FUNC_AW(NConsoleWriteChars)
```

Module

NConsole Unit [333]

7.1.2.7.2.6 NConsoleWriteCharsLine Macro

Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteCharsLine function.

C++

```
#define NConsoleWriteCharsLine N_FUNC_AW(NConsoleWriteCharsLine)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NConsole Unit [333]

7.1.2.7.2.7 NConsoleWriteFormat Macro

Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteFormat function.

C++

```
#define NConsoleWriteFormat N_FUNC_AW(NConsoleWriteFormat)
```

Module

NConsole Unit [333]

7.1.2.7.2.8 NConsoleWriteFormatLine Macro

Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteFormatLine function.

C++

```
#define NConsoleWriteFormatLine N_FUNC_AW(NConsoleWriteFormatLine)
```

Module

NConsole Unit [333]

7.1.2.7.2.9 NConsoleWriteFormatLineVA Macro

Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteFormatLineVA function.

C++

```
#define NConsoleWriteFormatLineVA N_FUNC_AW(NConsoleWriteFormatLineVA)
```

Module

NConsole Unit [333]

7.1.2.7.2.10 NConsoleWriteFormatVA Macro

Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteFormatVA function.

C++

```
#define NConsoleWriteFormatVA N_FUNC_AW(NConsoleWriteFormatVA)
```

Module

NConsole Unit [333]

7.1.2.7.2.11 NConsoleWriteLine Macro

Picks a Unicode (W) or ASCII (A) version of the NConsoleWriteLine function.

C++

```
#define NConsoleWriteLine N_FUNC_AW(NConsoleWriteLine)
```

Module

NConsole Unit [333]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.8 NCore Unit

Provides infrastructure/basic functionality for Neurotechnology components.

Files

Name	Description
NCore.h [356]	Header file for the NCore [257] module. Provides infrastructure/basic functionality for Neurotechnology components.

Functions

	Name	Description
	NCoreAddErrorSuppressedCallback [352]	Adds a suppressed error callback to NCore [257].
	NCoreGetInfo [353]	Retrieves information about the library.
	NCoreAddErrorSuppressed [353]	Adds a suppressed error callback to NCore [257].
	NCoreGetCallStack [353]	Gets stack trace information.
	NCoreOnExitEx [354]	This function should be called on program's exit.
	NCoreOnStart [354]	This function should be called on program's start.
	NCoreOnThreadExit [354]	This function should be called when program's thread finished its task.
	NCoreOnThreadStart [354]	This function should be called when new thread is created.
	NCoreRemoveErrorSuppressedCallback [355]	Removes a suppressed error callback from NCore [257].
	NCoreRemoveErrorSuppressed [355]	Removes a suppressed error callback from NCore [257].
	NCoreModuleOf [355]	

Macros

Name	Description
NCoreOnExit [356]	This function should be called on programs exit.
NCoreOnThreadExitEx [356]	This function should be called when program's thread finished its task.

7.1.2.8.1 Functions

7.1.2.8.1.1 NCoreAddErrorSuppressedCallback Function

Adds a suppressed error callback to NCore [257].

C++

```
NResult N_API NCoreAddErrorSuppressedCallback(NErrorSuppressedCallback pCallback, void *
pParam);
```

Parameters

Parameters	Description
NErrorSuppressedCallback pCallback	[in] The callback to be added.
void * pParam	[in] A pointer to data provided for the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCore Unit [352]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.8.1.2 NCoreGetInfo Function

Retrieves information about the library.

C++

```
NResult N_API NCoreGetInfo(NLibraryInfo * pValue);
```

Parameters

Parameters	Description
NLibraryInfo * pValue	[out] Pointer to NLibraryInfo structure that receives library information.

Returns

If the function succeeds, the return value is N_OK [433]. Function result can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCore Unit [352]

7.1.2.8.1.3 NCoreAddErrorSuppressed Function

Adds a suppressed error callback to NCore [257].

C++

```
NResult N_API NCoreAddErrorSuppressed(HNCallback hCallback);
```

Parameters

Parameters	Description
pCallback	[in] The callback to be added.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCore Unit [352]

7.1.2.8.1.4 NCoreGetCallStack Function

Gets stack trace information.

C++

```
NResult N_NO_INLINE N_API NCoreGetCallStack(NInt framesToSkip, HNString * phValue);
```

Parameters

Parameters	Description
NInt framesToSkip	[int] A number of frames to skip.
HNString * phValue	[out] A pointer to the string containing stack trace information. This value can be an empty string ("").

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCore Unit [352]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.8.1.5 NCoreOnExitEx Function

This function should be called on program's exit.

C++

```
NResult N_API NCoreOnExitEx(NBool isProcessTermination);
```

Parameters

Parameters	Description
NBool isProcessTermination	[in] NTrue [1064] if the process exits; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCore Unit [352]

7.1.2.8.1.6 NCoreOnStart Function

This function should be called on program's start.

C++

```
NResult N_API NCoreOnStart();
```

Returns

If the function succeeds, the return value is N_OK [433]. Function result can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function works on Lib-No-Dll and VAR products.

Module

NCore Unit [352]

7.1.2.8.1.7 NCoreOnThreadExit Function

This function should be called when program's thread finished its task.

C++

```
NResult N_API NCoreOnThreadExit();
```

Returns

If the function succeeds, the return value is N_OK [433]. Function result can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function works on Lib-No-Dll and VAR products.

Module

NCore Unit [352]

7.1.2.8.1.8 NCoreOnThreadStart Function

This function should be called when new thread is created.

C++

```
NResult N_API NCoreOnThreadStart();
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

If the function succeeds, the return value is N_OK [433]. Function result can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function works on Lib-No-Dll and VAR products.

Module

NCore Unit [352]

7.1.2.8.1.9 NCoreRemoveErrorSuppressedCallback Function

Removes a suppressed error callback from NCore [257].

C++

```
NResult N_API NCoreRemoveErrorSuppressedCallback(NErrorSuppressedCallback pCallback, void *
pParam);
```

Parameters

Parameters	Description
NErrorSuppressedCallback pCallback	[in] The callback to be removed.
void * pParam	[in] A pointer to data provided for the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCore Unit [352]

7.1.2.8.1.10 NCoreRemoveErrorSuppressed Function

Removes a suppressed error callback from NCore [257].

C++

```
NResult N_API NCoreRemoveErrorSuppressed(HNCallback hCallback);
```

Parameters

Parameters	Description
pCallback	[in] The callback to be removed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCore Unit [352]

7.1.2.8.1.11 NCoreModuleOf

Returns

Handle of the module that can be used with functions from NModule [516] unit to get the information about NCore [257] module.

Module

NCore Unit [352]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.8.2 Macros

7.1.2.8.2.1 NCoreOnExit Macro

This function should be called on programs exit.

C++

```
#define NCoreOnExit NCoreOnExitEx(NFalse)
```

Notes

This function works on Lib-No-Dll and VAR products.

Module

NCore Unit [352]

7.1.2.8.2.2 NCoreOnThreadExitEx Macro

This function should be called when program's thread finished its task.

C++

```
#define NCoreOnThreadExitEx(isProcessTermination) NCoreOnThreadExit()
```

Parameters

Parameters	Description
isProcessTermination	NTrue [1064] if the process is terminated; else NFalse [1041].

Returns

If the function succeeds, the return value is N_OK [433]. Function result can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCore Unit [352]

7.1.2.8.3 Files

7.1.2.8.3.1 NCore.h

Header file for the NCore [257] module. Provides infrastructure/basic functionality for Neurotechnology components.

Functions

	Name	Description
	NCoreAddErrorSuppressed [353]	Adds a suppressed error callback to NCore [257].
	NCoreAddErrorSuppressedCallback [352]	Adds a suppressed error callback to NCore [257].
	NCoreGetCallStack [353]	Gets stack trace information.
	NCoreGetInfo [353]	Retrieves information about the library.
	NCoreOnExitEx [354]	This function should be called on program's exit.
	NCoreOnStart [354]	This function should be called on program's start.
	NCoreOnThreadExit [354]	This function should be called when program's thread finished its task.
	NCoreOnThreadStart [354]	This function should be called when new thread is created.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NCoreRemoveErrorSuppressed [355]	Removes a suppressed error callback from NCore [257].
	NCoreRemoveErrorSuppressedCallback [355]	Removes a suppressed error callback from NCore [257].

Macros

Name	Description
NCoreOnExit [356]	This function should be called on programs exit.
NCoreOnThreadExitEx [356]	This function should be called when program's thread finished its task.

Module

NCore Unit [352]

7.1.2.9 NCustomParameterDescriptor Unit

Provides custom parameter descriptor management functionality.

Files

Name	Description
NCustomParameterDescriptor.h [358]	Header file for the NCustomParameterDescriptor unit.

Functions

	Name	Description
	NCustomParameterDescriptorCreate [357]	Creates a new custom parameter descriptor.
	NCustomParameterDescriptorGetData [358]	Retrieves the data of the specified descriptor.

7.1.2.9.1 Functions

7.1.2.9.1.1 NCustomParameterDescriptorCreate Function

Creates a new custom parameter descriptor.

C++

```
NResult N_API NCustomParameterDescriptorCreate(HNString hName, HNType hParameterType,
NAttributes attributes, HNString hFormat, HNValue hDefaultValue, HNValue hMinValue, HNValue
hMaxValue, const struct NNameValuePair_ * arStdValues, NInt stdValueCount, HNValue hData,
HNCustomParameterDescriptor * phParameterDescriptor);
```

Parameters

Parameters	Description
HNString hName	[in] Handle to the descriptor name.
HNType hParameterType	[in] Handle to the parameter type.
NAttributes attributes	[in] Descriptor attributes.
HNString hFormat	[in] Handle to the descriptor format.
HNValue hDefaultValue	[in] Handle to the default parameter value.
HNValue hMinValue	[in] Minimum parameter value.
HNValue hMaxValue	[in] Maximum parameter value.
const struct NNameValuePair_ * arStdValues	[in] Pointer to an array of standard values.
NInt stdValueCount	[in] Number of standard values.
HNValue hData	[in] Handle to the descriptor data.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNCustomParameterDescriptor * phParameterDescriptor	[out] Pointer to a handle to the parameter descriptor.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCustomParameterDescriptor Unit [357]

7.1.2.9.1.2 NCustomParameterDescriptorGetData Function

Retrieves the data of the specified descriptor.

C++

```
NResult N_API NCustomParameterDescriptorGetData(HNCustomParameterDescriptor
hParameterDescriptor, HNValue * phData);
```

Parameters

Parameters	Description
HNCustomParameterDescriptor hParameterDescriptor	[in] Handle to the custom parameter descriptor.
HNValue * phData	[out] Pointer to a handle to the data of the descriptor.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCustomParameterDescriptor Unit [357]

7.1.2.9.2 Files

7.1.2.9.2.1 NCustomParameterDescriptor.h

Header file for the NCustomParameterDescriptor [357] unit.

Functions

	Name	Description
	NCustomParameterDescriptorCreate [357]	Creates a new custom parameter descriptor.
	NCustomParameterDescriptorGetData [358]	Retrieves the data of the specified descriptor.

Module

NCustomParameterDescriptor Unit [357]

7.1.2.10 NCustomPropertyDescriptor Unit

Provides custom property descriptor management functionality.

Files

Name	Description
NCustomPropertyDescriptor.h [359]	Header file for the NCustomPropertyDescriptor unit.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NCustomPropertyDescriptorGetData [359]	Retrieves the data of the specified descriptor.

7.1.2.10.1 Functions

7.1.2.10.1.1 NCustomPropertyDescriptorGetData Function

Retrieves the data of the specified descriptor.

C++

```
NResult N_API NCustomPropertyDescriptorGetData(HNCustomPropertyDescriptor
hPropertyDescriptor, HNValue * phData);
```

Parameters

Parameters	Description
HNCustomPropertyDescriptor hPropertyDescriptor	[in] Handle to the custom property descriptor.
HNValue * phData	[out] Pointer to a handle to the data of the descriptor.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCustomPropertyDescriptor Unit [358]

7.1.2.10.2 Files

7.1.2.10.2.1 NCustomPropertyDescriptor.h

Header file for the NCustomPropertyDescriptor [358] unit.

Functions

	Name	Description
	NCustomPropertyDescriptorGetData [359]	Retrieves the data of the specified descriptor.

Module

NCustomPropertyDescriptor Unit [358]

7.1.2.11 NCustomStream Unit

Functions

	Name	Description
	NCustomStreamCreate [360]	Creates a custom stream.

7.1.2.11.1 Functions

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.11.1.1 NCustomStreamCreate Function

Creates a custom stream.

C++

```
NResult N_API NCustomStreamCreate(NBool canRead, NBool canWrite, NBool canSeek,
NCustomStreamFlushProc pFlush, void * pFlushParam, NCustomStreamGetLengthProc pGetLength,
void * pGetLengthParam, NCustomStreamSetLengthProc pSetLength, void * pSetLengthParam,
NCustomStreamGetPositionProc pGetPosition, void * pGetPositionParam,
NCustomStreamSetPositionProc pSetPosition, void * pSetPositionParam, NCustomStreamSeekProc
pSeek, void * pSeekParam, NCustomStreamReadByteProc pReadByte, void * pReadByteParam,
NCustomStreamReadProc pRead, void * pReadParam, NCustomStreamWriteByteProc pWriteByte, void
* pWriteByteParam, NCustomStreamWriteProc pWrite, void * pWriteParam, HNCustomStream *
phStream);
```

Parameters

Parameters	Description
NBool canRead	[in] NTrue [1064] if reading is permitted; else NFalse [1041].
NBool canWrite	[in] NTrue [1064] if writing is permitted; else NFalse [1041].
NBool canSeek	[in] NTrue [1064] if seeking is permitted; else NFalse [1041].
NCustomStreamFlushProc pFlush	[in] NCustomStreamFlushProc object.
void * pFlushParam	[out] Pointer to NCustomStreamFlushProc parameters.
NCustomStreamGetLengthProc pGetLength	[in] NCustomStreamGetLengthProc object.
void * pGetLengthParam	[out] Pointer to NCustomStreamGetLengthProc parameters.
NCustomStreamSetLengthProc pSetLength	[in] NCustomStreamSetLengthProc object.
void * pSetLengthParam	[out] Pointer to NCustomStreamSetLengthProc parameters.
NCustomStreamGetPositionProc pGetPosition	[in] NCustomStreamGetPositionProc object.
void * pGetPositionParam	[out] Pointer to NCustomStreamGetPositionProc parameters.
NCustomStreamSetPositionProc pSetPosition	[in] NCustomStreamSetPositionProc object.
void * pSetPositionParam	[out] Pointer to NCustomStreamSetPositionProc parameters.
NCustomStreamSeekProc pSeek	[in] NCustomStreamSeekProc object.
void * pSeekParam	[out] Pointer to NCustomStreamSeekProc parameters.
NCustomStreamReadByteProc pReadByte	[in] NCustomStreamReadByteProc object.
void * pReadByteParam	[out] Pointer to NCustomStreamReadByteProc parameters.
NCustomStreamReadProc pRead	[in] NCustomStreamReadProc object.
void * pReadParam	[out] Pointer to NCustomStreamReadProc parameters.
NCustomStreamWriteByteProc pWriteByte	[in] NCustomStreamWriteByteProc object.
void * pWriteByteParam	[out] Pointer to NCustomStreamWriteByteProc parameters.
NCustomStreamWriteProc pWrite	[in] NCustomStreamWriteProc object.
void * pWriteParam	[out] Pointer to NCustomStreamWriteProc parameters.
HNCustomStream * phStream	[out] Pointer to a handle to the new custom stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCustomStream Unit [359]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.12 NDateTime Unit

Represents an instant in time, typically expressed as a date and time of day.

C++

```
typedef NDateTime_ NDateTime;
```

Functions

	Name	Description
	NDateTimeAddDays [362]	Adds the specified number of days to the value of the specified instance.
	NDateTimeAddHours [363]	Adds the specified number of hours to the value of the specified instance.
	NDateTimeAddMilliseconds [363]	Adds the specified number of milliseconds to the value of the specified instance.
	NDateTimeAddMinutes [363]	Adds the specified number of minutes to the value of the specified instance.
	NDateTimeAddMonths [364]	Adds the specified number of months to the value of the specified instance.
	NDateTimeAddSeconds [364]	Adds the specified number of seconds to the value of the specified instance.
	NDateTimeAddTicks [364]	Adds the specified number of ticks to the value of the specified instance.
	NDateTimeAddTimeSpan [365]	Adds the specified TimeSpan to the value of the specified instance.
	NDateTimeAddYears [365]	Adds the specified number of years to the value of the specified instance.
	NDateTimeCompare [366]	Compares two instances of DateTime and returns an indication of their relative values.
	NDateTimeCreate [366]	Initializes a new instance of the DateTime structure to the specified year, month, day, hour, minute, second and millisecond.
	NDateTimeCreateDate [366]	Initializes a new instance of the DateTime structure to the specified year, month and day.
	NDateTimeCreateFromTicks [367]	Initializes a new instance of the DateTime structure from the specified number of ticks.
	NDateTimeDaysInMonth [367]	Gets the number of days the specified month of the specified year.
	NDateTimeDecode [367]	Decodes the specified DateTime instance.
	NDateTimeGetDate [368]	Gets the date (year, month and day) represented by the specified DateTime instance.
	NDateTimeGetDay [368]	Gets the day represented by the specified DateTime instance.
	NDateTimeGetDayOfWeek [369]	Gets the day of the week represented by the specified DateTime instance.
	NDateTimeGetDayOfYear [369]	Gets the day of the year represented by the specified DateTime instance.
	NDateTimeGetHour [369]	Gets the hour represented by the specified DateTime instance.
	NDateTimeGetMillisecond [370]	Gets the millisecond represented by the specified DateTime instance.
	NDateTimeGetMinute [370]	Gets the minute represented by the specified DateTime instance.
	NDateTimeGetMonth [370]	Gets the month represented by the specified DateTime instance.
	NDateTimeGetNow [371]	Gets current time of day.
	NDateTimeGetSecond [371]	Gets the second represented by the specified DateTime instance.
	NDateTimeGetTicks [371]	Gets the number of ticks that represent the specified DateTime instance.
	NDateTimeGetTimeOfDay [372]	Gets the time of the day (hour, minute and second) represented by the specified DateTime instance.
	NDateTimeGetToday [372]	Gets today's date.
	NDateTimeGetUtcNow [372]	Gets the current Coordinated Universal Time.
	NDateTimeGetYear [373]	Gets the year represented by the specified DateTime instance.
	NDateTimeIsDateTimeValid [373]	Checks whether the specified date and time are valid.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NDateTimeIsDateValid [373]	Checks whether the specified date is valid.
	NDateTimeIsLeapYear [374]	Checks whether the specified year is a leap year (366 days).
	NDateTimeIsValid [374]	Checks whether the specified DateTime instance is valid.
	NDateTimeSubtract [374]	Subtracts a DateTime instance from another DateTime instance.
	NDateTimeSubtractTimeSpan [375]	Subtracts the specified TimeSpan from the specified DateTime.
	NDateTimeToLocalTime [375]	Converts the specified DateTime to local time.
	NDateTimeToString [375]	Gets a string representation of the specified DateTime.
	NDateTimeToUniversalTime [376]	Converts the specified DateTime to Coordinated Universal Time.

Macros

Name	Description
N_DATE_TIME_MAX [377]	Maximum DateTime value.
N_DATE_TIME_MIN [377]	Minimum DateTime value.
NDateTimeToString [377]	Picks a Unicode (W) or ASCII (A) version of the NDateTimeToString function.

Structs, Records, Enums

	Name	Description
	NDayOfWeek_ [376]	Enumerates the days of the week.
	NDayOfWeek [376]	Enumerates the days of the week.

Types

Name	Description
NDateTime_ [377]	The DateTime instance.

7.1.2.12.1 Functions

7.1.2.12.1.1 NDateTimeAddDays Function

Adds the specified number of days to the value of the specified instance.

C++

```
NResult N_API NDateTimeAddDays(NDateTime_ dateTime, NDouble value, NDateTime_ * pResult);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The original DateTime instance.
NDouble value	[in] A number of whole and fractional days. This value can be negative or positive.
NDateTime_ * pResult	[out] Pointer to the modified DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.12.1.2 NDateTimeAddHours Function

Adds the specified number of hours to the value of the specified instance.

C++

```
NResult N_API NDateTimeAddHours(NDateTime_ dateTime, NDouble value, NDateTime_ * pResult);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The original DateTime instance.
NDouble value	[in] A number of whole and fractional hours. This value can be negative or positive.
NDateTime_ * pResult	[out] Pointer to the modified DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.3 NDateTimeAddMilliseconds Function

Adds the specified number of milliseconds to the value of the specified instance.

C++

```
NResult N_API NDateTimeAddMilliseconds(NDateTime_ dateTime, NDouble value, NDateTime_ *
pResult);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The original DateTime instance.
NDouble value	[in] A number of whole and fractional milliseconds. This value can be negative or positive.
NDateTime_ * pResult	[out] Pointer to the modified DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.4 NDateTimeAddMinutes Function

Adds the specified number of minutes to the value of the specified instance.

C++

```
NResult N_API NDateTimeAddMinutes(NDateTime_ dateTime, NDouble value, NDateTime_ * pResult);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The original DateTime instance.
NDouble value	[in] A number of whole and fractional minutes. This value can be negative or positive.
NDateTime_ * pResult	[out] Pointer to the modified DateTime instance.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.5 NDateTimeAddMonths Function

Adds the specified number of months to the value of the specified instance.

C++

```
NResult N_API NDateTimeAddMonths(NDateTime_ dateTime, NInt value, NDateTime_ * pResult);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The original DateTime instance.
NInt value	[in] A number of whole and fractional months. This value can be negative or positive.
NDateTime_ * pResult	[out] Pointer to the modified DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.6 NDateTimeAddSeconds Function

Adds the specified number of seconds to the value of the specified instance.

C++

```
NResult N_API NDateTimeAddSeconds(NDateTime_ dateTime, NDouble value, NDateTime_ * pResult);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The original DateTime instance.
NDouble value	[in] A number of whole and fractional seconds. This value can be negative or positive.
NDateTime_ * pResult	[out] Pointer to the modified DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.7 NDateTimeAddTicks Function

Adds the specified number of ticks to the value of the specified instance.

C++

```
NResult N_API NDateTimeAddTicks(NDateTime_ dateTime, NLong value, NDateTime_ * pResult);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The original DateTime instance.
NLong value	[in] A number of 100-nanosecond ticks. This value can be negative or positive.
NDateTime_ * pResult	[out] Pointer to the modified DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.8 NDateTimeAddTimeSpan Function

Adds the specified TimeSpan to the value of the specified instance.

C++

```
NResult N_API NDateTimeAddTimeSpan(NDateTime_ dateTime, NTimeSpan_ timeSpan, NDateTime_ *
pResult);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The original DateTime instance.
NTimeSpan_ timeSpan	[in] The TimeSpan that contains the interval to add.
NDateTime_ * pResult	[out] Pointer to the modified DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.9 NDateTimeAddYears Function

Adds the specified number of years to the value of the specified instance.

C++

```
NResult N_API NDateTimeAddYears(NDateTime_ dateTime, NInt value, NDateTime_ * pResult);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The original DateTime instance.
NInt value	[in] A number of whole and fractional years. This value can be negative or positive.
NDateTime_ * pResult	[out] Pointer to the modified DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.12.1.10 NDateTimeCompare Function

Compares two instances of DateTime and returns an indication of their relative values.

C++

```
NResult N_API NDateTimeCompare(NDateTime_ value1, NDateTime_ value2, NInt * pResult);
```

Parameters

Parameters	Description
NDateTime_ value1	[in] First DateTime instance.
NDateTime_ value2	[in] Second DateTime instance.
NInt * pResult	[out] Pointer to an integer indicating the relative values of the instances.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.11 NDateTimeCreate Function

Initializes a new instance of the DateTime structure to the specified year, month, day, hour, minute, second and millisecond.

C++

```
NResult N_API NDateTimeCreate(NInt year, NInt month, NInt day, NInt hour, NInt minute, NInt
second, NInt millisecond, NDateTime_ * pDateTime);
```

Parameters

Parameters	Description
NInt year	[in] The year value.
NInt month	[in] The month value.
NInt day	[in] The day value.
NInt hour	[in] The hour value.
NInt minute	[in] The minute value.
NInt second	[in] The second value.
NInt millisecond	[in] The millisecond value.
NDateTime_ * pDateTime	[out] Pointer to the new DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.12 NDateTimeCreateDate Function

Initializes a new instance of the DateTime structure to the specified year, month and day.

C++

```
NResult N_API NDateTimeCreateDate(NInt year, NInt month, NInt day, NDateTime_ * pDateTime);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NInt year	[in] Years value.
NInt month	[in] Months value.
NInt day	[in] Days value.
NDateTime_ * pDateTime	[out] Pointer to the new DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.13 NDateTimeCreateFromTicks Function

Initializes a new instance of the DateTime structure from the specified number of ticks.

C++

```
NResult N_API NDateTimeCreateFromTicks(NLong ticks, NDateTime_ * pDateTime);
```

Parameters

Parameters	Description
NLong ticks	[in] Number of 100-nanosecond ticks.
NDateTime_ * pDateTime	[out] Pointer to the new DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.14 NDateTimeDaysInMonth Function

Gets the number of days the specified month of the specified year.

C++

```
NResult N_API NDateTimeDaysInMonth(NInt year, NInt month, NInt * pValue);
```

Parameters

Parameters	Description
NInt year	[in] The year.
NInt month	[in] The month.
NInt * pValue	[out] Pointer to the number of days.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.15 NDateTimeDecode Function

Decodes the specified DateTime instance.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NDateTimeDecode(NDateTime_ dateTime, NInt * pYear, NInt * pDayOfYear, NInt *
pMonth, NInt * pDay, NDayOfWeek * pDayOfWeek, NInt * pHour, NInt * pMinute, NInt * pSecond,
NInt * pMillisecond);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance to decode.
NInt * pYear	[out] Pointer to the years value.
NInt * pDayOfYear	[out] Pointer to the day of the year value.
NInt * pMonth	[out] Pointer to the months value.
NInt * pDay	[out] Pointer to the days value.
NDayOfWeek * pDayOfWeek	[out] Pointer to the day of week value.
NInt * pHour	[out] Pointer to the hours value.
NInt * pMinute	[out] Pointer to the minutes value.
NInt * pSecond	[out] Pointer to the seconds value.
NInt * pMillisecond	[out] Pointer to the milliseconds value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.16 NDateTimeGetDate Function

Gets the date (year, month and day) represented by the specified DateTime instance.

C++

```
NResult N_API NDateTimeGetDate(NDateTime_ dateTime, NDateTime_ * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance.
NDateTime_ * pValue	[out] Pointer to DateTime instance containing the date.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.17 NDateTimeGetDay Function

Gets the day represented by the specified DateTime instance.

C++

```
NResult N_API NDateTimeGetDay(NDateTime_ dateTime, NInt * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance.
NInt * pValue	[out] Pointer to the day value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.18 NDateTimeGetDayOfWeek Function

Gets the day of the week represented by the specified DateTime instance.

C++

```
NResult N_API NDateTimeGetDayOfWeek(NDateTime_ dateTime, NDayOfWeek * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance.
NDayOfWeek * pValue	[out] Pointer to the day of the week value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.19 NDateTimeGetDayOfYear Function

Gets the day of the year represented by the specified DateTime instance.

C++

```
NResult N_API NDateTimeGetDayOfYear(NDateTime_ dateTime, NInt * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance.
NInt * pValue	[out] Pointer to the day of the year value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.20 NDateTimeGetHour Function

Gets the hour represented by the specified DateTime instance.

C++

```
NResult N_API NDateTimeGetHour(NDateTime_ dateTime, NInt * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance.
NInt * pValue	[out] Pointer to the hour value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.21 NDateTimeGetMillisecond Function

Gets the millisecond represented by the specified DateTime instance.

C++

```
NResult N_API NDateTimeGetMillisecond(NDateTime_ dateTime, NInt * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance.
NInt * pValue	[out] Pointer to the millisecond value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.22 NDateTimeGetMinute Function

Gets the minute represented by the specified DateTime instance.

C++

```
NResult N_API NDateTimeGetMinute(NDateTime_ dateTime, NInt * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance.
NInt * pValue	[out] Pointer to the minute value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.23 NDateTimeGetMonth Function

Gets the month represented by the specified DateTime instance.

C++

```
NResult N_API NDateTimeGetMonth(NDateTime_ dateTime, NInt * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance.
NInt * pValue	[out] Pointer to the month value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.24 NDateTimeGetNow Function

Gets current time of day.

C++

```
NResult N_API NDateTimeGetNow(NDateTime_ * pValue);
```

Parameters

Parameters	Description
NDateTime_ * pValue	[out] Pointer to DateTime containing current time of day.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.25 NDateTimeGetSecond Function

Gets the second represented by the specified DateTime instance.

C++

```
NResult N_API NDateTimeGetSecond(NDateTime_ dateTime, NInt * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance.
NInt * pValue	[out] Pointer to the second value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.26 NDateTimeGetTicks Function

Gets the number of ticks that represent the specified DateTime instance.

C++

```
NResult N_API NDateTimeGetTicks(NDateTime_ dateTime, NLong * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance.
NLong * pValue	[out] Pointer to the number of ticks.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Type

NDateTime Unit [361]

7.1.2.12.1.27 NDateTimeGetTimeOfDay Function

Gets the time of the day (hour, minute and second) represented by the specified DateTime instance.

C++

```
NResult N_API NDateTimeGetTimeOfDay(NDateTime_ dateTime, NTimeSpan_ * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance.
NTimeSpan_ * pValue	[out] Pointer to DateTime containing the time of day.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.28 NDateTimeGetToday Function

Gets today's date.

C++

```
NResult N_API NDateTimeGetToday(NDateTime_ * pValue);
```

Parameters

Parameters	Description
NDateTime_ * pValue	[out] Pointer to DateTime containing today's date.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.29 NDateTimeGetUtcNow Function

Gets the current Coordinated Universal Time.

C++

```
NResult N_API NDateTimeGetUtcNow(NDateTime_ * pValue);
```

Parameters

Parameters	Description
NDateTime_ * pValue	[out] Pointer to DateTime instance containing the current Coordinated Universal Time.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.12.1.30 NDateTimeGetYear Function

Gets the year represented by the specified DateTime instance.

C++

```
NResult N_API NDateTimeGetYear(NDateTime_ dateTime, NInt * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance.
NInt * pValue	[out] Pointer to the year value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.31 NDateTimeIsDateTimeValid Function

Checks whether the specified date and time are valid.

C++

```
NBool N_API NDateTimeIsDateTimeValid(NInt year, NInt month, NInt day, NInt hour, NInt
minute, NInt second, NInt millisecond);
```

Parameters

Parameters	Description
NInt year	[in] Year value.
NInt month	[in] Month value.
NInt day	[in] Day value.
NInt hour	[in] Hour value.
NInt minute	[in] Minute value.
NInt second	[in] Second value.
NInt millisecond	[in] Millisecond value.

Returns

NTrue [1064] if the date and time are valid; else NFalse [1041].

Type

NDateTime Unit [361]

7.1.2.12.1.32 NDateTimeIsDateValid Function

Checks whether the specified date is valid.

C++

```
NBool N_API NDateTimeIsDateValid(NInt year, NInt month, NInt day);
```

Parameters

Parameters	Description
NInt year	[in] Year value.
NInt month	[in] Month value.
NInt day	[in] Day value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NTrue [1064] if the date is valid; else NFalse [1041].

Type

NDateTime Unit [361]

7.1.2.12.1.33 NDateTimeIsLeapYear Function

Checks whether the specified year is a leap year (366 days).

C++

```
NResult N_API NDateTimeIsLeapYear(NInt year, NBool * pValue);
```

Parameters

Parameters	Description
NInt year	[in] The year to be checked.
NBool * pValue	[out] Pointer to NBool [1004] indicating whether the year is a leap year.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.34 NDateTimeIsValid Function

Checks whether the specified DateTime instance is valid.

C++

```
NBool N_API NDateTimeIsValid(NDateTime_ value);
```

Parameters

Parameters	Description
NDateTime_ value	[in] The DateTime instance.

Returns

NTrue [1064] if the DateTime is valid; else NFalse [1041].

Type

NDateTime Unit [361]

7.1.2.12.1.35 NDateTimeSubtract Function

Subtracts a DateTime instance from another DateTime instance.

C++

```
NResult N_API NDateTimeSubtract(NDateTime_ value1, NDateTime_ value2, NTimeSpan_ *
pTimeSpan);
```

Parameters

Parameters	Description
NDateTime_ value1	[in] The first DateTime instance.
NDateTime_ value2	[in] The second DateTime instance.
NTimeSpan_ * pTimeSpan	[out] Pointer to the resulting DateTime instance.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.36 NDateTimeSubtractTimeSpan Function

Subtracts the specified TimeSpan from the specified DateTime.

C++

```
NResult N_API NDateTimeSubtractTimeSpan(NDateTime_ dateTime, NTimeSpan_ timeSpan,
NDateTime_ * pResult);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The original DateTime instance.
NTimeSpan_ timeSpan	[in] The TimeSpan to subtract.
NDateTime_ * pResult	[out] Pointer to the resulting DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.37 NDateTimeToLocalTime Function

Converts the specified DateTime to local time.

C++

```
NResult N_API NDateTimeToLocalTime(NDateTime_ dateTime, NDateTime_ * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance to be coverted.
NDateTime_ * pValue	[out] Pointer to the converted DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.38 NDateTimeToString Function

Gets a string representation of the specified DateTime.

C++

```
NResult N_API NDateTimeToString(NDateTime value, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
NDateTime value	[in] DateTime object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const NChar * szFormat	[in] Pointer to the format.
HNString * phValue	[out] Pointer to a handle to a string representing the DateTime object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.1.39 NDateTimeToUniversalTime Function

Converts the specified DateTime to Coordinated Universal Time.

C++

```
NResult N_API NDateTimeToUniversalTime(NDateTime_ dateTime, NDateTime_ * pValue);
```

Parameters

Parameters	Description
NDateTime_ dateTime	[in] The DateTime instance to be coverted.
NDateTime_ * pValue	[out] Pointer to the converted DateTime instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NDateTime Unit [361]

7.1.2.12.2 Structs, Records, Enums

7.1.2.12.2.1 NDayOfWeek Enumeration

Enumerates the days of the week.

C++

```
typedef enum NDayOfWeek_ {
  ndowSunday = 0,
  ndowMonday = 1,
  ndowTuesday = 2,
  ndowWednesday = 3,
  ndowThursday = 4,
  ndowFriday = 5,
  ndowSaturday = 6
} NDayOfWeek;
```

Members

Members	Description
ndowSunday = 0	Sunday.
ndowMonday = 1	Monday.
ndowTuesday = 2	Tuesday.
ndowWednesday = 3	Wednesday.
ndowThursday = 4	Thursday.
ndowFriday = 5	Friday.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

ndowSaturday = 6	Saturday.

Type

NDateTime Unit [361]

7.1.2.12.3 Types

7.1.2.12.3.1 NDateTime_ Type

The DateTime instance.

C++

```
typedef NLong NDateTime_;
```

Type

NDateTime Unit [361]

7.1.2.12.4 Macros

7.1.2.12.4.1 N_DATE_TIME_MAX Macro

Maximum DateTime value.

C++

```
#define N_DATE_TIME_MAX 3155378975999999999LL
```

Type

NDateTime Unit [361]

7.1.2.12.4.2 N_DATE_TIME_MIN Macro

Minimum DateTime value.

C++

```
#define N_DATE_TIME_MIN 0LL
```

Type

NDateTime Unit [361]

7.1.2.12.4.3 NDateTimeToString Macro

Picks a Unicode (W) or ASCII (A) version of the NDateTimeToString function.

C++

```
#define NDateTimeToString N_FUNC_AW(NDateTimeToString)
```

Type

NDateTime Unit [361]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.13 NDefs Unit

Macros

Name	Description
N_64 [379]	Defined if compiling for 64-bit architecture.
N_ALIGN [379]	Controls the alignment of user-defined data
N_ANSI_C [379]	Defined if ANSI C language compliance is enabled in compiler.
N_API [379]	Defines calling conventions for functions. Tells the compiler that the declared function and the functions it calls never throw an exception.
N_API_PTR_RET [379]	Defines calling conventions for functions. Tells the compiler that the declared function and the functions it calls never throw an exception. Used for functions that return a pointer.
N_BIG_ENDIAN [380]	Defined if compiling for big-endian processor architecture.
N_CALL_CONV [380]	Defines calling conventions for functions.
N_CALLBACK [380]	Callback.
N_CPP [380]	Defined if compiling as C++ code.
N_DEBUG [380]	Defined if compiling in debug mode.
N_DEPRECATED [380]	Indicates that a function, type, or any other identifier may no longer be supported in a future release or should no longer be used.
N_FAST_FLOAT [381]	Defined if CPU has floating-point instructions.
N_GCC [381]	Defined if compiling with GCC.
N_LIB [381]	Defined if compiling static library.
N_LIKELY [381]	Used for Linux branching optimization.
N_LINUX [381]	Defined if compiling for Linux.
N_MAC [381]	Defined if compiling for Mac OS.
N_MSVC [382]	Defined if compiling with Microsoft Visual C++.
N_NO_ANSI_FUNC [382]	Defined if compiling for platform without ANSI versions of the functions support.
N_NO_INLINE [382]	Tells the compiler to never inline a particular member function (function in a class).
N_NO_INT_64 [382]	Defined if compiling for platform without 64-bit integer types support.
N_NO_RETURN [382]	Tells the compiler that a function does not return.
N_NO_THROW [383]	Tells the compiler that the declared function and the functions it calls never throw an exception.
N_NO_UNICODE [383]	Defined if compiling for platform without Unicode versions of the functions support.
N_PACKED [383]	Struct packing/alignment.
N_RETURN_ADDRESS [383]	The memory address a return statement hands control of a computer program to.
N_STATIC_ASSERT [383]	Evaluate an expression at compile time and generate an error when the result is NFalse [1041].
N_TRACE [383]	Defined if trace output is enabled during compilation.
N_UNICODE [384]	Defined if compiling with Unicode character set (affects NChar [1005] type).
N_UNLIKELY [384]	Used for Linux branching optimization.
N_WINDOWS [384]	Defined if compiling for Windows.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.13.1 Macros

7.1.2.13.1.1 N_64 Macro

Defined if compiling for 64-bit architecture.

C++

```
#define N_64
```

Module

NDefs Unit [378]

7.1.2.13.1.2 N_ALIGN Macro

Controls the alignment of user-defined data

C++

```
#define N_ALIGN(x)
```

Module

NDefs Unit [378]

7.1.2.13.1.3 N_ANSI_C Macro

Defined if ANSI C language compliance is enabled in compiler.

C++

```
#define N_ANSI_C
```

Module

NDefs Unit [378]

7.1.2.13.1.4 N_API Macro

Defines calling conventions for functions. Tells the compiler that the declared function and the functions it calls never throw an exception.

C++

```
#define N_API N_NO_THROW N_CALL_CONV
```

Module

NDefs Unit [378]

7.1.2.13.1.5 N_API_PTR_RET Macro

Defines calling conventions for functions. Tells the compiler that the declared function and the functions it calls never throw an exception.

Used for functions that return a pointer.

C++

```
#define N_API_PTR_RET N_NO_THROW N_CALL_CONV
```

Module

NDefs Unit [378]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.13.1.6 N_BIG_ENDIAN Macro

Defined if compiling for big-endian processor architecture.

C++

```
#define N_BIG_ENDIAN
```

Module

NDefs Unit [378]

7.1.2.13.1.7 N_CALL_CONV Macro

Defines calling conventions for functions.

C++

```
#define N_CALL_CONV
```

Module

NDefs Unit [378]

7.1.2.13.1.8 N_CALLBACK Macro

Callback.

C++

```
#define N_CALLBACK N_CALL_CONV *
```

Module

NDefs Unit [378]

7.1.2.13.1.9 N_CPP Macro

Defined if compiling as C++ code.

C++

```
#define N_CPP
```

Module

NDefs Unit [378]

7.1.2.13.1.10 N_DEBUG Macro

Defined if compiling in debug mode.

C++

```
#define N_DEBUG
```

Module

NDefs Unit [378]

7.1.2.13.1.11 N_DEPRECATED Macro

Indicates that a function, type, or any other identifier may no longer be supported in a future release or should no longer be used.

C++

```
#define N_DEPRECATED(message)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NDefs Unit [378]

7.1.2.13.1.12 N_FAST_FLOAT Macro

Defined if CPU has floating-point instructions.

C++

```
#define N_FAST_FLOAT
```

Module

NDefs Unit [378]

7.1.2.13.1.13 N_GCC Macro

Defined if compiling with GCC.

C++

```
#define N_GCC
```

Module

NDefs Unit [378]

7.1.2.13.1.14 N_LIB Macro

Defined if compiling static library.

C++

```
#define N_LIB
```

Module

NDefs Unit [378]

7.1.2.13.1.15 N_LIKELY Macro

Used for Linux branching optimization.

C++

```
#define N_LIKELY(x) (x)
```

Module

NDefs Unit [378]

7.1.2.13.1.16 N_LINUX Macro

Defined if compiling for Linux.

C++

```
#define N_LINUX
```

Module

NDefs Unit [378]

7.1.2.13.1.17 N_MAC Macro

Defined if compiling for Mac OS.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_MAC
```

Module

NDefs Unit [378]

7.1.2.13.1.18 N_MSVC Macro

Defined if compiling with Microsoft Visual C++.

C++

```
#define N_MSVC
```

Module

NDefs Unit [378]

7.1.2.13.1.19 N_NO_ANSI_FUNC Macro

Defined if compiling for platform without ANSI versions of the functions support.

C++

```
#define N_NO_ANSI_FUNC
```

Module

NDefs Unit [378]

7.1.2.13.1.20 N_NO_INLINE Macro

Tells the compiler to never inline a particular member function (function in a class).

C++

```
#define N_NO_INLINE
```

Module

NDefs Unit [378]

7.1.2.13.1.21 N_NO_INT_64 Macro

Defined if compiling for platform without 64-bit integer types support.

C++

```
#define N_NO_INT_64
```

Module

NDefs Unit [378]

7.1.2.13.1.22 N_NO_RETURN Macro

Tells the compiler that a function does not return.

C++

```
#define N_NO_RETURN
```

Module

NDefs Unit [378]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.13.1.23 N_NO_THROW Macro

Tells the compiler that the declared function and the functions it calls never throw an exception.

C++

```
#define N_NO_THROW
```

Module

NDefs Unit [378]

7.1.2.13.1.24 N_NO_UNICODE Macro

Defined if compiling for platform without Unicode versions of the functions support.

C++

```
#define N_NO_UNICODE
```

Module

NDefs Unit [378]

7.1.2.13.1.25 N_PACKED Macro

Struct packing/alignment.

C++

```
#define N_PACKED
```

Module

NDefs Unit [378]

7.1.2.13.1.26 N_RETURN_ADDRESS Macro

The memory address a return statement hands control of a computer program to.

C++

```
#define N_RETURN_ADDRESS NULL
```

Module

NDefs Unit [378]

7.1.2.13.1.27 N_STATIC_ASSERT Macro

Evaluate an expression at compile time and generate an error when the result is NFalse [1041].

C++

```
#define N_STATIC_ASSERT(x) ((void)sizeof(char[1 - 2 * !(x)]))
```

Module

NDefs Unit [378]

7.1.2.13.1.28 N_TRACE Macro

Defined if trace output is enabled during compilation.

C++

```
#define N_TRACE
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NDefs Unit [378]

7.1.2.13.1.29 N_UNICODE Macro

Defined if compiling with Unicode character set (affects NChar [1005] type).

C++

```
#define N_UNICODE
```

Module

NDefs Unit [378]

7.1.2.13.1.30 N_UNLIKELY Macro

Used for Linux branching optimization.

C++

```
#define N_UNLIKELY(x) (x)
```

Module

NDefs Unit [378]

7.1.2.13.1.31 N_WINDOWS Macro

Defined if compiling for Windows.

C++

```
#define N_WINDOWS
```

Module

NDefs Unit [378]

7.1.2.14 NDescriptor Unit

Provides descriptor management functionality.

Files

Name	Description
NDescriptor.h [385]	Header file for the NDescriptor unit.

Functions

	Name	Description
	NDescriptorGetAttributes [384]	Retrieves the attributes of the specified descriptor.
	NDescriptorGetName [385]	Retrieves the name of the specified descriptor.

7.1.2.14.1 Functions

7.1.2.14.1.1 NDescriptorGetAttributes Function

Retrieves the attributes of the specified descriptor.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NDescriptorGetAttributes(HNDescriptor hDescriptor, NAttributes * pValue);
```

Parameters

Parameters	Description
HNDescriptor hDescriptor	[in] Handle to the descriptor.
NAttributes * pValue	[out] Pointer to the attributes of the descriptor.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NDescriptor Unit [384]

7.1.2.14.1.2 NDescriptorGetName Function

Retrieves the name of the specified descriptor.

C++

```
NResult N_API NDescriptorGetName(HNDescriptor hDescriptor, HNString * phValue);
```

Parameters

Parameters	Description
HNDescriptor hDescriptor	[in] Handle to the descriptor.
HNString * phValue	[out] Pointer to the name of the descriptor.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NDescriptor Unit [384]

7.1.2.14.2 Files

7.1.2.14.2.1 NDescriptor.h

Header file for the NDescriptor [384] unit.

Functions

	Name	Description
	NDescriptorGetAttributes [384]	Retrieves the attributes of the specified descriptor.
	NDescriptorGetName [385]	Retrieves the name of the specified descriptor.

Module

NDescriptor Unit [384]

7.1.2.15 NDirectory Unit

Functions

	Name	Description
	NDirectoryExists [386]	Checks whether the specified directory exists.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Macros

Name	Description
NDirectoryExists [386]	Picks a Unicode (W) or ASCII (A) version of the NDirectoryExists function.

7.1.2.15.1 Functions

7.1.2.15.1.1 NDirectoryExists Function

Checks whether the specified directory exists.

C++

```
NResult N_API NDirectoryExists(const NChar * szPath, NBool * pValue);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the directory path.
NBool * pValue	[out] NTrue [1064] if the directory exists; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NDirectory Unit [385]

7.1.2.15.2 Macros

7.1.2.15.2.1 NDirectoryExists Macro

Picks a Unicode (W) or ASCII (A) version of the NDirectoryExists function.

C++

```
#define NDirectoryExists N_FUNC_AW(NDirectoryExists)
```

Module

NDirectory Unit [385]

7.1.2.16 NEnum Unit

Provides base functionality for enumerations.

Functions

	Name	Description
	NEnumParseN [387]	Converts the string representation of the name or numeric value of enumerated constant to an equivalent NInt [1006].
	NEnumParseP [387]	Converts the string representation of the name or numeric value of enumerated constant to an equivalent NInt [1006].
	NEnumToStringN [387]	Converts NEnum value specified as NInt [1006] to string.
	NEnumToStringP [388]	Converts NEnum value specified as NInt [1006] to string.
	NEnumTryParseN [388]	Converts the string representation of the name of enumerated constant to an equivalent NInt [1006].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NEnumTryParseP [389]	Converts the string representation of the name of enumerated constant to an equivalent NInt [1006].

7.1.2.16.1 Functions

7.1.2.16.1.1 NEnumParseN Function

Converts the string representation of the name or numeric value of enumerated constant to an equivalent NInt [1006].

C++

```
NResult N_API NEnumParseN(HNType hType, HNString hValue, HNString hFormat, NInt * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Enumeration type.
HNString hValue	[in] Handle to NString [678] containing the name or value to convert.
HNString hFormat	[in] Handle to NString [678] containing format specifier.
NInt * pValue	[out] Pointer to NInt [1006] containing integer representation of *hValue*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NEnum Unit [386]

7.1.2.16.1.2 NEnumParseP Function

Converts the string representation of the name or numeric value of enumerated constant to an equivalent NInt [1006].

C++

```
NResult N_API NEnumParseP(NTypeOfProc pTypeOf, const NChar * szValue, const NChar *
szFormat, NInt * pValue);
```

Parameters

Parameters	Description
NTypeOfProc pTypeOf	[in] Enumeration type.
const NChar * szValue	[in] Zero-terminated string containing the name or value to convert.
const NChar * szFormat	[in] Zero-terminated string containing format specifier.
NInt * pValue	[out] Pointer to NInt [1006] containing integer representation of *hValue*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NEnum Unit [386]

7.1.2.16.1.3 NEnumToStringN Function

Converts NEnum [386] value specified as NInt [1006] to string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NEnumToStringN(HNType hType, NInt value, HNString hFormat, HNString *
phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to NEnum [386] type value.
NInt value	[in] NEnum [386] value specified as NInt [1006].
HNString hFormat	[in] Handle to NString [678] containing format specifier.
HNString * phValue	[out] Pointer to NString [678] containing string representation of the specified *value*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NEnum Unit [386]

7.1.2.16.1.4 NEnumToStringP Function

Converts NEnum [386] value specified as NInt [1006] to string.

C++

```
NResult N_API NEnumToStringP(NTypeOfProc pTypeOf, NInt value, const NChar * szFormat,
HNString * phValue);
```

Parameters

Parameters	Description
NTypeOfProc pTypeOf	[in] Pointer to NEnum [386] type value.
NInt value	[in] NEnum [386] value specified as NInt [1006] to convert.
const NChar * szFormat	[in] Zero-terminated string containing format specifier.
HNString * phValue	[out] Pointer to NString [678] containing string representation of the specified *value*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NEnum Unit [386]

7.1.2.16.1.5 NEnumTryParseN Function

Converts the string representation of the name of enumerated constant to an equivalent NInt [1006].

C++

```
NResult N_API NEnumTryParseN(HNType hType, HNString hValue, HNString hFormat, NInt *
pValue, NBool * pResult);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to NEnum [386] type value.
HNString hValue	[in] Handle to NString [678] containing the name or value to convert.
HNString hFormat	[in] Handle to NString [678] containing format specifier.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt * pValue	[out] Pointer to NInt [1006] containing integer representation of *hValue*.
NBool * pResult	[out] NTrue [1064] if the *value* was converted successfully; otherwise NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NEnum Unit [386]

7.1.2.16.1.6 NEnumTryParseP Function

Converts the string representation of the name of enumerated constant to an equivalent NInt [1006].

C++

```
NResult N_API NEnumTryParseP(NTypeOfProc pTypeOf, const NChar * szValue, const NChar *
szFormat, NInt * pValue, NBool * pResult);
```

Parameters

Parameters	Description
NTypeOfProc pTypeOf	[in] NEnum [386] type value.
const NChar * szValue	[in] Zero-terminated string containing the name or value to convert.
const NChar * szFormat	[in] Zero-terminated string containing format specifier.
NInt * pValue	[out] Pointer to NInt [1006] containing integer representation of *hValue*.
NBool * pResult	[out] NTrue [1064] if the *value* was converted successfully; otherwise NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NEnum Unit [386]

7.1.2.17 NEncoding Unit

Represents a character encoding.

C++

```
typedef enum NEncoding_ {
  neDefault = 0,
  neUtf8 = 1,
  neAscii = 2,
  neUtf7 = 3,
  neUtf16BE = 4,
  neUtf16LE = 5,
  neUtf32BE = 6,
  neUtf32LE = 7
} NEncoding;
```

Members

Members	Description
neDefault = 0	Specified default encoding.
neUtf8 = 1	Specified UTF-8 encoding.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

neAscii = 2	Specified ASCII encoding.
neUtf7 = 3	Specified UTF-7 encoding.
neUtf16BE = 4	Specified UTF-16 BE encoding.
neUtf16LE = 5	Specified UTF-16 LE encoding.
neUtf32BE = 6	Specified UTF-32 BE encoding.
neUtf32LE = 7	Specified UTF-32 LE encoding.

Functions

	Name	Description
	NEncodingConvert [391]	Converts a range of bytes in a byte array from one encoding to another and writes the converted data to a new buffer.
	NEncodingDetect [391]	Detects the encoding of the specified data.
	NEncodingGetBytes [392]	Encodes a character into the specified byte array.
	NEncodingGetBytesForChars [392]	Encodes a set of characters from the specified character array into the specified byte array.
	NEncodingGetChars [393]	Decodes all the bytes in the specified byte array into a set of characters.
	NEncodingGetMaxByteCount [393]	Calculates the maximum number of bytes produced by encoding the specified number of characters.
	NEncodingGetMaxCharCount [393]	Calculates the maximum number of characters produced by decoding the specified number of bytes.
	NEncodingGetMinBytesPerChar [394]	Gets the minimum number of bytes used to encode one character.
	NEncodingGetPreamble [394]	Returns a sequence of bytes that specifies the encoding used.
	NEncodingGetPreambleLength [395]	Gets the length of the preamble of the specified encoding.
	NEncodingGetStringN [395]	Decodes all the bytes in the specified byte array into a string.
	NEncodingIsSingleByte [395]	Gets a value indicating whether the specified encoding uses single-byte code points.
	NEncodingIsValid [396]	Checks whether the specified encoding is valid.

Macros

Name	Description
N_ACHAR_ENCODING [397]	Specifies AChar encoding.
N_CHAR_ENCODING [397]	Specifies Char encoding.
N_WCHAR_ENCODING [397]	Specifies WChar encoding.
NEncodingGetBytes [397]	Picks a Unicode (W) or ASCII (A) version of the NEncodingGetBytes function.
NEncodingGetBytesForChars [397]	Picks a Unicode (W) or ASCII (A) version of the NEncodingGetBytesForChars function.
NEncodingGetChars [398]	Picks a Unicode (W) or ASCII (A) version of the NEncodingGetChars function.
NEncodingGetMaxByteCount [398]	Picks a Unicode (W) or ASCII (A) version of the NEncodingGetMaxByteCount function.
NEncodingGetMaxCharCount [398]	Picks a Unicode (W) or ASCII (A) version of the NEncodingGetMaxCharCount function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Structs, Records, Enums

	Name	Description
	NEncoding_ [396]	Enumerates encoding types.

7.1.2.17.1 Functions

7.1.2.17.1.1 NEncodingConvert Function

Converts a range of bytes in a byte array from one encoding to another and writes the converted data to a new buffer.

C++

```
NResult N_API NEncodingConvert(NEncoding srcEncoding, const void * pSrcData, NSizeType
srcDataSize, NBool srcIsBlock, NSizeType * pSrcDataUsed, NEncoding dstEncoding, void *
pDstData, NSizeType dstDataSize, NBool dstIsBlock, NSizeType * pDstDataUsed, NBool *
pCompleted);
```

Parameters

Parameters	Description
NEncoding srcEncoding	[in] The original data encoding.
const void * pSrcData	[in] Pointer to the buffer containing the original data.
NSizeType srcDataSize	[in] Size of the original data buffer.
NBool srcIsBlock	[in] NTrue [1064] if the original buffer is a block-level object; else NFalse [1041].
NSizeType * pSrcDataUsed	[out] Pointer to the size of the data to be converted.
NEncoding dstEncoding	[in] The new data encoding.
void * pDstData	[out] Pointer to the buffer containing the converted data.
NSizeType dstDataSize	[out] Pointer to the size of the original data buffer.
NBool dstIsBlock	[in] NTrue [1064] if the destination buffer is a block-level object; else NFalse [1041].
NSizeType * pDstDataUsed	[out] Pointer to the size of the converted data.
NBool * pCompleted	[out] NTrue [1064] if the conversion was successful; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Enumeration

NEncoding Unit [389]

7.1.2.17.1.2 NEncodingDetect Function

Detects the encoding of the specified data.

C++

```
NResult N_API NEncodingDetect(const void * pData, NSizeType dataSize, NSizeType *
pPreambleLength, NEncoding * pEncoding);
```

Parameters

Parameters	Description
const void * pData	[in] Pointer to the data to detect the encoding of.
NSizeType dataSize	[in] Size of the data.
NSizeType * pPreambleLength	[out] Pointer to the length of the preamble.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NEncoding * pEncoding	[out] Pointer to the data encoding.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Enumeration

NEncoding Unit [389]

7.1.2.17.1.3 NEncodingGetBytes Function

Encodes a character into the specified byte array.

C++

```
NResult N_API NEncodingGetBytes(NEncoding encoding, const NChar * szValue, void * pData,
NSizeType dataSize, NSizeType * pDataUsed);
```

Parameters

Parameters	Description
NEncoding encoding	[in] The encoding to be used.
const NChar * szValue	[in] Pointer to the char to be converted.
void * pData	[out] Pointer to the buffer to contain the converted bytes.
NSizeType dataSize	[in] Size of the buffer to contain the converted bytes.
NSizeType * pDataUsed	[out] Pointer to the number of bytes used.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Enumeration

NEncoding Unit [389]

7.1.2.17.1.4 NEncodingGetBytesForChars Function

Encodes a set of characters from the specified character array into the specified byte array.

C++

```
NResult N_API NEncodingGetBytesForChars(NEncoding encoding, const NChar * arChars, NInt
charsLength, NBool charsAreBlock, NInt * pCharsUsed, void * pData, NSizeType dataSize,
NSizeType * pDataUsed);
```

Parameters

Parameters	Description
NEncoding encoding	[in] The encoding to be used.
const NChar * arChars	[in] Pointer to a char array to be encoded.
NInt charsLength	[in] Length of the char array.
NBool charsAreBlock	[in] NTrue [1064] if the char array is a block-level object; else NFalse [1041].
NInt * pCharsUsed	[out] Pointer to the number of chars used.
void * pData	[out] Pointer to the buffer to contain the converted bytes.
NSizeType dataSize	[in] Size of the buffer to contain the converted bytes.
NSizeType * pDataUsed	[out] Pointer to the number of bytes used.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Enumeration

NEncoding Unit [389]

7.1.2.17.1.5 NEncodingGetChars Function

Decodes all the bytes in the specified byte array into a set of characters.

C++

```
NResult N_API NEncodingGetChars(NEncoding encoding, const void * pData, NSizeType dataSize,
NBool dataIsBlock, NSizeType * pDataUsed, NChar * arChars, NInt charsLength, NInt *
pCharsUsed);
```

Parameters

Parameters	Description
NEncoding encoding	[in] The encoding to be used.
const void * pData	[in] Pointer to the buffer containing the converted bytes.
NSizeType dataSize	[in] Size of the buffer containing the converted bytes.
NBool dataIsBlock	[in] NTrue [1064] if the byte array is a block-level object; else NFalse [1041].
NSizeType * pDataUsed	[out] Pointer to the number of bytes used.
NChar * arChars	[out] Pointer to the decoded character array.
NInt charsLength	[in] Length of the character array.
NInt * pCharsUsed	[out] Pointer to the number of chars used.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Enumeration

NEncoding Unit [389]

7.1.2.17.1.6 NEncodingGetMaxByteCount Function

Calculates the maximum number of bytes produced by encoding the specified number of characters.

C++

```
NResult N_API NEncodingGetMaxByteCount(NEncoding encoding, NInt charCount, NSizeType *
pByteCount);
```

Parameters

Parameters	Description
NEncoding encoding	[in] The encoding to be used.
NInt charCount	[in] The number of characters.
NSizeType * pByteCount	[out] Pointer to the number of bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Enumeration

NEncoding Unit [389]

7.1.2.17.1.7 NEncodingGetMaxCharCount Function

Calculates the maximum number of characters produced by decoding the specified number of bytes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NEncodingGetMaxCharCount(NEncoding encoding, NSizeType byteCount, NInt *
pCharCount);
```

Parameters

Parameters	Description
NEncoding encoding	[in] The encoding to be used.
NSizeType byteCount	[in] The number of bytes.
NInt * pCharCount	[out] Pointer to the number of characters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Enumeration

NEncoding Unit [389]

7.1.2.17.1.8 NEncodingGetMinBytesPerChar Function

Gets the minimum number of bytes used to encode one character.

C++

```
NResult N_API NEncodingGetMinBytesPerChar(NEncoding encoding, NSizeType * pValue);
```

Parameters

Parameters	Description
NEncoding encoding	[in] The encoding to be used.
NSizeType * pValue	[out] Pointer to the minimal size of an encoded char in bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Enumeration

NEncoding Unit [389]

7.1.2.17.1.9 NEncodingGetPreamble Function

Returns a sequence of bytes that specifies the encoding used.

C++

```
NResult N_API NEncodingGetPreamble(NEncoding encoding, void * pData, NSizeType dataSize,
NSizeType * pDataWritten);
```

Parameters

Parameters	Description
NEncoding encoding	[in] The encoding to be used.
void * pData	[out] Pointer to the buffer containing the preamble.
NSizeType dataSize	[in] Size of the buffer.
NSizeType * pDataWritten	[out] Pointer to the size of the preamble.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Enumeration

NEncoding Unit [389]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.17.1.10 NEncodingGetPreambleLength Function

Gets the length of the preamble of the specified encoding.

C++

```
NResult N_API NEncodingGetPreambleLength(NEncoding encoding, NSizeType * pValue);
```

Parameters

Parameters	Description
NEncoding encoding	[in] The encoding to be used.
NSizeType * pValue	[out] Pointer to the size of the preamble.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Enumeration

NEncoding Unit [389]

7.1.2.17.1.11 NEncodingGetStringN Function

Decodes all the bytes in the specified byte array into a string.

C++

```
NResult N_API NEncodingGetStringN(NEncoding encoding, const void * pData, NSizeType
dataSize, HNString * phValue);
```

Parameters

Parameters	Description
NEncoding encoding	[in] The encoding to be used.
const void * pData	[in] Pointer to the buffer containing the converted bytes.
NSizeType dataSize	[in] Size of the buffer containing the converted bytes.
HNString * phValue	[out] Pointer to a handle to the decoded NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Enumeration

NEncoding Unit [389]

7.1.2.17.1.12 NEncodingIsSingleByte Function

Gets a value indicating whether the specified encoding uses single-byte code points.

C++

```
NResult N_API NEncodingIsSingleByte(NEncoding encoding, NBool * pValue);
```

Parameters

Parameters	Description
NEncoding encoding	[in] The encoding to be checked.
NBool * pValue	[out] NTrue [1064] if the encoding uses single-byte code points; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Enumeration

NEncoding Unit [389]

7.1.2.17.1.13 NEncodingIsValid Function

Checks whether the specified encoding is valid.

C++

```
NBool N_API NEncodingIsValid(NEncoding value);
```

Parameters

Parameters	Description
NEncoding value	[in] The encoding to be checked.

Returns

NTrue [1064] if the encoding is valid; else NFalse [1041].

Enumeration

NEncoding Unit [389]

7.1.2.17.2 Structs, Records, Enums

7.1.2.17.2.1 NEncoding_ Enumeration

Enumerates encoding types.

C++

```
enum NEncoding_ {
  neDefault = 0,
  neUtf8 = 1,
  neAscii = 2,
  neUtf7 = 3,
  neUtf16BE = 4,
  neUtf16LE = 5,
  neUtf32BE = 6,
  neUtf32LE = 7
};
```

Members

Members	Description
neDefault = 0	Specified default encoding.
neUtf8 = 1	Specified UTF-8 encoding.
neAscii = 2	Specified ASCII encoding.
neUtf7 = 3	Specified UTF-7 encoding.
neUtf16BE = 4	Specified UTF-16 BE encoding.
neUtf16LE = 5	Specified UTF-16 LE encoding.
neUtf32BE = 6	Specified UTF-32 BE encoding.
neUtf32LE = 7	Specified UTF-32 LE encoding.

Enumeration

NEncoding Unit [389]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.17.3 Macros

7.1.2.17.3.1 N_ACHAR_ENCODING Macro

Specifies AChar encoding.

C++

```
#define N_ACHAR_ENCODING neDefault
```

Enumeration

NEncoding Unit [389]

7.1.2.17.3.2 N_CHAR_ENCODING Macro

Specifies Char encoding.

C++

```
#define N_CHAR_ENCODING N_ACHAR_ENCODING
```

Enumeration

NEncoding Unit [389]

7.1.2.17.3.3 N_WCHAR_ENCODING Macro

Specifies WChar encoding.

C++

```
#define N_WCHAR_ENCODING neUtf32LE
```

Enumeration

NEncoding Unit [389]

7.1.2.17.3.4 NEncodingGetBytes Macro

Picks a Unicode (W) or ASCII (A) version of the NEncodingGetBytes function.

C++

```
#define NEncodingGetBytes N_FUNC_AW(NEncodingGetBytes)
```

Enumeration

NEncoding Unit [389]

7.1.2.17.3.5 NEncodingGetBytesForChars Macro

Picks a Unicode (W) or ASCII (A) version of the NEncodingGetBytesForChars function.

C++

```
#define NEncodingGetBytesForChars N_FUNC_AW(NEncodingGetBytesForChars)
```

Enumeration

NEncoding Unit [389]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.17.3.6 NEncodingGetChars Macro

Picks a Unicode (W) or ASCII (A) version of the NEncodingGetChars function.

C++

```
#define NEncodingGetChars N_FUNC_AW(NEncodingGetChars)
```

Enumeration

NEncoding Unit [389]

7.1.2.17.3.7 NEncodingGetMaxByteCount Macro

Picks a Unicode (W) or ASCII (A) version of the NEncodingGetMaxByteCount function.

C++

```
#define NEncodingGetMaxByteCount N_FUNC_AW(NEncodingGetMaxByteCount)
```

Enumeration

NEncoding Unit [389]

7.1.2.17.3.8 NEncodingGetMaxCharCount Macro

Picks a Unicode (W) or ASCII (A) version of the NEncodingGetMaxCharCount function.

C++

```
#define NEncodingGetMaxCharCount N_FUNC_AW(NEncodingGetMaxCharCount)
```

Enumeration

NEncoding Unit [389]

7.1.2.18 NError Unit

Defines error codes used in Neurotechnology components.

Files

Name	Description
NError.h [443]	Header file for NError module. Defines error codes used in Neurotechnology components.

Functions

	Name	Description
	NCallStackEntryCopy [403]	
	NCallStackEntryCreate [403]	Creates a line and adds it to NCallStack.
	NCallStackEntryDispose [404]	Frees the specified call stack entry.
	NCallStackEntrySet [404]	Copies a call stack entry and returns a pointer to the new copy. Frees the source stack entry.
	NErrorAppend [404]	Adds a line to NCallStack.
	NErrorGetCallStackCount [405]	Retrieves the count of call stack entries.
	NErrorGetCallStackEntry [405]	Retrieves a certain call stack entry.
	NErrorGetCallStackFileEx [405]	Retrieves the file of a certain call stack entry.
	NErrorGetCallStackFunctionEx [406]	Retrieves the function of a certain call stack entry.
	NErrorGetCallStackLength [406]	Retrieves the length of the call stack.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NErrorGetCallStackLine [406]	Retrieves the specific call stack line.
	NErrorGetCallStackLineEx [407]	Retrieves the specific call stack line.
	NErrorGetCallStackN [407]	Retrieves the last entry of the call stack.
	NErrorGetCode [407]	Retrieves the code of the error.
	NErrorGetCodeEx [408]	Retrieves the code of the error.
	NErrorGetDefaultMessageEx [408]	Retrieves the default error message.
	NErrorGetDefaultMessageN [408]	Retrieves the default error message.
	NErrorGetExternalCallStackEx [409]	Retrieves an external call stack entry with specified size.
	NErrorGetExternalCallStackN [409]	Retrieves an external call stack entry.
	NErrorGetExternalError [410]	Retrieves an external error.
	NErrorGetExternalErrorEx [410]	Retrieves an external error.
	NErrorGetInnerError [410]	Retrieves an inner error.
	NErrorGetInnerErrorEx [411]	Retrieves an inner error.
	NErrorGetLast [411]	Retrieves the last error.
	NErrorGetLastEx [411]	Retrieves the error information pointer set by the previous call to NErrorSetLastEx [418]
	NErrorGetLastSysError [412]	Retrieves the last system error.
	NErrorGetLastWin32Error [412]	Retrieves the last Win32 error.
	NErrorGetMacErrorMessageEx [412]	Retrieves the message of the a specific Mac error.
	NErrorGetMacErrorMessageN [412]	Retrieves the message of the a specific Mac error.
	NErrorGetMachErrorMessageN [413]	Retrieves the message of the a specific Mach error.
	NErrorGetMessageEx [413]	Retrieves the message of the a specific error.
	NErrorGetMessageN [413]	Retrieves the message of the a specific error.
	NErrorGetParamEx [414]	Retrieves the parameters of an error.
	NErrorGetParamN [414]	Retrieves the parameters of an error.
	NErrorGetSysErrorMessageEx [414]	Retrieves the message of the a specific system error.
	NErrorGetSysErrorMessageN [415]	Retrieves the message of the a specific system error.
	NErrorGetWin32ErrorMessageN [415]	Retrieves the message of the a specific Win32 error.
	NErrorReport [416]	Prints the last error out in stderr or debugger output.
	NErrorReportEx [416]	Prints the specified error out in stderr or debugger output.
	NErrorSetCom [416]	Creates the error information object for the current logical thread of execution (COM error only).
	NErrorSetComMPN [417]	Creates the error information object for the current logical thread of execution (COM error only).
	NErrorSetComN [417]	Creates the error information object for the current logical thread of execution (COM error only).
	NErrorSetComPN [417]	Creates the error information object for the current logical thread of execution (COM error only).
	NErrorSetLast [418]	Creates the error information object for the current logical thread of execution.
	NErrorSetLastEx [418]	Sets the error information object for the current logical thread of execution.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NErrorSetLastMPN [419]	Creates the error information object for the current logical thread of execution.
	NErrorSetLastN [419]	Creates the error information object for the current logical thread of execution.
	NErrorSetLastPN [419]	Creates the error information object for the current logical thread of execution.
	NErrorSetMac [420]	Creates the error information object for the current logical thread of execution (Mac only).
	NErrorSetMach [420]	Creates the error information object for the current logical thread of execution (Mach only).
	NErrorSetMachMPN [421]	Creates the error information object for the current logical thread of execution (Mach only).
	NErrorSetMachN [421]	Creates the error information object for the current logical thread of execution (Mach only).
	NErrorSetMachPN [421]	Creates the error information object for the current logical thread of execution (Mach only).
	NErrorSetMacMPN [422]	Creates the error information object for the current logical thread of execution (Mac only).
	NErrorSetMacN [422]	Creates the error information object for the current logical thread of execution (Mac only).
	NErrorSetMacPN [423]	Creates the error information object for the current logical thread of execution (Mac only).
	NErrorSetSys [423]	Creates the error information object for the current logical thread of execution (System errors only).
	NErrorSetSysMPN [423]	Creates the error information object for the current logical thread of execution (System errors only).
	NErrorSetSysN [424]	Creates the error information object for the current logical thread of execution (System errors only).
	NErrorSetSysPN [424]	Creates the error information object for the current logical thread of execution (System errors only).
	NErrorSetWin32 [425]	Creates the error information object for the current logical thread of execution (Win32 only).
	NErrorSetWin32MPN [425]	Creates the error information object for the current logical thread of execution (Win32 only).
	NErrorSetWin32N [425]	Creates the error information object for the current logical thread of execution (Win32 only).
	NErrorSetWin32PN [426]	Creates the error information object for the current logical thread of execution (Win32 only).
	NErrorSuppress [426]	Calls NCoreErrorSuppressed event with the error of the current thread.

Macros

Name	Description
N_E_ARGUMENT [427]	This error occurs when argument value is invalid.
N_E_ABANDONED_MUTEX [427]	This error occurs when one thread acquires a NMutex [534] object that another thread has abandoned by exiting without releasing it.
N_E_ARGUMENT_NULL [427]	When argument value is NULL [1075] where non-NULL [1075] value was expected this error occurs.
N_E_ARGUMENT_OUT_OF_RANGE [428]	Argument value is out of range.
N_E_CLR [428]	The Common language runtime (CLR) exception has occurred.
N_E_COM [428]	COM error has occurred.
N_E_ARITHMETIC [428]	The exception that is thrown for errors in an arithmetic, casting, or conversion operation.
N_E_CORE [428]	Standard error has occurred (for internal use).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_BAD_IMAGE_FORMAT [428]	The file image of a dynamic link library (DLL) or an executable program is invalid.
N_E_END_OF_STREAM [429]	This error occurs when is attempted to read file or buffer after its end.
N_E_EXTERNAL [429]	Error in external code has occurred (for internal use).
N_E_FAILED [429]	Unspecified error has occurred.
N_E_FORMAT [429]	This error occurs when format of argument value is invalid.
N_E_CPP [429]	CPP error has occurred.
N_E_INDEX_OUT_OF_RANGE [430]	This error occurs when index is out of range (for internal use).
N_E_DIRECTORY_NOT_FOUND [430]	A part of a file or directory cannot be found.
N_E_INVALID_OPERATION [430]	Attempted to perform invalid operation.
N_E_DLL_NOT_FOUND [430]	The exception that is thrown when a DLL specified in a DLL import cannot be found.
N_E_IO [430]	Input/output error has occurred.
N_E_DRIVE_NOT_FOUND [430]	Trying to access a drive or share that is not available.
N_E_NOT_ACTIVATED [431]	Product not activated.
N_E_NOT_IMPLEMENTED [431]	This error occurs when trying to use functionality which is not implemented yet.
N_E_ENTRY_POINT_NOT_FOUND [431]	An attempt to load a class failed due to the absence of an entry method.
N_E_NOT_SUPPORTED [431]	This error occurs when trying to use functionality which is not supported.
N_E_NULL_REFERENCE [431]	Null reference has occurred (for internal use).
N_E_FILE_FORMAT [432]	This error occurs when an input file or a data stream that is supposed to conform to a certain file format specification is malformed.
N_E_OUT_OF_MEMORY [432]	This error occurs when there were not enough memory to proceed task.
N_E_FILE_LOAD [432]	A managed assembly is found but cannot be loaded.
N_E_OVERFLOW [432]	Arithmetic overflow has occurred.
N_E_FILE_NOT_FOUND [432]	The error that occurs when an attempt to access a file that does not exist on disk fails.
N_E_PARAMETER [432]	Parameter id is invalid.
N_E_PARAMETER_READ_ONLY [433]	Attempted to set read only parameter.
N_E_SYS [433]	System error.
N_E_INVALID_CAST [433]	The exception that is thrown for invalid casting or explicit conversion.
N_E_WIN32 [433]	Win32 error has occurred.
N_E_INVALID_ENUM_ARGUMENT [433]	This error occurs when using invalid arguments that are enumerators.
N_OK [433]	This value is returned when no error has occurred.
N_E_JVM [434]	JVM error has occurred.
N_E_MAC [434]	MAC error has occurred.
NFailed [434]	Determines whether function result indicates error.
NSucceeded [434]	Determines whether function result indicates success.
N_E_PATH_TOO_LONG [434]	A path or file name is longer than the system-defined maximum length.
N_E_SECURITY [435]	The exception that is thrown when a security error is detected.
N_E_SOCKET [435]	The exception that is thrown when a socket error occurs.
N_E_TIMEOUT [435]	The time allotted for a process or operation has expired.
NCallStackEntryCreate [435]	Picks a Unicode (W) or ANSCII (A) version of the NCallStackEntryCreate function.
NE_IS_DIRECTORY_ACCESS [435]	Indicates that the error has occurred due to directory access, rather than file access (For NErrorSetSys [423] and NErrorSetMac [420]).
NE_MERGE_CALL_STACK [435]	Indicates that the current error's call stack must not be overridden, but merged with the current backtrace.
NE_NO_CALL_STACK [436]	Indicates that the error's call stack does not get overridden when set (normally it gets overridden with the current backtrace).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NE_PRESERVE_INNER_ERROR [436]	Indicates that the current error must not be overridden and that an InenrError must be created.
NE_SKIP_ONE_FRAME [436]	Indicates that additional frame(s) must be skipped.
NErrorAppend [436]	Picks a Unicode (W) or ASCII (A) version of the NErrorAppend function.
NErrorGetCallStackFileEx [436]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetCallStackFileEx function.
NErrorGetCallStackFunctionEx [436]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetCallStackFunctionEx function.
NErrorGetClrError [437]	Gets the CLR error.
NErrorGetComError [437]	Gets the COM error.
NErrorGetDefaultMessageEx [437]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetDefaultMessageEx function.
NErrorGetExternalCallStack [437]	Gets the external call stack.
NErrorGetExternalCallStackEx [438]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetExternalCallStackEx function.
NErrorGetMacErrorMessage [438]	Gets the MacOS error message.
NErrorGetMacErrorMessageEx [438]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetMacErrorMessageEx function.
NErrorGetMessageEx [438]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetMessageEx function.
NErrorGetParam [439]	Retrieves the parameters of an error.
NErrorGetParamEx [439]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetParamEx function.
NErrorGetSysError [439]	Gets the system error.
NErrorGetSysErrorMessageEx [439]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetSysErrorMessageEx function.
NErrorGetWin32Error [440]	Gets the Win32 error.
NErrorSetCom [440]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetCom function.
NErrorSetComMPN [440]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetComMPN function.
NErrorSetComPN [440]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetComPN function.
NErrorSetLast [440]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetLast function.
NErrorSetLastMPN [441]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetLastMPN function.
NErrorSetLastPN [441]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetLastPN function.
NErrorSetMac [441]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetMac function.
NErrorSetMach [441]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetMach function.
NErrorSetMachMPN [441]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetMachMPN function.
NErrorSetMachPN [442]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetMachPN function.
NErrorSetMacMPN [442]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetMacMPN function.
NErrorSetMacPN [442]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetMacPN function.
NErrorSetSys [442]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetSys function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NErrorSetSysMPN [442]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetSysMPN function.
NErrorSetSysPN [442]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetSysPN function.
NErrorSetWin32 [443]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetWin32 function.
NErrorSetWin32MPN [443]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetWin32MPN function.
NErrorSetWin32PN [443]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetWin32PN function.

Structs, Records, Enums

	Name	Description
	NCallStackEntry_ [427]	Represents NCallStackEntry properties. Properties consist of address, function, file and line.

7.1.2.18.1 Functions

7.1.2.18.1.1 NCallStackEntryCopy Function

C++

```
NResult N_API NCallStackEntryCopy(const struct NCallStackEntry_ * pSrcValue, struct
NCallStackEntry_ * pDstValue);
```

Parameters

Parameters	Description
const struct NCallStackEntry_ * pSrcValue	_nt_
struct NCallStackEntry_ * pDstValue	_nt_

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.2 NCallStackEntryCreate Function

Creates a line and adds it to NCallStack.

C++

```
NResult N_API NCallStackEntryCreate(void * addr, const NChar * szFunction, const NChar *
szFile, NInt line, NCallStackEntry * pValue);
```

Parameters

Parameters	Description
void * addr	[in] A pointer to NCallStack that the line is added to.
const NChar * szFunction	[in] A pointer to the function name.
const NChar * szFile	[in] A pointer to the file name.
NInt line	[in] A line to be added to NCallStack.
NCallStackEntry * pValue	[out] A pointer to the created NCallStackEntry.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NError Unit [398]

7.1.2.18.1.3 NCallStackEntryDispose Function

Frees the specified call stack entry.

C++

```
NResult N_API NCallStackEntryDispose(struct NCallStackEntry_ * pValue);
```

Parameters

Parameters	Description
struct NCallStackEntry_ * pValue	[in] A pointer to the call stack entry to be freed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.4 NCallStackEntrySet Function

Copies a call stack entry and returns a pointer to the new copy. Frees the source stack entry.

C++

```
NResult N_API NCallStackEntrySet(const struct NCallStackEntry_ * pSrcValue, struct
NCallStackEntry_ * pDstValue);
```

Parameters

Parameters	Description
const struct NCallStackEntry_ * pSrcValue	[in] A pointer to the source call stack entry.
struct NCallStackEntry_ * pDstValue	[out] A pointer to the destination call stack entry.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.5 NErrorAppend Function

Adds a line to NCallStack.

C++

```
void N_NO_INLINE N_API NErrorAppend(void * addr, const NAChar * szFunction, const NChar *
szFile, NInt line);
```

Parameters

Parameters	Description
void * addr	[in] A pointer to NCallStack that the line is added to.
const NAChar * szFunction	[in] A pointer to the function name.
const NChar * szFile	[in] A pointer to the file name.
NInt line	[in] A line to be added to NCallStack.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NError Unit [398]

7.1.2.18.1.6 NErrorGetCallStackCount Function

Retrieves the count of call stack entries.

C++

```
NResult N_API NErrorGetCallStackCount(HNError hError, NInt * pValue);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
NInt * pValue	[out] A pointer to the number of call stack entries.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.7 NErrorGetCallStackEntry Function

Retrieves a certain call stack entry.

C++

```
NResult N_API NErrorGetCallStackEntry(HNError hError, NInt index, struct NCallStackEntry_ *
pValue);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
NInt index	[in] Index of the call stack entry.
struct NCallStackEntry_ * pValue	[out] A pointer to the call stack entry.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.8 NErrorGetCallStackFileEx Function

Retrieves the file of a certain call stack entry.

C++

```
NResult N_API NErrorGetCallStackFileEx(HNError hError, NInt index, NChar * szValue, NInt
valueSize);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
NInt index	[in] Index of the call stack entry.
NChar * szValue	[out] A pointer to file.
NInt valueSize	[in] Size of the file in bytes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.9 NErrorGetCallStackFunctionEx Function

Retrieves the function of a certain call stack entry.

C++

```
NResult N_API NErrorGetCallStackFunctionEx(HNError hError, NInt index, NChar * szValue,
NInt valueSize);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
NInt index	[in] Index of the call stack entry.
NChar * szValue	[out] A pointer to function.
NInt valueSize	[in] Size of the function in bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.10 NErrorGetCallStackLength Function

Retrieves the length of the call stack.

C++

```
NInt N_API NErrorGetCallStackLength(HNError hError);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.11 NErrorGetCallStackLine Function

Retrieves the specific call stack line.

C++

```
NInt N_API NErrorGetCallStackLine(HNError hError, NInt index);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
NInt index	[in] Index of the line to be retrieved.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This function is deprecated, use NErrorGetCallStackLineEx [407] instead.

Module

NError Unit [398]

7.1.2.18.1.12 NErrorGetCallStackLineEx Function

Retrieves the specific call stack line.

C++

```
NResult N_API NErrorGetCallStackLineEx(HNError hError, NInt index, NInt * pValue);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
NInt index	[in] Index of the line to be retrieved.
NInt * pValue	[out] A link to the call stack line retrieved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.13 NErrorGetCallStackN Function

Retrieves the last entry of the call stack.

C++

```
NResult N_API NErrorGetCallStackN(HNError hError, HNString * phValue);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
HNString * phValue	[out] A pointer to the last line.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.14 NErrorGetCode Function

Retrieves the code of the error.

C++

```
NResult N_API NErrorGetCode(HNError hError);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This function is deprecated, use NErrorGetCodeEx [408] instead.

Module

NError Unit [398]

7.1.2.18.1.15 NErrorGetCodeEx Function

Retrieves the code of the error.

C++

```
NResult N_API NErrorGetCodeEx(HNError hError, NResult * pValue);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
NResult * pValue	[out] A pointer to the code of the error.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.16 NErrorGetDefaultMessageEx Function

Retrieves the default error message.

C++

```
NResult N_API NErrorGetDefaultMessageEx(NResult code, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
NResult code	[in] Error code.
NChar * szValue	[out] A pointer to NChar [1005] value containing the error message.
NInt valueSize	[in] Size of the NChar [1005] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.17 NErrorGetDefaultMessageN Function

Retrieves the default error message.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NErrorGetDefaultMessageN(NResult code, HNString * phValue);
```

Parameters

Parameters	Description
NResult code	[in] The code of NError [398].
HNString * phValue	[out] A pointer to a string containing the message of NError [398].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.18 NErrorGetExternalCallStackEx Function

Retrieves an external call stack entry with specified size.

C++

```
NResult N_API NErrorGetExternalCallStackEx(HNError hError, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
NChar * szValue	[out] A pointer to a NChar [1005] value representing the external call stack entry.
NInt valueSize	[in] Size of the NChar [1005] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.19 NErrorGetExternalCallStackN Function

Retrieves an external call stack entry.

C++

```
NResult N_API NErrorGetExternalCallStackN(HNError hError, HNString * phValue);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
HNString * phValue	[out] A pointer to a string representing the external call stack entry.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.18.1.20 NErrorGetExternalError Function

Retrieves an external error.

C++

```
NInt N_API NErrorGetExternalError(HNError hError);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This function is deprecated, use NErrorGetExternalErrorEx [410] instead.

Module

NError Unit [398]

7.1.2.18.1.21 NErrorGetExternalErrorEx Function

Retrieves an external error.

C++

```
NResult N_API NErrorGetExternalErrorEx(HNError hError, NInt * pValue);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
NInt * pValue	[out] A pointer to the index of the error.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.22 NErrorGetInnerError Function

Retrieves an inner error.

C++

```
HNError N_API NErrorGetInnerError(HNError hError);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This function is deprecated, use NErrorGetInnerErrorEx [411] instead.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NError Unit [398]

7.1.2.18.1.23 NErrorGetInnerErrorEx Function

Retrieves an inner error.

C++

```
NResult N_API NErrorGetInnerErrorEx(HNError hError, HNError * phValue);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
HNError * phValue	[out] A pointer to the index of the error.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.24 NErrorGetLast Function

Retrieves the last error.

C++

```
HNError N_API NErrorGetLast();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This function is deprecated, use NErrorGetLastEx [411] instead.

Module

NError Unit [398]

7.1.2.18.1.25 NErrorGetLastEx Function

Retrieves the error information pointer set by the previous call to NErrorSetLastEx [418]

C++

```
NResult N_API NErrorGetLastEx(NUInt flags, HNError * phError);
```

Parameters

Parameters	Description
NUInt flags	[in] Any flags associated with the error.
HNError * phError	[out] A handle to the last NError [398].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.18.1.26 NErrorGetLastSysError Function

Retrieves the last system error.

C++

```
NInt N_API NErrorGetLastSysError();
```

Returns

0 on Windows CE; otherwise, error number.

Module

NError Unit [398]

7.1.2.18.1.27 NErrorGetLastWin32Error Function

Retrieves the last Win32 error.

C++

```
NUInt N_API NErrorGetLastWin32Error();
```

Returns

Last error number.

Module

NError Unit [398]

7.1.2.18.1.28 NErrorGetMacErrorMessageEx Function

Retrieves the message of the a specific Mac error.

C++

```
NResult N_API NErrorGetMacErrorMessageEx(NInt err, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
NInt err	[in] A NInt [1006] value indicating the error.
NChar * szValue	[out] A pointer to NChar [1005] value containing the error message.
NInt valueSize	[in] Size of the NChar [1005] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.29 NErrorGetMacErrorMessageN Function

Retrieves the message of the a specific Mac error.

C++

```
NResult N_API NErrorGetMacErrorMessageN(NInt err, HNString * phValue);
```

Parameters

Parameters	Description
NInt err	[in] A NInt [1006] value indicating the error.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNString * phValue	[out] A pointer to a string containing the message of NError [398].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.30 NErrorGetMachErrorMessageN Function

Retrieves the message of the a specific Mach error.

C++

```
NResult N_API NErrorGetMachErrorMessageN(NInt err, HNString * phValue);
```

Parameters

Parameters	Description
NInt err	[in] A NInt [1006] value indicating the error.
HNString * phValue	[out] A pointer to a string containing the message of NError [398].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.31 NErrorGetMessageEx Function

Retrieves the message of the a specific error.

C++

```
NResult N_API NErrorGetMessageEx(HNError hError, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
NChar * szValue	[out] A pointer to NChar [1005] value containing the error message.
NInt valueSize	[in] Size of the NChar [1005] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.32 NErrorGetMessageN Function

Retrieves the message of the a specific error.

C++

```
NResult N_API NErrorGetMessageN(HNError hError, HNString * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
HNString * phValue	[out] A pointer to a string containing the message of NError [398].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.33 NErrorGetParamEx Function

Retrieves the parameters of an error.

C++

```
NResult N_API NErrorGetParamEx(HNError hError, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
NChar * szValue	[out] A pointer to a NChar [1005] value containing the parameters of the error.
NInt valueSize	[in] Size of the NChar [1005] value containing the parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.34 NErrorGetParamN Function

Retrieves the parameters of an error.

C++

```
NResult N_API NErrorGetParamN(HNError hError, HNString * phValue);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to NError [398].
HNString * phValue	[out] A pointer to a string containing the parameters of the error.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.35 NErrorGetSysErrorMessageEx Function

Retrieves the message of the a specific system error.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NErrorGetSysErrorMessageEx(NInt errnum, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
NInt errnum	[in] A NInt [1006] value indicating the number of the NError [398].
NChar * szValue	[out] A pointer to a NChar [1005] value containing the message of NError [398].
NInt valueSize	[in] Size of the NChar [1005] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.36 NErrorGetSysErrorMessageN Function

Retrieves the message of the a specific system error.

C++

```
NResult N_API NErrorGetSysErrorMessageN(NInt errnum, HNString * phValue);
```

Parameters

Parameters	Description
NInt errnum	[in] A NInt [1006] value indicating the number of the NError [398].
HNString * phValue	[out] A pointer to a string containing the message of NError [398].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.37 NErrorGetWin32ErrorMessageN Function

Retrieves the message of the a specific Win32 error.

C++

```
NResult N_API NErrorGetWin32ErrorMessageN(NUInt errorCode, HNString * phValue);
```

Parameters

Parameters	Description
NUInt errorCode	[in] A NUInt [1009] value containing the error code.
HNString * phValue	[out] A pointer to a string containing the message of NError [398].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.18.1.38 NErrorReport Function

Prints the last error out in stderr or debugger output.

C++

```
NResult N_NO_INLINE N_API NErrorReport(NResult res);
```

Parameters

Parameters	Description
NResult res	[in] Value returned by the function with error.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.39 NErrorReportEx Function

Prints the specified error out in stderr or debugger output.

C++

```
NResult N_NO_INLINE N_API NErrorReportEx(NResult errorCode, HNError hError);
```

Parameters

Parameters	Description
NResult errorCode	[in] Value returned by the function with error.
HNError hError	[in] A handle to NError [398].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.40 NErrorSetCom Function

Creates the error information object for the current logical thread of execution (COM error only).

C++

```
NResult N_NO_INLINE N_API NErrorSetCom(NInt comError, const NChar * szMessage, const NChar
* szParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt comError	[in] A NInt [1006] value indicating the error code.
const NChar * szMessage	[in] A pointer to the error message.
const NChar * szParam	[in] A pointer to the error parameters.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NError Unit [398]

7.1.2.18.1.41 NErrorSetComMPN Function

Creates the error information object for the current logical thread of execution (COM error only).

C++

```
NResult N_NO_INLINE N_API NErrorSetComMPN(NInt comError, HNString hMessage, HNString
hParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt comError	[in] A NInt [1006] value indicating the error code.
HNString hMessage	[in] A handle to the error message.
HNString hParam	[in] A handle to the error parameter.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.42 NErrorSetComN Function

Creates the error information object for the current logical thread of execution (COM error only).

C++

```
NResult N_NO_INLINE N_API NErrorSetComN(NInt comError, HNString hMessage, HNString hParam,
HNString hExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt comError	[in] A NInt [1006] value indicating the error code.
HNString hMessage	[in] A handle to the error message.
HNString hParam	[in] A handle to the error parameter.
HNString hExternalCallStack	[in] A handle to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.43 NErrorSetComPN Function

Creates the error information object for the current logical thread of execution (COM error only).

C++

```
NResult N_NO_INLINE N_API NErrorSetComPN(NInt comError, const NChar * szMessage, HNString
hParam, const NChar * szExternalCallStack, NUInt flags);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NInt comError	[in] A NInt [1006] value indicating the error code.
const NChar * szMessage	[in] A pointer to the error message.
HNString hParam	[in] A handle to the error parameter.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.44 NErrorSetLast Function

Creates the error information object for the current logical thread of execution.

C++

```
NResult N_NO_INLINE N_API NErrorSetLast(NResult code, const NChar * szMessage, const NChar
* szParam, NInt externalError, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NResult code	[in] The error code.
const NChar * szMessage	[in] A pointer to the error message.
const NChar * szParam	[in] A pointer to the error parameters.
NInt externalError	[in] A pointer to an external error code.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.45 NErrorSetLastEx Function

Sets the error information object for the current logical thread of execution.

C++

```
NResult N_NO_INLINE N_API NErrorSetLastEx(HNError hError, NUInt flags);
```

Parameters

Parameters	Description
HNError hError	[in] A handle to the current error.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.18.1.46 NErrorSetLastMPN Function

Creates the error information object for the current logical thread of execution.

C++

```
NResult N_NO_INLINE N_API NErrorSetLastMPN(NResult code, HNString hMessage, HNString
hParam, NInt externalError, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NResult code	[in] The error code.
HNString hMessage	[in] A handle to the error message.
HNString hParam	[in] A handle to the error parameter.
NInt externalError	[in] A pointer to an external error code.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.47 NErrorSetLastN Function

Creates the error information object for the current logical thread of execution.

C++

```
NResult N_NO_INLINE N_API NErrorSetLastN(NResult code, HNString hMessage, HNString hParam,
NInt externalError, HNString hExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NResult code	[in] The error code.
HNString hMessage	[in] A handle to the error message.
HNString hParam	[in] A handle to the error parameter.
NInt externalError	[in] A handle to an external error code.
HNString hExternalCallStack	[in] A handle to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.48 NErrorSetLastPN Function

Creates the error information object for the current logical thread of execution.

C++

```
NResult N_NO_INLINE N_API NErrorSetLastPN(NResult code, const NChar * szMessage, HNString
hParam, NInt externalError, const NChar * szExternalCallStack, NUInt flags);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NResult code	[in] The error code.
const NChar * szMessage	[in] A pointer to the error message.
HNString hParam	[in] A handle to the error parameter.
NInt externalError	[in] A pointer to an external error code.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.49 NErrorSetMac Function

Creates the error information object for the current logical thread of execution (Mac only).

C++

```
NResult N_NO_INLINE N_API NErrorSetMac(NInt macError, const NChar * szMessage, const NChar
* szParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt macError	[in] A NInt [1006] value indicating the error code.
const NChar * szMessage	[in] A pointer to the error message.
const NChar * szParam	[in] A pointer to the error parameters.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.50 NErrorSetMach Function

Creates the error information object for the current logical thread of execution (Mach only).

C++

```
NResult N_NO_INLINE N_API NErrorSetMach(NInt32 machError, const NChar * szMessage, const
NChar * szParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt32 machError	[in] A NInt32 [1006] value indicating the error code.
const NChar * szMessage	[in] A pointer to the error message.
const NChar * szParam	[in] A pointer to the error parameters.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.51 NErrorSetMachMPN Function

Creates the error information object for the current logical thread of execution (Mach only).

C++

```
NResult N_NO_INLINE N_API NErrorSetMachMPN(NInt32 machError, HNString hMessage, HNString
hParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt32 machError	[in] A NInt32 [1006] value indicating the error code.
HNString hMessage	[in] A handle to the error message.
HNString hParam	[in] A handle to the error parameter.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.52 NErrorSetMachN Function

Creates the error information object for the current logical thread of execution (Mach only).

C++

```
NResult N_NO_INLINE N_API NErrorSetMachN(NInt32 machError, HNString hMessage, HNString
hParam, HNString hExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt32 machError	[in] A NInt32 [1006] value indicating the error code.
HNString hMessage	[in] A handle to the error message.
HNString hParam	[in] A handle to the error parameter.
HNString hExternalCallStack	[in] A handle to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.53 NErrorSetMachPN Function

Creates the error information object for the current logical thread of execution (Mach only).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_NO_INLINE N_API NErrorSetMachPN(NInt32 machError, const NChar * szMessage,
HNString hParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt32 machError	[in] A NInt32 [1006] value indicating the error code.
const NChar * szMessage	[in] A pointer to the error message.
HNString hParam	[in] A handle to the error parameter.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.54 NErrorSetMacMPN Function

Creates the error information object for the current logical thread of execution (Mac only).

C++

```
NResult N_API NErrorSetMacMPN(NInt macError, HNString hMessage, HNString hParam, const
NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt macError	[in] A NInt [1006] value indicating the error code.
HNString hMessage	[in] A handle to the error message.
HNString hParam	[in] A handle to the error parameter.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.55 NErrorSetMacN Function

Creates the error information object for the current logical thread of execution (Mac only).

C++

```
NResult N_NO_INLINE N_API NErrorSetMacN(NInt macError, HNString hMessage, HNString hParam,
HNString hExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt macError	[in] A NInt [1006] value indicating the error code.
HNString hMessage	[in] A handle to the error message.
HNString hParam	[in] A handle to the error parameter.
HNString hExternalCallStack	[in] A handle to an external call stack.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.56 NErrorSetMacPN Function

Creates the error information object for the current logical thread of execution (Mac only).

C++

```
NResult N_NO_INLINE N_API NErrorSetMacPN(NInt macError, const NChar * szMessage, HNString
hParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt macError	[in] A NInt [1006] value indicating the error code.
const NChar * szMessage	[in] A pointer to the error message.
HNString hParam	[in] A handle to the error parameter.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.57 NErrorSetSys Function

Creates the error information object for the current logical thread of execution (System errors only).

C++

```
NResult N_NO_INLINE N_API NErrorSetSys(NInt sysError, const NChar * szMessage, const NChar
* szParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt sysError	[in] A NInt [1006] value indicating the error code.
const NChar * szMessage	[in] A pointer to the error message.
const NChar * szParam	[in] A pointer to the error parameters.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.58 NErrorSetSysMPN Function

Creates the error information object for the current logical thread of execution (System errors only).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_NO_INLINE N_API NErrorSetSysMPN(NInt sysError, HNString hMessage, HNString
hParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt sysError	[in] A NInt [1006] value indicating the error code.
HNString hMessage	[in] A handle to the error message.
HNString hParam	[in] A handle to the error parameter.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.59 NErrorSetSysN Function

Creates the error information object for the current logical thread of execution (System errors only).

C++

```
NResult N_NO_INLINE N_API NErrorSetSysN(NInt sysError, HNString hMessage, HNString hParam,
HNString hExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt sysError	[in] A NInt [1006] value indicating the error code.
HNString hMessage	[in] A handle to the error message.
HNString hParam	[in] A handle to the error parameter.
HNString hExternalCallStack	[in] A handle to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.60 NErrorSetSysPN Function

Creates the error information object for the current logical thread of execution (System errors only).

C++

```
NResult N_NO_INLINE N_API NErrorSetSysPN(NInt sysError, const NChar * szMessage, HNString
hParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NInt sysError	[in] A NInt [1006] value indicating the error code.
const NChar * szMessage	[in] A pointer to the error message.
HNString hParam	[in] A handle to the error parameter.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.61 NErrorSetWin32 Function

Creates the error information object for the current logical thread of execution (Win32 only).

C++

```
NResult N_NO_INLINE N_API NErrorSetWin32(NUInt win32Error, const NChar * szMessage, const
NChar * szParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NUInt win32Error	[in] A NUInt [1009] value indicating the error code.
const NChar * szMessage	[in] A pointer to the error message.
const NChar * szParam	[in] A pointer to the error parameters.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.62 NErrorSetWin32MPN Function

Creates the error information object for the current logical thread of execution (Win32 only).

C++

```
NResult N_NO_INLINE N_API NErrorSetWin32MPN(NUInt win32Error, HNString hMessage, HNString
hParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NUInt win32Error	[in] A NUInt [1009] value indicating the error code.
HNString hMessage	[in] A handle to the error message.
HNString hParam	[in] A handle to the error parameter.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.63 NErrorSetWin32N Function

Creates the error information object for the current logical thread of execution (Win32 only).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_NO_INLINE N_API NErrorSetWin32N(NUInt win32Error, HNString hMessage, HNString
hParam, HNString hExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NUInt win32Error	[in] A NUInt [1009] value indicating the error code.
HNString hMessage	[in] A handle to the error message.
HNString hParam	[in] A handle to the error parameter.
HNString hExternalCallStack	[in] A handle to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.64 NErrorSetWin32PN Function

Creates the error information object for the current logical thread of execution (Win32 only).

C++

```
NResult N_NO_INLINE N_API NErrorSetWin32PN(NUInt win32Error, const NChar * szMessage,
HNString hParam, const NChar * szExternalCallStack, NUInt flags);
```

Parameters

Parameters	Description
NUInt win32Error	[in] A NUInt [1009] value indicating the error code.
const NChar * szMessage	[in] A pointer to the error message.
HNString hParam	[in] A handle to the error parameter.
const NChar * szExternalCallStack	[in] A pointer to an external call stack.
NUInt flags	[in] Flags associated with the thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.1.65 NErrorSuppress Function

Calls NCoreErrorSuppressed event with the error of the current thread.

C++

```
void N_NO_INLINE N_API NErrorSuppress(NResult result);
```

Parameters

Parameters	Description
NResult result	[in] Value returned by the function with error.

Module

NError Unit [398]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.18.2 Structs, Records, Enums

7.1.2.18.2.1 NCallStackEntry_ Structure

Represents NCallStackEntry properties. Properties consist of address, function, file and line.

C++

```
struct NCallStackEntry_ {
  void * addr;
  HNString hFunction;
  HNString hFile;
  NInt line;
};
```

Members

Members	Description
void * addr;	Call stack entry address.
HNString hFunction;	Call stack entry function.
HNString hFile;	Call stack entry file.
NInt line;	Call stack entry line.

Module

NError Unit [398]

7.1.2.18.3 Macros

7.1.2.18.3.1 N_E_ARGUMENT Macro

This error occurs when argument value is invalid.

C++

```
#define N_E_ARGUMENT -10
```

Module

NError Unit [398]

7.1.2.18.3.2 N_E_ABANDONED_MUTEX Macro

This error occurs when one thread acquires a NMutex [534] object that another thread has abandoned by exiting without releasing it.

C++

```
#define N_E_ABANDONED_MUTEX -25
```

Module

NError Unit [398]

7.1.2.18.3.3 N_E_ARGUMENT_NULL Macro

When argument value is NULL [1075] where non-NULL [1075] value was expected this error occurs.

C++

```
#define N_E_ARGUMENT_NULL -11
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NError Unit [398]

7.1.2.18.3.4 N_E_ARGUMENT_OUT_OF_RANGE Macro

Argument value is out of range.

C++

```
#define N_E_ARGUMENT_OUT_OF_RANGE -12
```

Module

NError Unit [398]

7.1.2.18.3.5 N_E_CLR Macro

The Common language runtime (CLR) exception has occurred.

C++

```
#define N_E_CLR -93
```

Module

NError Unit [398]

7.1.2.18.3.6 N_E_COM Macro

COM error has occurred.

C++

```
#define N_E_COM -92
```

Module

NError Unit [398]

7.1.2.18.3.7 N_E_ARITHMETIC Macro

The exception that is thrown for errors in an arithmetic, casting, or conversion operation.

C++

```
#define N_E_ARITHMETIC -17
```

Module

NError Unit [398]

7.1.2.18.3.8 N_E_CORE Macro

Standard error has occurred (for internal use).

C++

```
#define N_E_CORE -2
```

Module

NError Unit [398]

7.1.2.18.3.9 N_E_BAD_IMAGE_FORMAT Macro

The file image of a dynamic link library (DLL) or an executable program is invalid.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_E_BAD_IMAGE_FORMAT -26
```

Module

NError Unit [398]

7.1.2.18.3.10 N_E_END_OF_STREAM Macro

This error occurs when is attempted to read file or buffer after its end.

C++

```
#define N_E_END_OF_STREAM -15
```

Module

NError Unit [398]

7.1.2.18.3.11 N_E_EXTERNAL Macro

Error in external code has occurred (for internal use).

C++

```
#define N_E_EXTERNAL -90
```

Module

NError Unit [398]

7.1.2.18.3.12 N_E_FAILED Macro

Unspecified error has occurred.

C++

```
#define N_E_FAILED -1
```

Module

NError Unit [398]

7.1.2.18.3.13 N_E_FORMAT Macro

This error occurs when format of argument value is invalid.

C++

```
#define N_E_FORMAT -13
```

Module

NError Unit [398]

7.1.2.18.3.14 N_E_CPP Macro

CPP error has occurred.

C++

```
#define N_E_CPP -96
```

Module

NError Unit [398]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.18.3.15 N_E_INDEX_OUT_OF_RANGE Macro

This error occurs when index is out of range (for internal use).

C++

```
#define N_E_INDEX_OUT_OF_RANGE -9
```

Module

NError Unit [398]

7.1.2.18.3.16 N_E_DIRECTORY_NOT_FOUND Macro

A part of a file or directory cannot be found.

C++

```
#define N_E_DIRECTORY_NOT_FOUND -19
```

Module

NError Unit [398]

7.1.2.18.3.17 N_E_INVALID_OPERATION Macro

Attempted to perform invalid operation.

C++

```
#define N_E_INVALID_OPERATION -7
```

Module

NError Unit [398]

7.1.2.18.3.18 N_E_DLL_NOT_FOUND Macro

The exception that is thrown when a DLL specified in a DLL import cannot be found.

C++

```
#define N_E_DLL_NOT_FOUND -27
```

Module

NError Unit [398]

7.1.2.18.3.19 N_E_IO Macro

Input/output error has occurred.

C++

```
#define N_E_IO -14
```

Module

NError Unit [398]

7.1.2.18.3.20 N_E_DRIVE_NOT_FOUND Macro

Trying to access a drive or share that is not available.

C++

```
#define N_E_DRIVE_NOT_FOUND -20
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NError Unit [398]

7.1.2.18.3.21 N_E_NOT_ACTIVATED Macro

Product not activated.

C++

```
#define N_E_NOT_ACTIVATED -200
```

See Also

For more information about product activation see `Activation.pdf` within `documentation` folder of SDK.

Module

NError Unit [398]

7.1.2.18.3.22 N_E_NOT_IMPLEMENTED Macro

This error occurs when trying to use functionality which is not implemented yet.

C++

```
#define N_E_NOT_IMPLEMENTED -5
```

Module

NError Unit [398]

7.1.2.18.3.23 N_E_ENTRY_POINT_NOT_FOUND Macro

An attempt to load a class failed due to the absence of an entry method.

C++

```
#define N_E_ENTRY_POINT_NOT_FOUND -28
```

Module

NError Unit [398]

7.1.2.18.3.24 N_E_NOT_SUPPORTED Macro

This error occurs when trying to use functionality which is not supported.

C++

```
#define N_E_NOT_SUPPORTED -6
```

Module

NError Unit [398]

7.1.2.18.3.25 N_E_NULL_REFERENCE Macro

Null reference has occurred (for internal use).

C++

```
#define N_E_NULL_REFERENCE -3
```

Module

NError Unit [398]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.18.3.26 N_E_FILE_FORMAT Macro

This error occurs when an input file or a data stream that is supposed to conform to a certain file format specification is malformed.

C++

```
#define N_E_FILE_FORMAT -29
```

Module

NError Unit [398]

7.1.2.18.3.27 N_E_OUT_OF_MEMORY Macro

This error occurs when there were not enough memory to proceed task.

C++

```
#define N_E_OUT_OF_MEMORY -4
```

Module

NError Unit [398]

7.1.2.18.3.28 N_E_FILE_LOAD Macro

A managed assembly is found but cannot be loaded.

C++

```
#define N_E_FILE_LOAD -22
```

Module

NError Unit [398]

7.1.2.18.3.29 N_E_OVERFLOW Macro

Arithmetic overflow has occurred.

C++

```
#define N_E_OVERFLOW -8
```

Module

NError Unit [398]

7.1.2.18.3.30 N_E_FILE_NOT_FOUND Macro

The error that occurs when an attempt to access a file that does not exist on disk fails.

C++

```
#define N_E_FILE_NOT_FOUND -21
```

Module

NError Unit [398]

7.1.2.18.3.31 N_E_PARAMETER Macro

Parameter id is invalid.

C++

```
#define N_E_PARAMETER -100
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NError Unit [398]

7.1.2.18.3.32 N_E_PARAMETER_READ_ONLY Macro

Attempted to set read only parameter.

C++

```
#define N_E_PARAMETER_READ_ONLY -101
```

Module

NError Unit [398]

7.1.2.18.3.33 N_E_SYS Macro

System error.

C++

```
#define N_E_SYS -94
```

Module

NError Unit [398]

7.1.2.18.3.34 N_E_INVALID_CAST Macro

The exception that is thrown for invalid casting or explicit conversion.

C++

```
#define N_E_INVALID_CAST -18
```

Module

NError Unit [398]

7.1.2.18.3.35 N_E_WIN32 Macro

Win32 error has occurred.

C++

```
#define N_E_WIN32 -91
```

Module

NError Unit [398]

7.1.2.18.3.36 N_E_INVALID_ENUM_ARGUMENT Macro

This error occurs when using invalid arguments that are enumerators.

C++

```
#define N_E_INVALID_ENUM_ARGUMENT -16
```

Module

NError Unit [398]

7.1.2.18.3.37 N_OK Macro

This value is returned when no error has occurred.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_OK 0
```

Module

NError Unit [398]

7.1.2.18.3.38 N_E_JVM Macro

JVM error has occurred.

C++

```
#define N_E_JVM -97
```

Module

NError Unit [398]

7.1.2.18.3.39 N_E_MAC Macro

MAC error has occurred.

C++

```
#define N_E_MAC -95
```

Module

NError Unit [398]

7.1.2.18.3.40 NFailed Macro

Determines whether function result indicates error.

C++

```
#define NFailed(result) ((result) < 0)
```

Module

NError Unit [398]

7.1.2.18.3.41 NSucceeded Macro

Determines whether function result indicates success.

C++

```
#define NSucceeded(result) ((result) >= 0)
```

Module

NError Unit [398]

7.1.2.18.3.42 N_E_PATH_TOO_LONG Macro

A path or file name is longer than the system-defined maximum length.

C++

```
#define N_E_PATH_TOO_LONG -23
```

Module

NError Unit [398]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.18.3.43 N_E_SECURITY Macro

The exception that is thrown when a security error is detected.

C++

```
#define N_E_SECURITY -24
```

Module

NError Unit [398]

7.1.2.18.3.44 N_E_SOCKET Macro

The exception that is thrown when a socket error occurs.

C++

```
#define N_E_SOCKET -31
```

Module

NError Unit [398]

7.1.2.18.3.45 N_E_TIMEOUT Macro

The time allotted for a process or operation has expired.

C++

```
#define N_E_TIMEOUT -30
```

Module

NError Unit [398]

7.1.2.18.3.46 NCallStackEntryCreate Macro

Picks a Unicode (W) or ANSCII (A) version of the NCallStackEntryCreate function.

C++

```
#define NCallStackEntryCreate N_FUNC_AW(NCallStackEntryCreate)
```

Module

NError Unit [398]

7.1.2.18.3.47 NE_IS_DIRECTORY_ACCESS Macro

Indicates that the error has occurred due to directory access, rather than file access (For NErrorSetSys [423] and NErrorSetMac [420]).

C++

```
#define NE_IS_DIRECTORY_ACCESS 0x00000100
```

Module

NError Unit [398]

7.1.2.18.3.48 NE_MERGE_CALL_STACK Macro

Indicates that the current error's call stack must not be overridden, but merged with the current backtrace.

C++

```
#define NE_MERGE_CALL_STACK 0x00000004
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NError Unit [398]

7.1.2.18.3.49 NE_NO_CALL_STACK Macro

Indicates that the error's call stack does not get overridden when set (normally it gets overridden with the current backtrace).

C++

```
#define NE_NO_CALL_STACK 0x00000002
```

Module

NError Unit [398]

7.1.2.18.3.50 NE_PRESERVE_INNER_ERROR Macro

Indicates that the current error must not be overridden and that an InenrError must be created.

C++

```
#define NE_PRESERVE_INNER_ERROR 0x00000001
```

Module

NError Unit [398]

7.1.2.18.3.51 NE_SKIP_ONE_FRAME Macro

Indicates that additional frame(s) must be skipped.

C++

```
#define NE_SKIP_ONE_FRAME 0x01000000
```

Module

NError Unit [398]

7.1.2.18.3.52 NErrorAppend Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorAppend function.

C++

```
#define NErrorAppend N_FUNC_AW(NErrorAppend)
```

Module

NError Unit [398]

7.1.2.18.3.53 NErrorGetCallStackFileEx Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorGetCallStackFileEx function.

C++

```
#define NErrorGetCallStackFileEx N_FUNC_AW(NErrorGetCallStackFileEx)
```

Module

NError Unit [398]

7.1.2.18.3.54 NErrorGetCallStackFunctionEx Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorGetCallStackFunctionEx function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NErrorGetCallStackFunctionEx N_FUNC_AW(NErrorGetCallStackFunctionEx)
```

Module

NError Unit [398]

7.1.2.18.3.55 NErrorGetClrError Macro

Gets the CLR error.

C++

```
#define NErrorGetClrError(hError) NULL
```

Parameters

Parameters	Description
hError	Handle to the error.

Module

NError Unit [398]

7.1.2.18.3.56 NErrorGetComError Macro

Gets the COM error.

C++

```
#define NErrorGetComError(hError) (NErrorGetCode(hError) == N_E_COM ?
NErrorGetExternalError(HNError hError) : 0)
```

Parameters

Parameters	Description
hError	Handle to the error.

Module

NError Unit [398]

7.1.2.18.3.57 NErrorGetDefaultMessageEx Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorGetDefaultMessageEx function.

C++

```
#define NErrorGetDefaultMessageEx N_FUNC_AW(NErrorGetDefaultMessageEx)
```

Module

NError Unit [398]

7.1.2.18.3.58 NErrorGetExternalCallStack Macro

Gets the external call stack.

C++

```
#define NErrorGetExternalCallStack(hError, szValue) NErrorGetExternalCallStackEx(hError,
szValue, N_INT_MAX)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
hError	[in] Handle to NError [398].
szValue	[out] Pointer to the external call stack.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.3.59 NErrorGetExternalCallStackEx Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorGetExternalCallStackEx function.

C++

```
#define NErrorGetExternalCallStackEx N_FUNC_AW(NErrorGetExternalCallStackEx)
```

Module

NError Unit [398]

7.1.2.18.3.60 NErrorGetMacErrorMessage Macro

Gets the MacOS error message.

C++

```
#define NErrorGetMacErrorMessage(err, szValue) NErrorGetMacErrorMessageEx(err, szValue,
N_INT_MAX)
```

Parameters

Parameters	Description
err	[in] A NInt [1006] value indicating the error.
szValue	[out] A pointer to NChar [1005] value containing the error message.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.3.61 NErrorGetMacErrorMessageEx Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorGetMacErrorMessageEx function.

C++

```
#define NErrorGetMacErrorMessageEx N_FUNC_AW(NErrorGetMacErrorMessageEx)
```

Module

NError Unit [398]

7.1.2.18.3.62 NErrorGetMessageEx Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorGetMessageEx function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NErrorGetMessageEx N_FUNC_AW(NErrorGetMessageEx)
```

Module

NError Unit [398]

7.1.2.18.3.63 NErrorGetParam Macro

Retrieves the parameters of an error.

C++

```
#define NErrorGetParam(hError, szValue) NErrorGetParamEx(hError, szValue, N_INT_MAX)
```

Parameters

Parameters	Description
hError	[in] A handle to NError [398].
szValue	[out] A pointer to a NChar [1005] value containing the parameters of the error.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NError Unit [398]

7.1.2.18.3.64 NErrorGetParamEx Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorGetParamEx function.

C++

```
#define NErrorGetParamEx N_FUNC_AW(NErrorGetParamEx)
```

Module

NError Unit [398]

7.1.2.18.3.65 NErrorGetSysError Macro

Gets the system error.

C++

```
#define NErrorGetSysError(hError) (NErrorGetCode(hError) == N_E_SYS ?
NErrorGetExternalError(HNError hError) : 0)
```

Parameters

Parameters	Description
hError	Handle to the error.

Module

NError Unit [398]

7.1.2.18.3.66 NErrorGetSysErrorMessageEx Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorGetSysErrorMessageEx function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NErrorGetSysErrorMessageEx N_FUNC_AW(NErrorGetSysErrorMessageEx)
```

Module

NError Unit [398]

7.1.2.18.3.67 NErrorGetWin32Error Macro

Gets the Win32 error.

C++

```
#define NErrorGetWin32Error(hError) (NErrorGetCode(hError) == N_E_WIN32 ?
(NUInt)NErrorGetExternalError(HNError hError) : 0)
```

Parameters

Parameters	Description
hError	Handle to the error.

Module

NError Unit [398]

7.1.2.18.3.68 NErrorSetCom Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetCom function.

C++

```
#define NErrorSetCom N_FUNC_AW(NErrorSetCom)
```

Module

NError Unit [398]

7.1.2.18.3.69 NErrorSetComMPN Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetComMPN function.

C++

```
#define NErrorSetComMPN N_FUNC_AW(NErrorSetComMPN)
```

Module

NError Unit [398]

7.1.2.18.3.70 NErrorSetComPN Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetComPN function.

C++

```
#define NErrorSetComPN N_FUNC_AW(NErrorSetComPN)
```

Module

NError Unit [398]

7.1.2.18.3.71 NErrorSetLast Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetLast function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NErrorSetLast N_FUNC_AW(NErrorSetLast)
```

Module

NError Unit [398]

7.1.2.18.3.72 NErrorSetLastMPN Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetLastMPN function.

C++

```
#define NErrorSetLastMPN N_FUNC_AW(NErrorSetLastMPN)
```

Module

NError Unit [398]

7.1.2.18.3.73 NErrorSetLastPN Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetLastPN function.

C++

```
#define NErrorSetLastPN N_FUNC_AW(NErrorSetLastPN)
```

Module

NError Unit [398]

7.1.2.18.3.74 NErrorSetMac Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetMac function.

C++

```
#define NErrorSetMac N_FUNC_AW(NErrorSetMac)
```

Module

NError Unit [398]

7.1.2.18.3.75 NErrorSetMach Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetMach function.

C++

```
#define NErrorSetMach N_FUNC_AW(NErrorSetMach)
```

Module

NError Unit [398]

7.1.2.18.3.76 NErrorSetMachMPN Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetMachMPN function.

C++

```
#define NErrorSetMachMPN N_FUNC_AW(NErrorSetMachMPN)
```

Module

NError Unit [398]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.18.3.77 NErrorSetMachPN Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetMachPN function.

C++

```
#define NErrorSetMachPN N_FUNC_AW(NErrorSetMachPN)
```

Module

NError Unit [398]

7.1.2.18.3.78 NErrorSetMacMPN Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetMacMPN function.

C++

```
#define NErrorSetMacMPN N_FUNC_AW(NErrorSetMacMPN)
```

Module

NError Unit [398]

7.1.2.18.3.79 NErrorSetMacPN Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetMacPN function.

C++

```
#define NErrorSetMacPN N_FUNC_AW(NErrorSetMacPN)
```

Module

NError Unit [398]

7.1.2.18.3.80 NErrorSetSys Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetSys function.

C++

```
#define NErrorSetSys N_FUNC_AW(NErrorSetSys)
```

Module

NError Unit [398]

7.1.2.18.3.81 NErrorSetSysMPN Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetSysMPN function.

C++

```
#define NErrorSetSysMPN N_FUNC_AW(NErrorSetSysMPN)
```

Module

NError Unit [398]

7.1.2.18.3.82 NErrorSetSysPN Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetSysPN function.

C++

```
#define NErrorSetSysPN N_FUNC_AW(NErrorSetSysPN)
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NError Unit [398]

7.1.2.18.3.83 NErrorSetWin32 Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetWin32 function.

C++

```
#define NErrorSetWin32 N_FUNC_AW(NErrorSetWin32)
```

Module

NError Unit [398]

7.1.2.18.3.84 NErrorSetWin32MPN Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetWin32MPN function.

C++

```
#define NErrorSetWin32MPN N_FUNC_AW(NErrorSetWin32MPN)
```

Module

NError Unit [398]

7.1.2.18.3.85 NErrorSetWin32PN Macro

Picks a Unicode (W) or ASCII (A) version of the NErrorSetWin32PN function.

C++

```
#define NErrorSetWin32PN N_FUNC_AW(NErrorSetWin32PN)
```

Module

NError Unit [398]

7.1.2.18.4 Files

7.1.2.18.4.1 NError.h

Header file for NError [398] module. Defines error codes used in Neurotechnology components.

Functions

	Name	Description
	NCallStackEntryCopy [403]	
	NCallStackEntryCreate [403]	Creates a line and adds it to NCallStack.
	NCallStackEntryDispose [404]	Frees the specified call stack entry.
	NCallStackEntrySet [404]	Copies a call stack entry and returns a pointer to the new copy. Frees the source stack entry.
	NErrorAppend [404]	Adds a line to NCallStack.
	NErrorGetCallStackCount [405]	Retrieves the count of call stack entries.
	NErrorGetCallStackEntry [405]	Retrieves a certain call stack entry.
	NErrorGetCallStackFileEx [405]	Retrieves the file of a certain call stack entry.
	NErrorGetCallStackFunctionEx [406]	Retrieves the function of a certain call stack entry.
	NErrorGetCallStackLength [406]	Retrieves the length of the call stack.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NErrorGetCallStackLine [406]	Retrieves the specific call stack line.
	NErrorGetCallStackLineEx [407]	Retrieves the specific call stack line.
	NErrorGetCallStackN [407]	Retrieves the last entry of the call stack.
	NErrorGetCode [407]	Retrieves the code of the error.
	NErrorGetCodeEx [408]	Retrieves the code of the error.
	NErrorGetDefaultMessageEx [408]	Retrieves the default error message.
	NErrorGetDefaultMessageN [408]	Retrieves the default error message.
	NErrorGetExternalCallStackEx [409]	Retrieves an external call stack entry with specified size.
	NErrorGetExternalCallStackN [409]	Retrieves an external call stack entry.
	NErrorGetExternalError [410]	Retrieves an external error.
	NErrorGetExternalErrorEx [410]	Retrieves an external error.
	NErrorGetInnerError [410]	Retrieves an inner error.
	NErrorGetInnerErrorEx [411]	Retrieves an inner error.
	NErrorGetLast [411]	Retrieves the last error.
	NErrorGetLastEx [411]	Retrieves the error information pointer set by the previous call to NErrorSetLastEx [418]
	NErrorGetLastSysError [412]	Retrieves the last system error.
	NErrorGetLastWin32Error [412]	Retrieves the last Win32 error.
	NErrorGetMacErrorMessageEx [412]	Retrieves the message of the a specific Mac error.
	NErrorGetMacErrorMessageN [412]	Retrieves the message of the a specific Mac error.
	NErrorGetMachErrorMessageN [413]	Retrieves the message of the a specific Mach error.
	NErrorGetMessageEx [413]	Retrieves the message of the a specific error.
	NErrorGetMessageN [413]	Retrieves the message of the a specific error.
	NErrorGetParamEx [414]	Retrieves the parameters of an error.
	NErrorGetParamN [414]	Retrieves the parameters of an error.
	NErrorGetSysErrorMessageEx [414]	Retrieves the message of the a specific system error.
	NErrorGetSysErrorMessageN [415]	Retrieves the message of the a specific system error.
	NErrorGetWin32ErrorMessageN [415]	Retrieves the message of the a specific Win32 error.
	NErrorReport [416]	Prints the last error out in stderr or debugger output.
	NErrorReportEx [416]	Prints the specified error out in stderr or debugger output.
	NErrorSetCom [416]	Creates the error information object for the current logical thread of execution (COM error only).
	NErrorSetComMPN [417]	Creates the error information object for the current logical thread of execution (COM error only).
	NErrorSetComN [417]	Creates the error information object for the current logical thread of execution (COM error only).
	NErrorSetComPN [417]	Creates the error information object for the current logical thread of execution (COM error only).
	NErrorSetLast [418]	Creates the error information object for the current logical thread of execution.
	NErrorSetLastEx [418]	Sets the error information object for the current logical thread of execution.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NErrorSetLastMPN [419]	Creates the error information object for the current logical thread of execution.
	NErrorSetLastN [419]	Creates the error information object for the current logical thread of execution.
	NErrorSetLastPN [419]	Creates the error information object for the current logical thread of execution.
	NErrorSetMac [420]	Creates the error information object for the current logical thread of execution (Mac only).
	NErrorSetMach [420]	Creates the error information object for the current logical thread of execution (Mach only).
	NErrorSetMachMPN [421]	Creates the error information object for the current logical thread of execution (Mach only).
	NErrorSetMachN [421]	Creates the error information object for the current logical thread of execution (Mach only).
	NErrorSetMachPN [421]	Creates the error information object for the current logical thread of execution (Mach only).
	NErrorSetMacMPN [422]	Creates the error information object for the current logical thread of execution (Mac only).
	NErrorSetMacN [422]	Creates the error information object for the current logical thread of execution (Mac only).
	NErrorSetMacPN [423]	Creates the error information object for the current logical thread of execution (Mac only).
	NErrorSetSys [423]	Creates the error information object for the current logical thread of execution (System errors only).
	NErrorSetSysMPN [423]	Creates the error information object for the current logical thread of execution (System errors only).
	NErrorSetSysN [424]	Creates the error information object for the current logical thread of execution (System errors only).
	NErrorSetSysPN [424]	Creates the error information object for the current logical thread of execution (System errors only).
	NErrorSetWin32 [425]	Creates the error information object for the current logical thread of execution (Win32 only).
	NErrorSetWin32MPN [425]	Creates the error information object for the current logical thread of execution (Win32 only).
	NErrorSetWin32N [425]	Creates the error information object for the current logical thread of execution (Win32 only).
	NErrorSetWin32PN [426]	Creates the error information object for the current logical thread of execution (Win32 only).
	NErrorSuppress [426]	Calls NCoreErrorSuppressed event with the error of the current thread.

Macros

Name	Description
N_E_ABANDONED_MUTEX [427]	This error occurs when one thread acquires a NMutex [534] object that another thread has abandoned by exiting without releasing it.
N_E_ARGUMENT [427]	This error occurs when argument value is invalid.
N_E_ARGUMENT_NULL [427]	When argument value is NULL [1075] where non-NULL [1075] value was expected this error occurs.
N_E_ARGUMENT_OUT_OF_RANGE [428]	Argument value is out of range.
N_E_ARITHMETIC [428]	The exception that is thrown for errors in an arithmetic, casting, or conversion operation.
N_E_BAD_IMAGE_FORMAT [428]	The file image of a dynamic link library (DLL) or an executable program is invalid.
N_E_CLR [428]	The Common language runtime (CLR) exception has occurred.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_COM [428]	COM error has occurred.
N_E_CORE [428]	Standard error has occurred (for internal use).
N_E_CPP [429]	CPP error has occurred.
N_E_DIRECTORY_NOT_FOUND [430]	A part of a file or directory cannot be found.
N_E_DLL_NOT_FOUND [430]	The exception that is thrown when a DLL specified in a DLL import cannot be found.
N_E_DRIVE_NOT_FOUND [430]	Trying to access a drive or share that is not available.
N_E_END_OF_STREAM [429]	This error occurs when is attempted to read file or buffer after its end.
N_E_ENTRY_POINT_NOT_FOUND [431]	An attempt to load a class failed due to the absence of an entry method.
N_E_EXTERNAL [429]	Error in external code has occurred (for internal use).
N_E_FAILED [429]	Unspecified error has occurred.
N_E_FILE_FORMAT [432]	This error occurs when an input file or a data stream that is supposed to conform to a certain file format specification is malformed.
N_E_FILE_LOAD [432]	A managed assembly is found but cannot be loaded.
N_E_FILE_NOT_FOUND [432]	The error that occurs when an attempt to access a file that does not exist on disk fails.
N_E_FORMAT [429]	This error occurs when format of argument value is invalid.
N_E_INDEX_OUT_OF_RANGE [430]	This error occurs when index is out of range (for internal use).
N_E_INVALID_CAST [433]	The exception that is thrown for invalid casting or explicit conversion.
N_E_INVALID_ENUM_ARGUMENT [433]	This error occurs when using invalid arguments that are enumerators.
N_E_INVALID_OPERATION [430]	Attempted to perform invalid operation.
N_E_IO [430]	Input/output error has occurred.
N_E_JVM [434]	JVM error has occurred.
N_E_MAC [434]	MAC error has occurred.
N_E_NOT_ACTIVATED [431]	Product not activated.
N_E_NOT_IMPLEMENTED [431]	This error occurs when trying to use functionality which is not implemented yet.
N_E_NOT_SUPPORTED [431]	This error occurs when trying to use functionality which is not supported.
N_E_NULL_REFERENCE [431]	Null reference has occurred (for internal use).
N_E_OUT_OF_MEMORY [432]	This error occurs when there were not enough memory to proceed task.
N_E_OVERFLOW [432]	Arithmetic overflow has occurred.
N_E_PARAMETER [432]	Parameter id is invalid.
N_E_PARAMETER_READ_ONLY [433]	Attempted to set read only parameter.
N_E_PATH_TOO_LONG [434]	A path or file name is longer than the system-defined maximum length.
N_E_SECURITY [435]	The exception that is thrown when a security error is detected.
N_E_SOCKET [435]	The exception that is thrown when a socket error occurs.
N_E_SYS [433]	System error.
N_E_TIMEOUT [435]	The time allotted for a process or operation has expired.
N_E_WIN32 [433]	Win32 error has occurred.
N_OK [433]	This value is returned when no error has occurred.
NCallStackEntryCreate [435]	Picks a Unicode (W) or ANSCII (A) version of the NCallStackEntryCreate function.
NE_IS_DIRECTORY_ACCESS [435]	Indicates that the error has occurred due to directory access, rather than file access (For NErrorSetSys [423] and NErrorSetMac [420]).
NE_MERGE_CALL_STACK [435]	Indicates that the current error's call stack must not be overridden, but merged with the current backtrace.
NE_NO_CALL_STACK [436]	Indicates that the error's call stack does not get overridden when set (normally it gets overridden with the current backtrace).
NE_PRESERVE_INNER_ERROR [436]	Indicates that the current error must not be overridden and that an InenrError must be created.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NE_SKIP_ONE_FRAME [436]	Indicates that additional frame(s) must be skipped.
NErrorAppend [436]	Picks a Unicode (W) or ASCII (A) version of the NErrorAppend function.
NErrorGetCallStackFileEx [436]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetCallStackFileEx function.
NErrorGetCallStackFunctionEx [436]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetCallStackFunctionEx function.
NErrorGetClrError [437]	Gets the CLR error.
NErrorGetComError [437]	Gets the COM error.
NErrorGetDefaultMessageEx [437]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetDefaultMessageEx function.
NErrorGetExternalCallStack [437]	Gets the external call stack.
NErrorGetExternalCallStackEx [438]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetExternalCallStackEx function.
NErrorGetMacErrorMessage [438]	Gets the MacOS error message.
NErrorGetMacErrorMessageEx [438]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetMacErrorMessageEx function.
NErrorGetMessageEx [438]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetMessageEx function.
NErrorGetParam [439]	Retrieves the parameters of an error.
NErrorGetParamEx [439]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetParamEx function.
NErrorGetSysError [439]	Gets the system error.
NErrorGetSysErrorMessageEx [439]	Picks a Unicode (W) or ASCII (A) version of the NErrorGetSysErrorMessageEx function.
NErrorGetWin32Error [440]	Gets the Win32 error.
NErrorSetCom [440]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetCom function.
NErrorSetComMPN [440]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetComMPN function.
NErrorSetComPN [440]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetComPN function.
NErrorSetLast [440]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetLast function.
NErrorSetLastMPN [441]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetLastMPN function.
NErrorSetLastPN [441]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetLastPN function.
NErrorSetMac [441]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetMac function.
NErrorSetMach [441]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetMach function.
NErrorSetMachMPN [441]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetMachMPN function.
NErrorSetMachPN [442]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetMachPN function.
NErrorSetMacMPN [442]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetMacMPN function.
NErrorSetMacPN [442]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetMacPN function.
NErrorSetSys [442]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetSys function.
NErrorSetSysMPN [442]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetSysMPN function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NErrorSetSysPN [442]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetSysPN function.
NErrorSetWin32 [443]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetWin32 function.
NErrorSetWin32MPN [443]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetWin32MPN function.
NErrorSetWin32PN [443]	Picks a Unicode (W) or ASCII (A) version of the NErrorSetWin32PN function.
NFailed [434]	Determines whether function result indicates error.
NSucceeded [434]	Determines whether function result indicates success.

Module

NError Unit [398]

Structures

	Name	Description
	NCallStackEntry_ [427]	Represents NCallStackEntry properties. Properties consist of address, function, file and line.

7.1.2.19 NEvent Unit

Files

Name	Description
NEvent.h [457]	Header file for the NEvent unit.

Functions

	Name	Description
	NEventAddN [449]	Adds a new event with the specified callback.
	NEventAddRaw [449]	Adds a new event.
	NEventDispose [449]	Disposes of the specified event.
	NEventGetCallbacks [450]	Retrieves all callbacks associated with the event.
	NEventInit [450]	Initializes the specified event.
	NEventIsEmpty [450]	
	NEventReleaseCallbacks [451]	Releases all callbacks associated with the event.
	NEventRemoveN [451]	Removes the specified event with the specified callback.
	NEventRemoveRaw [452]	Removes the specified event.

Macros

Name	Description
N_EVENT_RAISE_1 [452]	Raises the specified event with specified arguments.
N_EVENT_RAISE_2 [452]	Raises the specified event with specified arguments.
N_EVENT_RAISE_3 [453]	Raises the specified event with specified arguments.
N_EVENT_RAISE_4 [453]	Raises the specified event with specified arguments.
N_EVENT_RAISE_5 [454]	Raises the specified event with specified arguments.
N_EVENT_RAISE_6 [454]	Raises the specified event with specified arguments.
N_EVENT_RAISE_RAW [455]	Raises the specified event with specified arguments.
N_EVENT_SIZE [455]	Defines the size of the event.
NEventAdd [456]	Adds a new event.
NEventRemove [456]	Removes the specified event.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.19.1 Functions

7.1.2.19.1.1 NEventAddN Function

Adds a new event with the specified callback.

C++

```
NResult N_API NEventAddN(NEvent * pEvent, HNCallback hCallback);
```

Parameters

Parameters	Description
NEvent * pEvent	[in/out] Pointer to the event.
HNCallback hCallback	[in] Handle to the event callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

Raw functions (i.e. NEventAddRaw [449]) represent the functions themselves.

Non-raw functions (i.e. NEventAdd [456]) represent macros that call those functions for the convenience of using them in C++.

Module

NEvent Unit [448]

7.1.2.19.1.2 NEventAddRaw Function

Adds a new event.

C++

```
NResult N_API NEventAddRaw(NEvent * pEvent, void * pProc, void * pParam);
```

Parameters

Parameters	Description
NEvent * pEvent	[in/out] Pointer to the event.
void * pProc	[in/out] Pointer to the process.
void * pParam	[in/out] Pointer to the parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

Raw functions (i.e. NEventAddRaw) represent the functions themselves.

Non-raw functions (i.e. NEventAdd [456]) represent macros that call those functions for the convenience of using them in C++.

Module

NEvent Unit [448]

7.1.2.19.1.3 NEventDispose Function

Disposes of the specified event.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NEventDispose(NEvent * pEvent);
```

Parameters

Parameters	Description
NEvent * pEvent	[in/out] Pointer to the event.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NEvent Unit [448]

7.1.2.19.1.4 NEventGetCallbacks Function

Retrieves all callbacks associated with the event.

C++

```
NResult N_API NEventGetCallbacks(NEvent * pEvent, HNCallback * * parhCallbacks, NInt *
pCount);
```

Parameters

Parameters	Description
NEvent * pEvent	[in/out] Pointer to the event.
HNCallback * * parhCallbacks	[out] Pointer to an array of callback handles.
NInt * pCount	[out] Pointer to the number of callbacks.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NEvent Unit [448]

7.1.2.19.1.5 NEventInit Function

Initializes the specified event.

C++

```
NResult N_API NEventInit(NEvent * pEvent);
```

Parameters

Parameters	Description
NEvent * pEvent	[in/out] Pointer to the event.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NEvent Unit [448]

7.1.2.19.1.6 NEventIsEmpty Function

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NEventIsEmpty(NEvent * pEvent, NBool * pResult);
```

Parameters

Parameters	Description
NEvent * pEvent	[in/out] Pointer to the event.
NBool * pResult	_nt_

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NEvent Unit [448]

7.1.2.19.1.7 NEventReleaseCallbacks Function

Releases all callbacks associated with the event.

C++

```
NResult N_API NEventReleaseCallbacks(NEvent * pEvent, HNCallback * arhCallbacks, NInt
count);
```

Parameters

Parameters	Description
NEvent * pEvent	[in/out] Pointer to the event.
HNCallback * arhCallbacks	[out] Pointer to an array of callback handles.
NInt count	[in] Number of callbacks.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NEvent Unit [448]

7.1.2.19.1.8 NEventRemoveN Function

Removes the specified event with the specified callback.

C++

```
NResult N_API NEventRemoveN(NEvent * pEvent, HNCallback hCallback);
```

Parameters

Parameters	Description
NEvent * pEvent	[in/out] Pointer to the event.
HNCallback hCallback	[in] Handle to the event callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

Raw functions (i.e. NEventAddRaw [449]) represent the functions themselves.

Non-raw functions (i.e. NEventAdd [456]) represent macros that call those functions for the convenience of using them in C++.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NEvent Unit [448]

7.1.2.19.1.9 NEventRemoveRaw Function

Removes the specified event.

C++

```
NResult N_API NEventRemoveRaw(NEvent * pEvent, void * pProc, void * pParam);
```

Parameters

Parameters	Description
NEvent * pEvent	[in/out] Pointer to the event.
void * pProc	[in/out] Pointer to the process.
void * pParam	[in/out] Pointer to the parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

Raw functions (i.e. NEventAddRaw [449]) represent the functions themselves.

Non-raw functions (i.e. NEventAdd [456]) represent macros that call those functions for the convenience of using them in C++.

Module

NEvent Unit [448]

7.1.2.19.2 Macros

7.1.2.19.2.1 N_EVENT_RAISE_1 Macro

Raises the specified event with specified arguments.

C++

```
#define N_EVENT_RAISE_1(callbackType, pEvent, arg1) N_EVENT_RAISE_RAW(callbackType, pEvent,
((arg1), NCallbackGetParam(N_EVENT_RAISE_RAW_hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
pEvent	[in/out] Pointer to the event.
arg1	[in] Event argument.

Returns

The process that raises the event.

Module

NEvent Unit [448]

7.1.2.19.2.2 N_EVENT_RAISE_2 Macro

Raises the specified event with specified arguments.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_EVENT_RAISE_2(callbackType, pEvent, arg1, arg2) N_EVENT_RAISE_RAW(callbackType,
pEvent, ((arg1), (arg2), NCallbackGetParam(N_EVENT_RAISE_RAW_hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
pEvent	[in/out] Pointer to the event.
arg1	[in] Event argument.
arg2	[in] Event argument.

Returns

The process that raises the event.

Module

NEvent Unit [448]

7.1.2.19.2.3 N_EVENT_RAISE_3 Macro

Raises the specified event with specified arguments.

C++

```
#define N_EVENT_RAISE_3(callbackType, pEvent, arg1, arg2, arg3)
N_EVENT_RAISE_RAW(callbackType, pEvent, ((arg1), (arg2), (arg3),
NCallbackGetParam(N_EVENT_RAISE_RAW_hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
pEvent	[in/out] Pointer to the event.
arg1	[in] Event argument.
arg2	[in] Event argument.
arg3	[in] Event argument.

Returns

The process that raises the event.

Module

NEvent Unit [448]

7.1.2.19.2.4 N_EVENT_RAISE_4 Macro

Raises the specified event with specified arguments.

C++

```
#define N_EVENT_RAISE_4(callbackType, pEvent, arg1, arg2, arg3, arg4)
N_EVENT_RAISE_RAW(callbackType, pEvent, ((arg1), (arg2), (arg3), (arg4),
NCallbackGetParam(N_EVENT_RAISE_RAW_hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
pEvent	[in/out] Pointer to the event.
arg1	[in] Event argument.
arg2	[in] Event argument.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

arg3	[in] Event argument.
arg4	[in] Event argument.

Returns

The process that raises the event.

Module

NEvent Unit [448]

7.1.2.19.2.5 N_EVENT_RAISE_5 Macro

Raises the specified event with specified arguments.

C++

```
#define N_EVENT_RAISE_5(callbackType, pEvent, arg1, arg2, arg3, arg4, arg5)
N_EVENT_RAISE_RAW(callbackType, pEvent, ((arg1), (arg2), (arg3), (arg4), (arg5),
NCallbackGetParam(N_EVENT_RAISE_RAW_hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
pEvent	[in/out] Pointer to the event.
arg1	[in] Event argument.
arg2	[in] Event argument.
arg3	[in] Event argument.
arg4	[in] Event argument.
arg5	[in] Event argument.

Returns

The process that raises the event.

Module

NEvent Unit [448]

7.1.2.19.2.6 N_EVENT_RAISE_6 Macro

Raises the specified event with specified arguments.

C++

```
#define N_EVENT_RAISE_6(callbackType, pEvent, arg1, arg2, arg3, arg4, arg5, arg6)
N_EVENT_RAISE_RAW(callbackType, pEvent, ((arg1), (arg2), (arg3), (arg4), (arg5), (arg6),
NCallbackGetParam(N_EVENT_RAISE_RAW_hCallback)))
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
pEvent	[in/out] Pointer to the event.
arg1	[in] Event argument.
arg2	[in] Event argument.
arg3	[in] Event argument.
arg4	[in] Event argument.
arg5	[in] Event argument.
arg6	[in] Event argument.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The process that raises the event.

Module

NEvent Unit [448]

7.1.2.19.2.7 N_EVENT_RAISE_RAW Macro

Raises the specified event with specified arguments.

C++

```
#define N_EVENT_RAISE_RAW(callbackType, pEvent, args) \
    {\
        NResult N_EVENT_RAISE_RAW_res = N_OK;\
        HNCallback * N_EVENT_RAISE_RAW_arhCallbacks;\
        NInt N_EVENT_RAISE_RAW_count;\
        N_CHECK(NEventGetCallbacks(pEvent, &N_EVENT_RAISE_RAW_arhCallbacks,
&N_EVENT_RAISE_RAW_count));\
        N_FOREACH(HNCallback, N_EVENT_RAISE_RAW_hCallback, N_EVENT_RAISE_RAW_arhCallbacks,
N_EVENT_RAISE_RAW_count)\
            N_EVENT_RAISE_RAW_res = NCallbackInvokeRaw(callbackType,
N_EVENT_RAISE_RAW_hCallback, args);\
            if (NFailed(N_EVENT_RAISE_RAW_res)) break;\
        N_FOREACH_END\
        N_CHECK(NEventReleaseCallbacks(pEvent, N_EVENT_RAISE_RAW_arhCallbacks,
N_EVENT_RAISE_RAW_count));\
        N_CHECK(N_EVENT_RAISE_RAW_res);\
    }
```

Parameters

Parameters	Description
callbackType	[in] The type of the callback.
pEvent	[in/out] Pointer to the event.
args	[in] Event arguments.

Returns

The process that raises the event.

Notes

Raw functions (i.e. NEventAddRaw [449]) represent the functions themselves.

Non-raw functions (i.e. NEventAdd [456]) represent macros that call those functions for the convenience of using them in C++.

Module

NEvent Unit [448]

7.1.2.19.2.8 N_EVENT_SIZE Macro

Defines the size of the event.

C++

```
#define N_EVENT_SIZE 8
```

Module

NEvent Unit [448]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.19.2.9 NEventAdd Macro

Adds a new event.

C++

```
#define NEventAdd(pEvent, pProc, pParam) NEventAddRaw(pEvent, (void *)(NSizeType)pProc,
pParam)
```

Parameters

Parameters	Description
pEvent	[in/out] Pointer to the event.
pProc	[in/out] Pointer to the process.
pParam	[in/out] Pointer to the parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

Raw functions (i.e. NEventAddRaw [449]) represent the functions themselves.

Non-raw functions (i.e. NEventAdd) represent macros that call those functions for the convenience of using them in C++.

Module

NEvent Unit [448]

7.1.2.19.2.10 NEventRemove Macro

Removes the specified event.

C++

```
#define NEventRemove(pEvent, pProc, pParam) NEventRemoveRaw(pEvent, (void
*)(NSizeType)pProc, pParam)
```

Parameters

Parameters	Description
pEvent	[in/out] Pointer to the event.
pProc	[in/out] Pointer to the process.
pParam	[in/out] Pointer to the parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

Raw functions (i.e. NEventAddRaw [449]) represent the functions themselves.

Non-raw functions (i.e. NEventAdd [456]) represent macros that call those functions for the convenience of using them in C++.

Module

NEvent Unit [448]

7.1.2.19.3 Files

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.19.3.1 NEvent.h

Header file for the NEvent [448] unit.

Functions

	Name	Description
	NEventAddN [449]	Adds a new event with the specified callback.
	NEventAddRaw [449]	Adds a new event.
	NEventDispose [449]	Disposes of the specified event.
	NEventGetCallbacks [450]	Retrieves all callbacks associated with the event.
	NEventInit [450]	Initializes the specified event.
	NEventIsEmpty [450]	
	NEventReleaseCallbacks [451]	Releases all callbacks associated with the event.
	NEventRemoveN [451]	Removes the specified event with the specified callback.
	NEventRemoveRaw [452]	Removes the specified event.

Macros

Name	Description
N_EVENT_RAISE_1 [452]	Raises the specified event with specified arguments.
N_EVENT_RAISE_2 [452]	Raises the specified event with specified arguments.
N_EVENT_RAISE_3 [453]	Raises the specified event with specified arguments.
N_EVENT_RAISE_4 [453]	Raises the specified event with specified arguments.
N_EVENT_RAISE_5 [454]	Raises the specified event with specified arguments.
N_EVENT_RAISE_6 [454]	Raises the specified event with specified arguments.
N_EVENT_RAISE_RAW [455]	Raises the specified event with specified arguments.
N_EVENT_SIZE [455]	Defines the size of the event.
NEventAdd [456]	Adds a new event.
NEventRemove [456]	Removes the specified event.

Module

NEvent Unit [448]

7.1.2.20 NExpandableObject Unit

Provides functionality for retrieving information about specified object.

Files

Name	Description
NExpandableObject.h [458]	Header file for the NExpandableObject module.

Functions

	Name	Description
	NExpandableObjectGetProperties [458]	Retrieves the properties of the specified expandable object.

7.1.2.20.1 Functions

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.20.1.1 NExpandableObjectGetProperties Function

Retrieves the properties of the specified expandable object.

C++

```
NResult N_API NExpandableObjectGetProperties(HNExpandableObject hObject, HNPropertyBag *
phValue);
```

Parameters

Parameters	Description
HNExpandableObject hObject	[in] Handle to the expandable object.
HNPropertyBag * phValue	[out] Pointer to a handle to the property bag of the expandable object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NExpandableObject Unit [457]

7.1.2.20.2 Files

7.1.2.20.2.1 NExpandableObject.h

Header file for the NExpandableObject [457] module.

Functions

	Name	Description
	NExpandableObjectGetProperties [458]	Retrieves the properties of the specified expandable object.

Module

NExpandableObject Unit [457]

7.1.2.21 NFile Unit

Provides static methods for the creation, copying, deletion, moving, and opening of files, and aids in the creation of FileStream objects.

Functions

	Name	Description
	NFileAppendAllLines [461]	Picks a Unicode (W) or ANSCII (A) version of the NFileAppendAllLines function.
	NFileAppendAllLinesPN [461]	Picks a Unicode (W) or ANSCII (A) version of the NFileAppendAllLinesPN function.
	NFileAppendAllLinesWithEncoding [461]	Picks a Unicode (W) or ANSCII (A) version of the NFileAppendAllLinesWithEncoding function.
	NFileAppendAllLinesWithEncodingPN [462]	Picks a Unicode (W) or ANSCII (A) version of the NFileAppendAllLinesWithEncodingPN function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NFileAppendAllText [462]	Opens a file, appends the specified string to the file, and then closes the file.
	NFileAppendAllTextPN [462]	Opens a file, appends the specified string to the file, and then closes the file.
	NFileAppendAllTextWithEncoding [463]	Opens a file, appends the specified string to the file, and then closes the file.
	NFileAppendAllTextWithEncodingPN [463]	Opens a file, appends the specified string to the file, and then closes the file.
	NFileAppendText [464]	Creates a StreamWriter that appends UTF-8 encoded text to an existing file.
	NFileCreate [464]	Creates a file in the specified path.
	NFileCreateText [464]	Creates or opens a file for writing UTF-8 encoded text.
	NFileCreateWithBufferSize [465]	Creates a file in the specified path with the specified buffer size.
	NFileDelete [465]	Deletes the specified file. An exception is not thrown if the specified file does not exist.
	NFileExists [465]	Determines whether the file with the specified path exists.
	NFileGetAttributes [466]	Gets the attributes of the file on the path.
	NFileOpen [466]	Opens a NFileStream [483] on the specified path with read/write access.
	NFileOpenRead [466]	Opens an existing file for reading.
	NFileOpenText [467]	Opens an existing UTF-8 encoded text file for reading.
	NFileOpenWithAccess [467]	Opens a NFileStream [483] on the specified path, with the specified mode and access.
	NFileOpenWithAccessAndShare [468]	Opens a NFileStream [483] on the specified path, having the specified mode with read, write, or read/write access and the specified sharing option.
	NFileOpenWrite [468]	Opens an existing file for writing.
	NFileReadAllBytesCN [468]	Opens a binary file, reads the contents of the file into a byte array, and then closes the file.
	NFileReadAllBytesN [469]	Opens a binary file, reads the contents of the file into a byte array, and then closes the file.
	NFileReadAllLinesCN [469]	Opens a text file, reads all lines of the file, and then closes the file.
	NFileReadAllLinesN [469]	Opens a text file, reads all lines of the file, and then closes the file.
	NFileReadAllLinesWithEncodingCN [470]	Opens a file, reads all lines of the file with the specified encoding, and then closes the file.
	NFileReadAllLinesWithEncodingN [470]	Opens a file, reads all lines of the file with the specified encoding, and then closes the file.
	NFileReadAllTextCN [471]	Opens a text file, reads all lines of the file, and then closes the file.
	NFileReadAllTextN [471]	Opens a text file, reads all lines of the file, and then closes the file.
	NFileReadAllTextWithEncodingCN [471]	Opens a file, reads all lines of the file with the specified encoding, and then closes the file.
	NFileReadAllTextWithEncodingN [472]	Opens a file, reads all lines of the file with the specified encoding, and then closes the file.
	NFileWriteAllBytes [472]	Creates a new file, writes the specified byte array to the file, and then closes the file.
	NFileWriteAllBytesCN [473]	Creates a new file, writes the specified buffer to the file, and then closes the file.
	NFileWriteAllBytesPN [473]	Creates a new file, writes the specified byte array to the file, and then closes the file.
	NFileWriteAllLines [473]	Creates a new file, write the specified char array to the file, and then closes the file.
	NFileWriteAllLinesPN [474]	Creates a new file, write the specified char array to the file, and then closes the file.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NFileWriteAllLinesWithEncoding [474]	Creates a new file, writes the specified char array to the file using the specified encoding, and then closes the file.
	NFileWriteAllLinesWithEncodingPN [475]	Creates a new file, writes the specified char array to the file using the specified encoding, and then closes the file.
	NFileWriteAllText [475]	Creates a new file, writes the specified char to the file, and then closes the file.
	NFileWriteAllTextPN [476]	Creates a new file, writes the specified string array to the file, and then closes the file.
	NFileWriteAllTextWithEncoding [476]	Creates a new file, writes the specified string array to the file using the specified encoding, and then closes the file.
	NFileWriteAllTextWithEncodingPN [476]	Creates a new file, writes the specified string array to the file using the specified encoding, and then closes the file.

Macros

Name	Description
NFileAppendAllText [477]	Picks a Unicode (W) or ASCII (A) version of the NFileAppendAllText function.
NFileAppendAllTextPN [477]	Picks a Unicode (W) or ASCII (A) version of the NFileAppendAllTextPN function.
NFileAppendAllTextWithEncoding [478]	Picks a Unicode (W) or ASCII (A) version of the NFileAppendAllTextWithEncoding function.
NFileAppendAllTextWithEncodingPN [478]	Picks a Unicode (W) or ASCII (A) version of the NFileAppendAllTextWithEncodingPN function.
NFileAppendText [478]	Picks a Unicode (W) or ASCII (A) version of the NFileAppendText function.
NFileCreate [478]	Picks a Unicode (W) or ASCII (A) version of the NFileCreate function.
NFileCreateText [478]	Picks a Unicode (W) or ASCII (A) version of the NFileAppendCreateText function.
NFileCreateWithBufferSize [478]	Picks a Unicode (W) or ASCII (A) version of the NFileAppendCreateWithBufferSize function.
NFileDelete [479]	Picks a Unicode (W) or ASCII (A) version of the NFileDelete function.
NFileExists [479]	Picks a Unicode (W) or ASCII (A) version of the NFileExists function.
NFileGetAttributes [479]	Picks a Unicode (W) or ASCII (A) version of the NFileGetAttributes function.
NFileOpen [479]	Picks a Unicode (W) or ASCII (A) version of the NFileOpen function.
NFileOpenRead [479]	Picks a Unicode (W) or ASCII (A) version of the NFileOpenRead function.
NFileOpenText [479]	Picks a Unicode (W) or ASCII (A) version of the NFileOpenText function.
NFileOpenWithAccess [480]	Picks a Unicode (W) or ASCII (A) version of the NFileOpenWithAccess function.
NFileOpenWithAccessAndShare [480]	Picks a Unicode (W) or ASCII (A) version of the NFileOpenWithAccessAndShare function.
NFileOpenWrite [480]	Picks a Unicode (W) or ASCII (A) version of the NFileOpenWrite function.
NFileReadAllBytesCN [480]	Picks a Unicode (W) or ASCII (A) version of the NFileReadAllBytesCN function.
NFileReadAllLinesCN [480]	Picks a Unicode (W) or ASCII (A) version of the NFileReadAllLinesCN function.
NFileReadAllLinesWithEncodingCN [481]	Picks a Unicode (W) or ASCII (A) version of the NFileReadAllLinesWithEncodingCN function.
NFileReadAllTextCN [481]	Picks a Unicode (W) or ASCII (A) version of the NFileReadAllTextCN function.
NFileReadAllTextWithEncodingCN [481]	Picks a Unicode (W) or ASCII (A) version of the NFileReadAllTextWithEncodingCN function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFileWriteAllBytes [481]	Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllBytes function.
NFileWriteAllBytesCN [481]	Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllBytesCN function.
NFileWriteAllLines [481]	Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllLines function.
NFileWriteAllLinesPN [482]	Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllLinesPN function.
NFileWriteAllLinesWithEncoding [482]	Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllLinesWithEncoding function.
NFileWriteAllLinesWithEncodingPN [482]	Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllLinesWithEncodingPN function.
NFileWriteAllText [482]	Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllText function.
NFileWriteAllTextPN [482]	Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllTextPN function.
NFileWriteAllTextWithEncoding [482]	Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllTextWithEncoding function.
NFileWriteAllTextWithEncodingPN [483]	Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllTextWithEncodingPN function.

Structs, Records, Enums

	Name	Description
	NFileAttributes_ [477]	Enumerates file attributes.
	NFileAttributes [477]	Enumerates file attributes.

7.1.2.21.1 Functions

7.1.2.21.1.1 NFileAppendAllLines Function

Picks a Unicode (W) or ANSCII (A) version of the NFileAppendAllLines function.

C++

```
NResult N_API NFileAppendAllLines(const NChar * szPath, const NChar * * arszContent, NInt
contentLength);
```

Module

NFile Unit [458]

7.1.2.21.1.2 NFileAppendAllLinesPN Function

Picks a Unicode (W) or ANSCII (A) version of the NFileAppendAllLinesPN function.

C++

```
NResult N_API NFileAppendAllLinesPN(HNString hPath, const NChar * * arszContent, NInt
contentLength);
```

Module

NFile Unit [458]

7.1.2.21.1.3 NFileAppendAllLinesWithEncoding Function

Picks a Unicode (W) or ANSCII (A) version of the NFileAppendAllLinesWithEncoding function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NFileAppendAllLinesWithEncoding(const NChar * szPath, const NChar * *
arszContent, NInt contentLength, NEncoding encoding);
```

Module

NFile Unit [458]

7.1.2.21.1.4 NFileAppendAllLinesWithEncodingPN Function

Picks a Unicode (W) or ANSCII (A) version of the NFileAppendAllLinesWithEncodingPN function.

C++

```
NResult N_API NFileAppendAllLinesWithEncodingPN(HNString hPath, const NChar * *
arszContent, NInt contentLength, NEncoding encoding);
```

Module

NFile Unit [458]

7.1.2.21.1.5 NFileAppendAllText Function

Opens a file, appends the specified string to the file, and then closes the file.

C++

```
NResult N_API NFileAppendAllText(const NChar * szPath, const NChar * szContent);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to append the specified string to.
const NChar * szContent	[in] Pointer to the string to append to the file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If the file does not exist, this method creates a file, writes the specified string to the file, then closes the file.

Module

NFile Unit [458]

7.1.2.21.1.6 NFileAppendAllTextPN Function

Opens a file, appends the specified string to the file, and then closes the file.

C++

```
NResult N_API NFileAppendAllTextPN(HNString hPath, const NChar * szContent);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path of the file to append the specified string to.
const NChar * szContent	[in] Pointer to the string to append to the file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

If the file does not exist, this method creates a file, writes the specified string to the file, then closes the file.

Module

NFile Unit [458]

7.1.2.21.1.7 NFileAppendAllTextWithEncoding Function

Opens a file, appends the specified string to the file, and then closes the file.

C++

```
NResult N_API NFileAppendAllTextWithEncoding(const NChar * szPath, const NChar * szContent,
NEncoding encoding);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to append the specified string to.
const NChar * szContent	[in] Pointer to the string to append to the file.
NEncoding encoding	[in] The encoding to be used.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If the file does not exist, this method creates a file, writes the specified string to the file, then closes the file.

Module

NFile Unit [458]

7.1.2.21.1.8 NFileAppendAllTextWithEncodingPN Function

Opens a file, appends the specified string to the file, and then closes the file.

C++

```
NResult N_API NFileAppendAllTextWithEncodingPN(HNString hPath, const NChar * szContent,
NEncoding encoding);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path of the file to append the specified string to.
const NChar * szContent	[in] Pointer to the string to append to the file.
NEncoding encoding	[in] The encoding to be used.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If the file does not exist, this method creates a file, writes the specified string to the file, then closes the file.

Module

NFile Unit [458]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.21.1.9 NFileAppendText Function

Creates a StreamWriter that appends UTF-8 encoded text to an existing file.

C++

```
NResult N_API NFileAppendText(const NChar * szPath, HNStreamWriter * phWriter);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to append the specified string to.
HNStreamWriter * phWriter	[out] Pointer to NStreamWriter [673] that appends UTF-8 encoded text to an existing file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.10 NFileCreate Function

Creates a file in the specified path.

C++

```
NResult N_API NFileCreate(const NChar * szPath, HNFileStream * phStream);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path and name of the file to create.
HNFileStream * phStream	[out] Pointer to a handle to a NFileStream [483] that provides read/write access to the file specified in *szPath*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.11 NFileCreateText Function

Creates or opens a file for writing UTF-8 encoded text.

C++

```
NResult N_API NFileCreateText(const NChar * szPath, HNStreamWriter * phWriter);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to path of the file to be opened for writing.
HNStreamWriter * phWriter	[out] Pointer to a handle to a NStreamWriter [673] that writes to the specified file using UTF-8 encoding.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NFile Unit [458]

7.1.2.21.1.12 NFileCreateWithBufferSize Function

Creates a file in the specified path with the specified buffer size.

C++

```
NResult N_API NFileCreateWithBufferSize(const NChar * szPath, NSizeType bufferSize,
HNFileStream * phStream);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path and name of the file to create.
NSizeType bufferSize	[in] The buffer size.
HNFileStream * phStream	[out] Pointer to a handle to a NFileStream [483] that provides read/write access to the file specified in *szPath*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.13 NFileDelete Function

Deletes the specified file. An exception is not thrown if the specified file does not exist.

C++

```
NResult N_API NFileDelete(const NChar * szPath);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the name of the file to be deleted.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.14 NFileExists Function

Determines whether the file with the specified path exists.

C++

```
NResult N_API NFileExists(const NChar * szPath, NBool * pValue);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the file path to check.
NBool * pValue	[out] NTrue [1064] if the file exists; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NFile Unit [458]

7.1.2.21.1.15 NFileGetAttributes Function

Gets the attributes of the file on the path.

C++

```
NResult N_API NFileGetAttributes(const NChar * szPath, NFileAttributes * pValue);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file.
NFileAttributes * pValue	[out] Pointer to the NFileAttributes [477] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.16 NFileOpen Function

Opens a NFileStream [483] on the specified path with read/write access.

C++

```
NResult N_API NFileOpen(const NChar * szPath, NFileMode mode, HNFileStream * phStream);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to be opened.
NFileMode mode	[in] A FileMode value that specifies whether a file is created if one does not exist, and determines whether the contents of existing files are retained or overwritten.
HNFileStream * phStream	[out] Pointer to a handle to NFileStream [483] opened in the specified mode and path, with read/write access and not shared.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.17 NFileOpenRead Function

Opens an existing file for reading.

C++

```
NResult N_API NFileOpenRead(const NChar * szPath, HNFileStream * phStream);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to be opened.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNFileStream * phStream	[out] Pointer to a handle to a read-only NFileStream [483] on the specified path.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.18 NFileOpenText Function

Opens an existing UTF-8 encoded text file for reading.

C++

```
NResult N_API NFileOpenText(const NChar * szPath, HNStreamReader * phReader);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to be opened.
HNStreamReader * phReader	[out] Pointer to a handle to a NStreamReader [668] on the specified path.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.19 NFileOpenWithAccess Function

Opens a NFileStream [483] on the specified path, with the specified mode and access.

C++

```
NResult N_API NFileOpenWithAccess(const NChar * szPath, NFileMode mode, NFileAccess access,
HNFileStream * phStream);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to be opened.
NFileMode mode	[in] A FileMode value that specifies whether a file is created if one does not exist, and determines whether the contents of existing files are retained or overwritten.
NFileAccess access	[in] A FileAccess value that specifies the operations that can be performed on the file.
HNFileStream * phStream	[out] Pointer to a handle to an unshared NFileStream [483] that provides access to the specified file, with the specified mode and access.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.21.1.20 NFileOpenWithAccessAndShare Function

Opens a NFileStream [483] on the specified path, having the specified mode with read, write, or read/write access and the specified sharing option.

C++

```
NResult N_API NFileOpenWithAccessAndShare(const NChar * szPath, NFileMode mode, NFileAccess access, NFileShare share, HNFileStream * phStream);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to be opened.
NFileMode mode	[in] A FileMode value that specifies whether a file is created if one does not exist, and determines whether the contents of existing files are retained or overwritten.
NFileAccess access	[in] A FileAccess value that specifies the operations that can be performed on the file.
NFileShare share	[in] A FileShare value specifying the type of access other threads have to the file.
HNFileStream * phStream	[out] Pointer to a handle to a NFileStream [483] on the specified path, having the specified mode with read, write, or read/write access and the specified sharing option.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.21 NFileOpenWrite Function

Opens an existing file for writing.

C++

```
NResult N_API NFileOpenWrite(const NChar * szPath, HNFileStream * phStream);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to be opened.
HNFileStream * phStream	[out] Pointer to a handle to an unshared NFileStream [483] object on the specified path with Write access.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.22 NFileReadAllBytesCN Function

Opens a binary file, reads the contents of the file into a byte array, and then closes the file.

C++

```
NResult N_API NFileReadAllBytesCN(const NChar * szPath, HNBuffer * phContent);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to be opened for reading.
HNBuffer * phContent	[out] Pointer to a handle to NBuffer [313] containing the read bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.23 NFileReadAllBytesN Function

Opens a binary file, reads the contents of the file into a byte array, and then closes the file.

C++

```
NResult N_API NFileReadAllBytesN(HNString hPath, HNBuffer * phContent);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path of the file to be opened for reading.
HNBuffer * phContent	[out] Pointer to a handle to NBuffer [313] containing the read bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.24 NFileReadAllLinesCN Function

Opens a text file, reads all lines of the file, and then closes the file.

C++

```
NResult N_API NFileReadAllLinesCN(const NChar * szPath, HNString * * parhContent, NInt *
pContentLength);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to be opened for reading.
HNString * * parhContent	[out] Pointer to an array of string handles containing all lines of the file.
NInt * pContentLength	[out] Pointer to the size of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.25 NFileReadAllLinesN Function

Opens a text file, reads all lines of the file, and then closes the file.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NFileReadAllLinesN(HNString hPath, HNString * * parhContent, NInt *
pContentLength);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path of the file to be opened for reading.
HNString * * parhContent	[out] Pointer to an array of string handles containing all lines of the file.
NInt * pContentLength	[out] Pointer to the size of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.26 NFileReadAllLinesWithEncodingCN Function

Opens a file, reads all lines of the file with the specified encoding, and then closes the file.

C++

```
NResult N_API NFileReadAllLinesWithEncodingCN(const NChar * szPath, NEncoding encoding,
NBool detectEncodingFromByteOrderMarks, HNString * * parhContent, NInt * pContentLength);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to be opened for reading.
NEncoding encoding	[in] The encoding applied to the contents of the file.
NBool detectEncodingFromByteOrderMarks	[in] NTrue [1064] if the encoding should be detected form byte order marks; else NFalse [1041].
HNString * * parhContent	[out] Pointer to an array of string handles containing all lines of the file.
NInt * pContentLength	[out] Pointer to the size of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.27 NFileReadAllLinesWithEncodingN Function

Opens a file, reads all lines of the file with the specified encoding, and then closes the file.

C++

```
NResult N_API NFileReadAllLinesWithEncodingN(HNString hPath, NEncoding encoding, NBool
detectEncodingFromByteOrderMarks, HNString * * parhContent, NInt * pContentLength);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path of the file to be opened for reading.
NEncoding encoding	[in] The encoding applied to the contents of the file.
NBool detectEncodingFromByteOrderMarks	[in] NTrue [1064] if the encoding should be detected form byte order marks; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNString * * parhContent	[out] Pointer to an array of string handles containing all lines of the file.
NInt * pContentLength	[out] Pointer to the size of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.28 NFileReadAllTextCN Function

Opens a text file, reads all lines of the file, and then closes the file.

C++

```
NResult N_API NFileReadAllTextCN(const NChar * szPath, HNString * phContent);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to be opened for reading.
HNString * phContent	[out] Pointer to a handle to NString [678] containing all lines of the file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.29 NFileReadAllTextN Function

Opens a text file, reads all lines of the file, and then closes the file.

C++

```
NResult N_API NFileReadAllTextN(HNString hPath, HNString * phContent);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path of the file to be opened for reading.
HNString * phContent	[out] Pointer to a handle to NString [678] containing all lines of the file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.30 NFileReadAllTextWithEncodingCN Function

Opens a file, reads all lines of the file with the specified encoding, and then closes the file.

C++

```
NResult N_API NFileReadAllTextWithEncodingCN(const NChar * szPath, NEncoding encoding,
NBool detectEncodingFromByteOrderMarks, HNString * phContent);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to be opened for reading.
NEncoding encoding	[in] The encoding applied to the contents of the file.
NBool detectEncodingFromByteOrderMarks	[in] NTrue [1064] if the encoding should be detected form byte order marks; else NFalse [1041].
HNString * phContent	[out] Pointer to a handle to NString [678] containing all lines of the file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.31 NFileReadAllTextWithEncodingN Function

Opens a file, reads all lines of the file with the specified encoding, and then closes the file.

C++

```
NResult N_API NFileReadAllTextWithEncodingN(HNString hPath, NEncoding encoding, NBool
detectEncodingFromByteOrderMarks, HNString * phContent);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path of the file to be opened for reading.
NEncoding encoding	[in] The encoding applied to the contents of the file.
NBool detectEncodingFromByteOrderMarks	[in] NTrue [1064] if the encoding should be detected form byte order marks; else NFalse [1041].
HNString * phContent	[out] Pointer to a handle to NString [678] containing all lines of the file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFile Unit [458]

7.1.2.21.1.32 NFileWriteAllBytes Function

Creates a new file, writes the specified byte array to the file, and then closes the file.

C++

```
NResult N_API NFileWriteAllBytes(const NChar * szPath, const void * pContent, NSizeType
contentSize);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to write to.
const void * pContent	[in] Pointer to a byte array to write the bytes from.
NSizeType contentSize	[in] Size of the byte array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Notes

If the target file already exists, it is overwritten.

Module

NFile Unit [458]

7.1.2.21.1.33 NFileWriteAllBytesCN Function

Creates a new file, writes the specified buffer to the file, and then closes the file.

C++

```
NResult N_API NFileWriteAllBytesCN(const NChar * szPath, HNBuffer hContent);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to write to.
HNBuffer hContent	[in] Handle to the buffer to write the bytes from.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If the target file already exists, it is overwritten.

Module

NFile Unit [458]

7.1.2.21.1.34 NFileWriteAllBytesPN Function

Creates a new file, writes the specified byte array to the file, and then closes the file.

C++

```
NResult N_API NFileWriteAllBytesPN(HNString hPath, const void * pContent, NSizeType
contentSize);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path of the file to write to.
const void * pContent	[in] Pointer to a byte array to write the bytes from.
NSizeType contentSize	[in] Size of the byte array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If the target file already exists, it is overwritten.

Module

NFile Unit [458]

7.1.2.21.1.35 NFileWriteAllLines Function

Creates a new file, write the specified char array to the file, and then closes the file.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NFileWriteAllLines(const NChar * szPath, const NChar * * arszContent, NInt contentLength);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to write to.
const NChar * * arszContent	[in] Pointer to the char array to write to the file.
NInt contentLength	[in] Size of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If the target file already exists, it is overwritten.

Module

NFile Unit [458]

7.1.2.21.1.36 NFileWriteAllLinesPN Function

Creates a new file, write the specified char array to the file, and then closes the file.

C++

```
NResult N_API NFileWriteAllLinesPN(HNString hPath, const NChar * * arszContent, NInt contentLength);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path of the file to write to.
const NChar * * arszContent	[in] Pointer to the char array to write to the file.
NInt contentLength	[in] Size of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If the target file already exists, it is overwritten.

Module

NFile Unit [458]

7.1.2.21.1.37 NFileWriteAllLinesWithEncoding Function

Creates a new file, writes the specified char array to the file using the specified encoding, and then closes the file.

C++

```
NResult N_API NFileWriteAllLinesWithEncoding(const NChar * szPath, const NChar * * arszContent, NInt contentLength, NEncoding encoding);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to write to.
const NChar * * arszContent	[in] Pointer to the char array to write to the file.
NInt contentLength	[in] Size of the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NEncoding encoding	[in] Encoding applied to the characters written into the file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If the target file already exists, it is overwritten.

Module

NFile Unit [458]

7.1.2.21.1.38 NFileWriteAllLinesWithEncodingPN Function

Creates a new file, writes the specified char array to the file using the specified encoding, and then closes the file.

C++

```
NResult N_API NFileWriteAllLinesWithEncodingPN(HNString hPath, const NChar * * arszContent,
NInt contentLength, NEncoding encoding);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path of the file to write to.
const NChar * * arszContent	[in] Pointer to the char array to write to the file.
NInt contentLength	[in] Size of the array.
NEncoding encoding	[in] Encoding applied to the characters written into the file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If the target file already exists, it is overwritten.

Module

NFile Unit [458]

7.1.2.21.1.39 NFileWriteAllText Function

Creates a new file, writes the specified char to the file, and then closes the file.

C++

```
NResult N_API NFileWriteAllText(const NChar * szPath, const NChar * szContent);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to write to.
const NChar * szContent	[in] Pointer to char to write to the file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If the target file already exists, it is overwritten.

Module

NFile Unit [458]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.21.1.40 NFileWriteAllTextPN Function

Creates a new file, writes the specified string array to the file, and then closes the file.

C++

```
NResult N_API NFileWriteAllTextPN(HNString hPath, const NChar * szContent);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path of the file to write to.
const NChar * szContent	[in] Pointer to char to write to the file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If the target file already exists, it is overwritten.

Module

NFile Unit [458]

7.1.2.21.1.41 NFileWriteAllTextWithEncoding Function

Creates a new file, writes the specified string array to the file using the specified encoding, and then closes the file.

C++

```
NResult N_API NFileWriteAllTextWithEncoding(const NChar * szPath, const NChar * szContent,
NEncoding encoding);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of the file to write to.
const NChar * szContent	[in] Pointer to char to write to the file.
NEncoding encoding	[in] Encoding applied to the characters written into the file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If the target file already exists, it is overwritten.

Module

NFile Unit [458]

7.1.2.21.1.42 NFileWriteAllTextWithEncodingPN Function

Creates a new file, writes the specified string array to the file using the specified encoding, and then closes the file.

C++

```
NResult N_API NFileWriteAllTextWithEncodingPN(HNString hPath, const NChar * szContent,
NEncoding encoding);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path of the file to write to.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const NChar * szContent	[in] Pointer to char to write to the file.
NEncoding encoding	[in] Encoding applied to the characters written into the file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If the target file already exists, it is overwritten.

Module

NFile Unit [458]

7.1.2.21.2 Structs, Records, Enums

7.1.2.21.2.1 NFileAttributes Enumeration

Enumerates file attributes.

C++

```
typedef enum NFileAttributes_ {
  nfatNone = 0,
  nfatDirectory = 1
} NFileAttributes;
```

Members

Members	Description
nfatNone = 0	None.
nfatDirectory = 1	Directory.

Module

NFile Unit [458]

7.1.2.21.3 Macros

7.1.2.21.3.1 NFileAppendAllText Macro

Picks a Unicode (W) or ASCII (A) version of the NFileAppendAllText function.

C++

```
#define NFileAppendAllText N_FUNC_AW(NFileAppendAllText)
```

Module

NFile Unit [458]

7.1.2.21.3.2 NFileAppendAllTextPN Macro

Picks a Unicode (W) or ASCII (A) version of the NFileAppendAllTextPN function.

C++

```
#define NFileAppendAllTextPN N_FUNC_AW(NFileAppendAllTextPN)
```

Module

NFile Unit [458]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.21.3.3 NFileAppendAllTextWithEncoding Macro

Picks a Unicode (W) or ASCII (A) version of the NFileAppendAllTextWithEncoding function.

C++

```
#define NFileAppendAllTextWithEncoding N_FUNC_AW(NFileAppendAllTextWithEncoding)
```

Module

NFile Unit [458]

7.1.2.21.3.4 NFileAppendAllTextWithEncodingPN Macro

Picks a Unicode (W) or ASCII (A) version of the NFileAppendAllTextWithEncodingPN function.

C++

```
#define NFileAppendAllTextWithEncodingPN N_FUNC_AW(NFileAppendAllTextWithEncodingPN)
```

Module

NFile Unit [458]

7.1.2.21.3.5 NFileAppendText Macro

Picks a Unicode (W) or ASCII (A) version of the NFileAppendText function.

C++

```
#define NFileAppendText N_FUNC_AW(NFileAppendText)
```

Module

NFile Unit [458]

7.1.2.21.3.6 NFileCreate Macro

Picks a Unicode (W) or ASCII (A) version of the NFileCreate function.

C++

```
#define NFileCreate N_FUNC_AW(NFileCreate)
```

Module

NFile Unit [458]

7.1.2.21.3.7 NFileCreateText Macro

Picks a Unicode (W) or ASCII (A) version of the NFileAppendCreateText function.

C++

```
#define NFileCreateText N_FUNC_AW(NFileCreateText)
```

Module

NFile Unit [458]

7.1.2.21.3.8 NFileCreateWithBufferSize Macro

Picks a Unicode (W) or ASCII (A) version of the NFileAppendCreateWithBufferSize function.

C++

```
#define NFileCreateWithBufferSize N_FUNC_AW(NFileCreateWithBufferSize)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NFile Unit [458]

7.1.2.21.3.9 NFileDelete Macro

Picks a Unicode (W) or ASCII (A) version of the NFileDelete function.

C++

```
#define NFileDelete N_FUNC_AW(NFileDelete)
```

Module

NFile Unit [458]

7.1.2.21.3.10 NFileExists Macro

Picks a Unicode (W) or ASCII (A) version of the NFileExists function.

C++

```
#define NFileExists N_FUNC_AW(NFileExists)
```

Module

NFile Unit [458]

7.1.2.21.3.11 NFileGetAttributes Macro

Picks a Unicode (W) or ASCII (A) version of the NFileGetAttributes function.

C++

```
#define NFileGetAttributes N_FUNC_AW(NFileGetAttributes)
```

Module

NFile Unit [458]

7.1.2.21.3.12 NFileOpen Macro

Picks a Unicode (W) or ASCII (A) version of the NFileOpen function.

C++

```
#define NFileOpen N_FUNC_AW(NFileOpen)
```

Module

NFile Unit [458]

7.1.2.21.3.13 NFileOpenRead Macro

Picks a Unicode (W) or ASCII (A) version of the NFileOpenRead function.

C++

```
#define NFileOpenRead N_FUNC_AW(NFileOpenRead)
```

Module

NFile Unit [458]

7.1.2.21.3.14 NFileOpenText Macro

Picks a Unicode (W) or ASCII (A) version of the NFileOpenText function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NFileOpenText N_FUNC_AW(NFileOpenText)
```

Module

NFile Unit [458]

7.1.2.21.3.15 NFileOpenWithAccess Macro

Picks a Unicode (W) or ASCII (A) version of the NFileOpenWithAccess function.

C++

```
#define NFileOpenWithAccess N_FUNC_AW(NFileOpenWithAccess)
```

Module

NFile Unit [458]

7.1.2.21.3.16 NFileOpenWithAccessAndShare Macro

Picks a Unicode (W) or ASCII (A) version of the NFileOpenWithAccessAndShare function.

C++

```
#define NFileOpenWithAccessAndShare N_FUNC_AW(NFileOpenWithAccessAndShare)
```

Module

NFile Unit [458]

7.1.2.21.3.17 NFileOpenWrite Macro

Picks a Unicode (W) or ASCII (A) version of the NFileOpenWrite function.

C++

```
#define NFileOpenWrite N_FUNC_AW(NFileOpenWrite)
```

Module

NFile Unit [458]

7.1.2.21.3.18 NFileReadAllBytesCN Macro

Picks a Unicode (W) or ASCII (A) version of the NFileReadAllBytesCN function.

C++

```
#define NFileReadAllBytesCN N_FUNC_AW(NFileReadAllBytesCN)
```

Module

NFile Unit [458]

7.1.2.21.3.19 NFileReadAllLinesCN Macro

Picks a Unicode (W) or ASCII (A) version of the NFileReadAllLinesCN function.

C++

```
#define NFileReadAllLinesCN N_FUNC_AW(NFileReadAllLinesCN)
```

Module

NFile Unit [458]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.21.3.20 NFileReadAllLinesWithEncodingCN Macro

Picks a Unicode (W) or ASCII (A) version of the NFileReadAllLinesWithEncodingCN function.

C++

```
#define NFileReadAllLinesWithEncodingCN N_FUNC_AW(NFileReadAllLinesWithEncodingCN)
```

Module

NFile Unit [458]

7.1.2.21.3.21 NFileReadAllTextCN Macro

Picks a Unicode (W) or ASCII (A) version of the NFileReadAllTextCN function.

C++

```
#define NFileReadAllTextCN N_FUNC_AW(NFileReadAllTextCN)
```

Module

NFile Unit [458]

7.1.2.21.3.22 NFileReadAllTextWithEncodingCN Macro

Picks a Unicode (W) or ASCII (A) version of the NFileReadAllTextWithEncodingCN function.

C++

```
#define NFileReadAllTextWithEncodingCN N_FUNC_AW(NFileReadAllTextWithEncodingCN)
```

Module

NFile Unit [458]

7.1.2.21.3.23 NFileWriteAllBytes Macro

Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllBytes function.

C++

```
#define NFileWriteAllBytes N_FUNC_AW(NFileWriteAllBytes)
```

Module

NFile Unit [458]

7.1.2.21.3.24 NFileWriteAllBytesCN Macro

Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllBytesCN function.

C++

```
#define NFileWriteAllBytesCN N_FUNC_AW(NFileWriteAllBytesCN)
```

Module

NFile Unit [458]

7.1.2.21.3.25 NFileWriteAllLines Macro

Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllLines function.

C++

```
#define NFileWriteAllLines N_FUNC_AW(NFileWriteAllLines)
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NFile Unit [458]

7.1.2.21.3.26 NFileWriteAllLinesPN Macro

Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllLinesPN function.

C++

```
#define NFileWriteAllLinesPN N_FUNC_AW(NFileWriteAllLinesPN)
```

Module

NFile Unit [458]

7.1.2.21.3.27 NFileWriteAllLinesWithEncoding Macro

Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllLinesWithEncoding function.

C++

```
#define NFileWriteAllLinesWithEncoding N_FUNC_AW(NFileWriteAllLinesWithEncoding)
```

Module

NFile Unit [458]

7.1.2.21.3.28 NFileWriteAllLinesWithEncodingPN Macro

Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllLinesWithEncodingPN function.

C++

```
#define NFileWriteAllLinesWithEncodingPN N_FUNC_AW(NFileWriteAllLinesWithEncodingPN)
```

Module

NFile Unit [458]

7.1.2.21.3.29 NFileWriteAllText Macro

Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllText function.

C++

```
#define NFileWriteAllText N_FUNC_AW(NFileWriteAllText)
```

Module

NFile Unit [458]

7.1.2.21.3.30 NFileWriteAllTextPN Macro

Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllTextPN function.

C++

```
#define NFileWriteAllTextPN N_FUNC_AW(NFileWriteAllTextPN)
```

Module

NFile Unit [458]

7.1.2.21.3.31 NFileWriteAllTextWithEncoding Macro

Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllTextWithEncoding function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NFileWriteAllTextWithEncoding N_FUNC_AW(NFileWriteAllTextWithEncoding)
```

Module

NFile Unit [458]

7.1.2.21.3.32 NFileWriteAllTextWithEncodingPN Macro

Picks a Unicode (W) or ASCII (A) version of the NFileWriteAllTextWithEncodingPN function.

C++

```
#define NFileWriteAllTextWithEncodingPN N_FUNC_AW(NFileWriteAllTextWithEncodingPN)
```

Module

NFile Unit [458]

7.1.2.22 NFileStream Unit

Provides functions supporting both synchronous and asynchronous file read and write operations.

Functions

	Name	Description
	NFileStreamCreate [484]	Creates file stream with the specified path and creation mode.
	NFileStreamCreateFromOSHandle [484]	Creates a new NFileStream for the specified file handle, with the specified read/write permission.
	NFileStreamCreateFromOSHandleWithBufferSize [484]	Creates a new NFileStream for the specified file handle, with the specified read/write permission, and buffer size.
	NFileStreamCreateWithAccess [485]	Creates a new NFileStream with the specified path, creation mode, and read/write permission.
	NFileStreamCreateWithAccessAndShare [485]	Creates a new NFileStream with the specified path, creation mode, read/write permission, and sharing permission.
	NFileStreamCreateWithAccessShareAndBufferSize [486]	Creates a new NFileStream with the specified path, creation mode, read/write and sharing permission, the access other NFileStreams can have to the same file, and the buffer size.
	NFileStreamGetOSHandle [486]	Gets OS handle for the specified file.

Macros

Name	Description
NFileStreamCreate [487]	Picks a Unicode (W) or ASCII (A) version of the NFileStreamCreate function.
NFileStreamCreateWithAccess [487]	Picks a Unicode (W) or ASCII (A) version of the NFileStreamCreateWithAccess function.
NFileStreamCreateWithAccessAndShare [487]	Picks a Unicode (W) or ASCII (A) version of the NFileStreamCreateWithAccessAndShare function.
NFileStreamCreateWithAccessShareAndBufferSize [487]	Picks a Unicode (W) or ASCII (A) version of the NFileStreamCreateWithAccessShareAndBufferSize function.

7.1.2.22.1 Functions

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.22.1.1 NFileStreamCreate Function

Creates file stream with the specified path and creation mode.

C++

```
NResult N_API NFileStreamCreate(const NChar * szPath, NFileMode mode, HNFileStream *
phStream);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Zero terminated string containing path for the file that the current NFileStream [483] will encapsulate.
NFileMode mode	[in] File mode determining how to open or create the file.
HNFileStream * phStream	[out] A pointer to the created NFileStream [483].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFileStream Unit [483]

7.1.2.22.1.2 NFileStreamCreateFromOSHandle Function

Creates a new NFileStream [483] for the specified file handle, with the specified read/write permission.

C++

```
NResult N_API NFileStreamCreateFromOSHandle(NHandle handle, NBool ownsHandle, NFileAccess
access, HNFileStream * phStream);
```

Parameters

Parameters	Description
NHandle handle	[in] A file handle for the file that the current NFileStream [483] will encapsulate.
NBool ownsHandle	[in] NTrue [1064] if the file handle will be owned by this NFileStream [483]; otherwise, NFalse [1041].
NFileAccess access	[in] File access value determing how the file can be accessed by the NFileStream [483].
HNFileStream * phStream	[out] A pointer to the created NFileStream [483].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFileStream Unit [483]

7.1.2.22.1.3 NFileStreamCreateFromOSHandleWithBufferSize Function

Creates a new NFileStream [483] for the specified file handle, with the specified read/write permission, and buffer size.

C++

```
NResult N_API NFileStreamCreateFromOSHandleWithBufferSize(NHandle handle, NBool ownsHandle,
NFileAccess access, NSizeType bufferSize, HNFileStream * phStream);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NHandle handle	[in] A file handle for the file that the current NFileStream [483] will encapsulate.
NBool ownsHandle	[in] NTrue [1064] if the file handle will be owned by this NFileStream [483]; otherwise, NFalse [1041].
NFileAccess access	[in] File access value determining how the file can be accessed by the NFileStream [483].
NSizeType bufferSize	[in] A positive NInt32 [1006] value greater than 0 indicating the buffer size.
HNFileStream * phStream	[out] A pointer to the created NFileStream [483].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFileStream Unit [483]

7.1.2.22.1.4 NFileStreamCreateWithAccess Function

Creates a new NFileStream [483] with the specified path, creation mode, and read/write permission.

C++

```
NResult N_API NFileStreamCreateWithAccess(const NChar * szPath, NFileMode mode, NFileAccess
access, HNFileStream * phStream);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Zero terminated string containing path for the file that the current NFileStream [483] will encapsulate.
NFileMode mode	[in] File mode determining how to open or create the file.
NFileAccess access	[in] File access value determining how the file can be accessed by the NFileStream [483]. This gets the NStreamCanRead [661] and NStreamCanWrite [662] properties of the NFileStream [483] object. NStreamCanSeek [661] is NTrue [1064] if *szPath* specifies a disk file.
HNFileStream * phStream	[out] A pointer to the created NFileStream [483].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFileStream Unit [483]

7.1.2.22.1.5 NFileStreamCreateWithAccessAndShare Function

Creates a new NFileStream [483] with the specified path, creation mode, read/write permission, and sharing permission.

C++

```
NResult N_API NFileStreamCreateWithAccessAndShare(const NChar * szPath, NFileMode mode,
NFileAccess access, NFileShare share, HNFileStream * phStream);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const NChar * szPath	[in] Zero terminated string containing path for the file that the current NFileStream [483] will encapsulate.
NFileMode mode	[in] File mode determining how to open or create the file.
NFileAccess access	[in] File access value determining how the file can be accessed by the NFileStream [483].
NFileShare share	[in] Value determining how the file will be shared by processes.
HNFileStream * phStream	[out] A pointer to the created NFileStream [483].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFileStream Unit [483]

7.1.2.22.1.6 NFileStreamCreateWithAccessShareAndBufferSize Function

Creates a new NFileStream [483] with the specified path, creation mode, read/write and sharing permission, the access other NFileStreams can have to the same file, and the buffer size.

C++

```
NResult N_API NFileStreamCreateWithAccessShareAndBufferSize(const NChar * szPath, NFileMode
mode, NFileAccess access, NFileShare share, NSizeType bufferSize, HNFileStream * phStream);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Zero terminated string containing path for the file that the current NFileStream [483] will encapsulate.
NFileMode mode	[in] File mode determining how to open or create the file.
NFileAccess access	[in] File access value determining how the file can be accessed by the NFileStream [483]. This gets the NStreamCanRead [661] and NStreamCanWrite [662] properties of the NFileStream [483] object. NStreamCanSeek [661] is NTrue [1064] if *szPath* specifies a disk file.
NFileShare share	[in] A constant that determines how the file will be shared by processes.
NSizeType bufferSize	[in] A positive NInt32 [1006] value greater than 0 indicating the buffer size.
HNFileStream * phStream	[out] A pointer to the created NFileStream [483].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFileStream Unit [483]

7.1.2.22.1.7 NFileStreamGetOSHandle Function

Gets OS handle for the specified file.

C++

```
NResult N_API NFileStreamGetOSHandle(HNFileStream hFileStream, NHandle * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNFileStream hFileStream	[in] A handle to NFileStream [483].
NHandle * phValue	[out] OS handle to a file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NFileStream Unit [483]

7.1.2.22.2 Macros

7.1.2.22.2.1 NFileStreamCreate Macro

Picks a Unicode (W) or ASCII (A) version of the NFileStreamCreate function.

C++

```
#define NFileStreamCreate N_FUNC_AW(NFileStreamCreate)
```

Module

NFileStream Unit [483]

7.1.2.22.2.2 NFileStreamCreateWithAccess Macro

Picks a Unicode (W) or ASCII (A) version of the NFileStreamCreateWithAccess function.

C++

```
#define NFileStreamCreateWithAccess N_FUNC_AW(NFileStreamCreateWithAccess)
```

Module

NFileStream Unit [483]

7.1.2.22.2.3 NFileStreamCreateWithAccessAndShare Macro

Picks a Unicode (W) or ASCII (A) version of the NFileStreamCreateWithAccessAndShare function.

C++

```
#define NFileStreamCreateWithAccessAndShare N_FUNC_AW(NFileStreamCreateWithAccessAndShare)
```

Module

NFileStream Unit [483]

7.1.2.22.2.4 NFileStreamCreateWithAccessShareAndBufferSize Macro

Picks a Unicode (W) or ASCII (A) version of the NFileStreamCreateWithAccessShareAndBufferSize function.

C++

```
#define NFileStreamCreateWithAccessShareAndBufferSize
N_FUNC_AW(NFileStreamCreateWithAccessShareAndBufferSize)
```

Module

NFileStream Unit [483]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.23 NInterlocked Unit

Provides atomic operations for variables that are shared by multiple threads.

Functions

	Name	Description
	NInterlockedAdd [488]	Adds two integers and replaces the first integer with the sum, as an atomic operation.
	NInterlockedAnd [489]	Performs an atomic AND operation on the specified NInt [1006] values.
	NInterlockedCompareExchange [489]	Compares two integers for equality and, if they are equal, replaces one of the values.
	NInterlockedCompareExchangePointer [489]	Compares two integers for equality and, if they are equal, replaces one of the values.
	NInterlockedDecrement [490]	Decrements a specified variable and stores the result, as an atomic operation.
	NInterlockedExchange [490]	Sets an integer to a specified value and returns the original value, as an atomic operation.
	NInterlockedExchangeAdd [490]	Performs an atomic addition of two NInt [1006] values.
	NInterlockedExchangePointer [491]	Sets an integer to a specified value and returns the original value, as an atomic operation.
	NInterlockedIncrement [491]	Increments a specified variable and stores the result, as an atomic operation.
	NInterlockedOr [491]	Performs an atomic OR operation on the specified NInt [1006] values.
	NInterlockedXor [492]	Performs an atomic XOR operation on the specified NInt [1006] values.

Macros

Name	Description
N_SLOW_INTERLOCKED [492]	Determines whether the current platform has built-in interlocked functions. If the platform has inbuilt interlocked functions, it will use macros, which will directly map to the platform's interlocked functions. If the platform has no inbuilt interlocked functions, it will use the functions from the NInterlocked.

7.1.2.23.1 Functions

7.1.2.23.1.1 NInterlockedAdd Function

Adds two integers and replaces the first integer with the sum, as an atomic operation.

C++

```
NInt NInterlockedAdd(NInt volatile * pDst, NInt val);
```

Parameters

Parameters	Description
NInt volatile * pDst	[in/out] Volatile pointer to a variable containing the first value to be added. The sum of the two values is stored in *pDst*.
NInt val	[in] The value to be added to the integer at *pDst*.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The new value stored at *pDst*.

Module

NInterlocked Unit [488]

7.1.2.23.1.2 NInterlockedAnd Function

Performs an atomic AND operation on the specified NInt [1006] values.

C++

```
NInt NInterlockedAnd(NInt volatile * pDst, NInt val);
```

Parameters

Parameters	Description
NInt volatile * pDst	[in/out] A volatile pointer to the first operand. This value will be replaced with the result of the operation.
NInt val	[in] The second operand.

Returns

the original value of the *pDst* parameter.

Module

NInterlocked Unit [488]

7.1.2.23.1.3 NInterlockedCompareExchange Function

Compares two integers for equality and, if they are equal, replaces one of the values.

C++

```
NInt NInterlockedCompareExchange(NInt volatile * pDst, NInt exch, NInt comp);
```

Parameters

Parameters	Description
NInt volatile * pDst	[in/out] Volatile pointer to the destination NInt [1006] , whose value is compared with *comp* and possibly replaced.
NInt exch	[in] The value that replaces the destination value if the comparison results in equality.
NInt comp	[in] The value that is compared to the value at *pDst*.

Returns

The original value in *pDst*.

Module

NInterlocked Unit [488]

7.1.2.23.1.4 NInterlockedCompareExchangePointer Function

Compares two integers for equality and, if they are equal, replaces one of the values.

C++

```
void * NInterlockedCompareExchangePointer(void * volatile * pDst, void * exch, void * comp);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
void * volatile * pDst	[in/out] Volatile double pointer to the destination NInt [1006] , whose value is compared with *comp* and possibly replaced.
void * exch	[out] Pointer to the value that replaces the destination value if the comparison results in equality.
void * comp	[out] Pointer to the value that is compared to the value at *pDst*.

Returns

The original value in *pDst*.

Module

NInterlocked Unit [488]

7.1.2.23.1.5 NInterlockedDecrement Function

Decrements a specified variable and stores the result, as an atomic operation.

C++

```
NInt NInterlockedDecrement(NInt volatile * pDst);
```

Parameters

Parameters	Description
NInt volatile * pDst	[in/out] The variable whose value is to be decremented.

Returns

The decremented value.

Module

NInterlocked Unit [488]

7.1.2.23.1.6 NInterlockedExchange Function

Sets an integer to a specified value and returns the original value, as an atomic operation.

C++

```
NInt NInterlockedExchange(NInt volatile * pDst, NInt exch);
```

Parameters

Parameters	Description
NInt volatile * pDst	[in/out] Volatile pointer to the variable to set to the specified value.
NInt exch	[in] The value to which the *pDst* parameter is set.

Returns

The original value of *pDst*.

Module

NInterlocked Unit [488]

7.1.2.23.1.7 NInterlockedExchangeAdd Function

Performs an atomic addition of two NInt [1006] values.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NInt NInterlockedExchangeAdd(NInt volatile * pDst, NInt val);
```

Parameters

Parameters	Description
NInt volatile * pDst	[in/out] A volatile pointer to a variable. The value of this variable will be replaced with the result of the operation.
NInt val	[in] The value to be added to the variable pointed to by the *pDst* parameter.

Returns

The initial value of *pDst*.

Module

NInterlocked Unit [488]

7.1.2.23.1.8 NInterlockedExchangePointer Function

Sets an integer to a specified value and returns the original value, as an atomic operation.

C++

```
void * NInterlockedExchangePointer(void * volatile * pDst, void * exch);
```

Parameters

Parameters	Description
void * volatile * pDst	[out] Volatile double pointer to the variable to set to the specified value.
void * exch	[out] Pointer to the value to which the *pDst* parameter is set.

Returns

Pointer to the original value of *pDst*.

Module

NInterlocked Unit [488]

7.1.2.23.1.9 NInterlockedIncrement Function

Increments a specified variable and stores the result, as an atomic operation.

C++

```
NInt NInterlockedIncrement(NInt volatile * pDst);
```

Parameters

Parameters	Description
NInt volatile * pDst	[in/out] Volatile pointer to the variable whose value is to be incremented.

Returns

The incremented value.

Module

NInterlocked Unit [488]

7.1.2.23.1.10 NInterlockedOr Function

Performs an atomic OR operation on the specified NInt [1006] values.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NInt NInterlockedOr(NInt volatile * pDst, NInt val);
```

Parameters

Parameters	Description
NInt volatile * pDst	[in/out] A volatile pointer to the first operand. This value will be replaced with the result of the operation.
NInt val	[in] The second operand.

Returns

The original value of the *pDst* parameter.

Remarks

The function prevents more than one thread from using the same variable simultaneously.

Module

NInterlocked Unit [488]

7.1.2.23.1.11 NInterlockedXor Function

Performs an atomic XOR operation on the specified NInt [1006] values.

C++

```
NInt NInterlockedXor(NInt volatile * pDst, NInt val);
```

Parameters

Parameters	Description
NInt volatile * pDst	[in/out] A pointer to the first operand. This value will be replaced with the result of the operation.
NInt val	[in] The second operand.

Returns

The original value of the *pDst* parameter.

Remarks

The function prevents more than one thread from using the same variable simultaneously.

Module

NInterlocked Unit [488]

7.1.2.23.2 Macros

7.1.2.23.2.1 N_SLOW_INTERLOCKED Macro

Determines whether the current platform has built-in interlocked functions.

If the platform has inbuilt interlocked functions, it will use macros, which will directly map to the platform's interlocked functions.

If the platform has no inbuilt interlocked functions, it will use the functions from the NInterlocked [488].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_SLOW_INTERLOCKED
```

Module

NInterlocked Unit [488]

7.1.2.24 NIOTypes Unit

Defines input/output types used in Neurotechnology components.

Files

Name	Description
NIOTypes.h [498]	Header file for NIOTypes Unit.

Functions

	Name	Description
	NByteOrderGetSystem [493]	Gets system byte order.
	NByteOrderIsReverse [494]	Checks if specified byte order is reverse.
	NByteOrderIsValid [494]	Checks whether enumeration value is valid.
	NFileAccessIsValid [494]	Checks whether enumeration value is valid.
	NFileModeIsValid [495]	Checks whether enumeration value is valid.
	NFileShareIsValid [495]	Checks whether enumeration value is valid.
	NSeekOriginIsValid [495]	Checks whether enumeration value is valid.

Structs, Records, Enums

	Name	Description
	NByteOrder_ [496]	Specifies byte order.
	NFileAccess_ [496]	Specifies access to a file.
	NFileMode_ [496]	Specifies how the operating system should open a file.
	NFileShare_ [497]	Specifies values for controlling the kind of access to a file.
	NSeekOrigin_ [497]	Specifies values that represent reference points in streams for seeking.
	NByteOrder [496]	Specifies byte order.
	NFileAccess [496]	Specifies access to a file.
	NFileMode [496]	Specifies how the operating system should open a file.
	NFileShare [497]	Specifies values for controlling the kind of access to a file.
	NSeekOrigin [497]	Specifies values that represent reference points in streams for seeking.

7.1.2.24.1 Functions

7.1.2.24.1.1 NByteOrderGetSystem Function

Gets system byte order.

C++

```
NByteOrder N_API NByteOrderGetSystem();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

See Also

NByteOrder [496]

Module

NIOTypes Unit [493]

7.1.2.24.1.2 NByteOrderIsReverse Function

Checks if specified byte order is reverse.

C++

```
NBool N_API NByteOrderIsReverse(NByteOrder value);
```

Parameters

Parameters	Description
NByteOrder value	[in] NByteOrder [496] to check.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NIOTypes Unit [493]

7.1.2.24.1.3 NByteOrderIsValid Function

Checks whether enumeration value is valid.

C++

```
NBool N_API NByteOrderIsValid(NByteOrder value);
```

Parameters

Parameters	Description
NByteOrder value	[in] Enumeration value to check.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NIOTypes Unit [493]

7.1.2.24.1.4 NFileAccessIsValid Function

Checks whether enumeration value is valid.

C++

```
NBool N_API NFileAccessIsValid(NFileAccess value);
```

Parameters

Parameters	Description
NFileAccess value	[in] Enumeration value to check.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NIOTypes Unit [493]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.24.1.5 NFileModeIsValid Function

Checks whether enumeration value is valid.

C++

```
NBool N_API NFileModeIsValid(NFileMode value);
```

Parameters

Parameters	Description
NFileMode value	[in] Enumeration value to check.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NIOTypes Unit [493]

7.1.2.24.1.6 NFileShareIsValid Function

Checks whether enumeration value is valid.

C++

```
NBool N_API NFileShareIsValid(NFileShare value);
```

Parameters

Parameters	Description
NFileShare value	[in] Enumeration value to check.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NIOTypes Unit [493]

7.1.2.24.1.7 NSeekOriginIsValid Function

Checks whether enumeration value is valid.

C++

```
NBool N_API NSeekOriginIsValid(NSeekOrigin value);
```

Parameters

Parameters	Description
NSeekOrigin value	[in] Enumeration value to check.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NIOTypes Unit [493]

7.1.2.24.2 Structs, Records, Enums

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.24.2.1 NByteOrder Enumeration

Specifies byte order.

C++

```
typedef enum NByteOrder_ {
  nboLittleEndian = 0,
  nboBigEndian = 1
} NByteOrder;
```

Members

Members	Description
nboLittleEndian = 0	Little-endian byte order.
nboBigEndian = 1	Big-endian byte order.

Module

NIOTypes Unit [493]

7.1.2.24.2.2 NFileAccess Enumeration

Specifies access to a file.

C++

```
typedef enum NFileAccess_ {
  nfaRead = 1,
  nfaWrite = 2,
  nfaReadWrite = nfaRead|nfaWrite
} NFileAccess;
```

Members

Members	Description
nfaRead = 1	Read access to the file.
nfaWrite = 2	Write access to the file.
nfaReadWrite = nfaRead|nfaWrite	Read and write access to the file.

Module

NIOTypes Unit [493]

7.1.2.24.2.3 NFileMode Enumeration

Specifies how the operating system should open a file.

C++

```
typedef enum NFileMode_ {
  nfmCreateNew = 1,
  nfmCreate = 2,
  nfmOpen = 3,
  nfmOpenOrCreate = 4,
  nfmTruncate = 5,
  nfmAppend = 6
} NFileMode;
```

Members

Members	Description
nfmCreateNew = 1	Specifies that the operating system should create a new file.
nfmCreate = 2	Specifies that the operating system should create a new file. If the file already exists, it will be overwritten.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

nfmOpen = 3	Specifies that the operating system should open an existing file.
nfmOpenOrCreate = 4	Specifies that the operating system should open a file if it exists; otherwise, a new file should be created.
nfmTruncate = 5	Specifies that the operating system should open an existing file. When the file is opened, it should be truncated so that its size is zero bytes.
nfmAppend = 6	File can be appended. Opens the file if it exists and seeks to the end of the file, or creates a new file.

Module

NIOTypes Unit [493]

7.1.2.24.2.4 NFileShare Enumeration

Specifies values for controlling the kind of access to a file.

C++

```
typedef enum NFileShare_ {
  nfsNone = 0,
  nfsRead = 1,
  nfsWrite = 2,
  nfsReadWrite = nfsRead|nfsWrite
} NFileShare;
```

Members

Members	Description
nfsNone = 0	Declines sharing of the current file. Any request to open the file (by this process or another process) will fail until the file is closed.
nfsRead = 1	Allows opening of the file for reading. If this value is not specified, any request to open the file for reading will fail until the file is closed
nfsWrite = 2	Allows opening of the file for writing.
nfsReadWrite = nfsRead|nfsWrite	Allows opening of the file for reading or writing.

Module

NIOTypes Unit [493]

7.1.2.24.2.5 NSeekOrigin Enumeration

Specifies values that represent reference points in streams for seeking.

C++

```
typedef enum NSeekOrigin_ {
  nsoBegin = 0,
  nsoCurrent = 1,
  nsoEnd = 2
} NSeekOrigin;
```

Members

Members	Description
nsoBegin = 0	Specifies the beginning of a stream.
nsoCurrent = 1	Specifies the current position within a stream.
nsoEnd = 2	Specifies the end of a stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NIOTypes Unit [493]

7.1.2.24.3 Files

7.1.2.24.3.1 NIOTypes.h

Header file for NIOTypes [493] Unit.

Enumerations

	Name	Description
	NByteOrder_ [496]	Specifies byte order.
	NFileAccess_ [496]	Specifies access to a file.
	NFileMode_ [496]	Specifies how the operating system should open a file.
	NFileShare_ [497]	Specifies values for controlling the kind of access to a file.
	NSeekOrigin_ [497]	Specifies values that represent reference points in streams for seeking.
	NByteOrder [496]	Specifies byte order.
	NFileAccess [496]	Specifies access to a file.
	NFileMode [496]	Specifies how the operating system should open a file.
	NFileShare [497]	Specifies values for controlling the kind of access to a file.
	NSeekOrigin [497]	Specifies values that represent reference points in streams for seeking.

Functions

	Name	Description
	NByteOrderGetSystem [493]	Gets system byte order.
	NByteOrderIsReverse [494]	Checks if specified byte order is reverse.
	NByteOrderIsValid [494]	Checks whether enumeration value is valid.
	NFileAccessIsValid [494]	Checks whether enumeration value is valid.
	NFileModeIsValid [495]	Checks whether enumeration value is valid.
	NFileShareIsValid [495]	Checks whether enumeration value is valid.
	NSeekOriginIsValid [495]	Checks whether enumeration value is valid.

Module

NIOTypes Unit [493]

7.1.2.25 NLibraryInfo Unit

Provides definitions of library info structure type.

Files

Name	Description
NLibraryInfo.h [500]	Header file for the NLibraryInfo module. Provides definitions of library info structure type.

Macros

Name	Description
N_LI_ACTIVATED_MAX_LENGTH [499]	Maximum length of "Activated" field.
N_LI_COMPANY_MAX_LENGTH [500]	Maximum length of company name.
N_LI_COPYRIGHT_MAX_LENGTH [500]	Maximum length of copyright notice.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_LI_PRODUCT_MAX_LENGTH [500]	Maximum length of product name.
N_LI_TITLE_MAX_LENGTH [500]	Maximum length of library title.

Structs, Records, Enums

	Name	Description
	NLibraryInfo_ [499]	Structure defining information about the library as library title, product name, company name, copyright string and library version.

7.1.2.25.1 Structs, Records, Enums

7.1.2.25.1.1 NLibraryInfo_ Structure

Structure defining information about the library as library title, product name, company name, copyright string and library version.

C++

```
struct NLibraryInfo_ {
  NChar Title[N_LI_TITLE_MAX_LENGTH];
  NChar Product[N_LI_PRODUCT_MAX_LENGTH];
  NChar Company[N_LI_COMPANY_MAX_LENGTH];
  NChar Copyright[N_LI_COPYRIGHT_MAX_LENGTH];
  NInt VersionMajor;
  NInt VersionMinor;
  NInt VersionBuild;
  NInt VersionRevision;
  NChar Activated[N_LI_ACTIVATED_MAX_LENGTH];
};
```

Members

Members	Description
NChar Title[N_LI_TITLE_MAX_LENGTH];	Title of the library.
NChar Product[N_LI_PRODUCT_MAX_LENGTH];	Product name.
NChar Company[N_LI_COMPANY_MAX_LENGTH];	Company name.
NChar Copyright[N_LI_COPYRIGHT_MAX_LENGTH];	Copyright string for the library.
NInt VersionMajor;	Major part of the library version.
NInt VersionMinor;	Minor part of the library version.
NInt VersionBuild;	Build part of the library version.
NInt VersionRevision;	Revision part of the library version.
NChar Activated[N_LI_ACTIVATED_MAX_LENGTH];	Indicates if the library was activated.

Module

NLibraryInfo Unit [498]

7.1.2.25.2 Macros

7.1.2.25.2.1 N_LI_ACTIVATED_MAX_LENGTH Macro

Maximum length of "Activated" field.

C++

```
#define N_LI_ACTIVATED_MAX_LENGTH 256
```

Module

NLibraryInfo Unit [498]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.25.2.2 N_LI_COMPANY_MAX_LENGTH Macro

Maximum length of company name.

C++

```
#define N_LI_COMPANY_MAX_LENGTH 64
```

Module

NLibraryInfo Unit [498]

7.1.2.25.2.3 N_LI_COPYRIGHT_MAX_LENGTH Macro

Maximum length of copyright notice.

C++

```
#define N_LI_COPYRIGHT_MAX_LENGTH 192
```

Module

NLibraryInfo Unit [498]

7.1.2.25.2.4 N_LI_PRODUCT_MAX_LENGTH Macro

Maximum length of product name.

C++

```
#define N_LI_PRODUCT_MAX_LENGTH 128
```

Module

NLibraryInfo Unit [498]

7.1.2.25.2.5 N_LI_TITLE_MAX_LENGTH Macro

Maximum length of library title.

C++

```
#define N_LI_TITLE_MAX_LENGTH 128
```

Module

NLibraryInfo Unit [498]

7.1.2.25.3 Files

7.1.2.25.3.1 NLibraryInfo.h

Header file for the NLibraryInfo module. Provides definitions of library info structure type.

Macros

Name	Description
N_LI_ACTIVATED_MAX_LENGTH [499]	Maximum length of "Activated" field.
N_LI_COMPANY_MAX_LENGTH [500]	Maximum length of company name.
N_LI_COPYRIGHT_MAX_LENGTH [500]	Maximum length of copyright notice.
N_LI_PRODUCT_MAX_LENGTH [500]	Maximum length of product name.
N_LI_TITLE_MAX_LENGTH [500]	Maximum length of library title.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NLibraryInfo Unit [498]

Structures

	Name	Description
	NLibraryInfo_ [499]	Structure defining information about the library as library title, product name, company name, copyright string and library version.

7.1.2.26 NMemberDescriptor Unit

Provides member descriptor management functionality.

Files

Name	Description
NMemberDescriptor.h [501]	Header file for the NMemberDescriptor unit.

Functions

	Name	Description
	NMemberDescriptorGetComponentType [501]	Retrieves the component type of the specified member descriptor.

7.1.2.26.1 Functions

7.1.2.26.1.1 NMemberDescriptorGetComponentType Function

Retrieves the component type of the specified member descriptor.

C++

```
NResult N_API NMemberDescriptorGetComponentType(HNMemberDescriptor hMemberDescriptor,
HNType * phValue);
```

Parameters

Parameters	Description
HNMemberDescriptor hMemberDescriptor	[in] Handle to the member descriptor.
HNType * phValue	[out] Pointer to a handle to the member descriptor type.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemberDescriptor Unit [501]

7.1.2.26.2 Files

7.1.2.26.2.1 NMemberDescriptor.h

Header file for the NMemberDescriptor [501] unit.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NMemberDescriptorGetComponentType [501]	Retrieves the component type of the specified member descriptor.

Module

NMemberDescriptor Unit [501]

7.1.2.27 NMemory Unit

Provides memory management for Neurotechnology components.

Files

Name	Description
NMemory.h [509]	Header file for the NMemory module. Provides memory management for Neurotechnology components.

Functions

	Name	Description
	NAlignedFree [503]	Frees an aligned memory resources.
	NAlignedOffsetAlloc [503]	Allocates an aligned memory block.
	NAlignedOffsetCAlloc [503]	Allocates an aligned memory block with all bytes set to zero.
	NAlignedOffsetReAlloc [504]	Reallocates an aligned memory block.
	NAlloc [504]	Allocates memory block.
	NCAlloc [505]	Allocates memory block with all bytes set to zero.
	NCompare [505]	Compares bytes in two memory blocks.
	NCopy [506]	Copies data between memory blocks.
	NFill [506]	Sets bytes of memory block to specified value.
	NFree [506]	Deallocates memory block.
	NFreeEx [507]	Deallocates memory block of a specified type.
	NMemoryTypeIsValid [507]	Checks if enumeration value is correct.
	NMemoryTypeIsValidForFree [507]	Checks if enumeration value for NFreeEx [507] is correct.
	NMove [508]	Move data from one memory block to another.
	NReAlloc [508]	Reallocates memory block.

Macros

Name	Description
NClear [509]	Clears memory block.

Structs, Records, Enums

	Name	Description
	NMemoryType_ [509]	Enumerates memory types.
	NMemoryType [509]	Enumerates memory types.

7.1.2.27.1 Functions

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.27.1.1 NAlignedFree Function

Frees an aligned memory resources.

C++

```
void N_API NAlignedFree(void * pBlock);
```

Parameters

Parameters	Description
void * pBlock	[out] A pointer to previously allocated memory block to free.

Remarks

If passed NULL [1075] as a parameter, function does nothing. Attempting to free an invalid pointer, can cause errors.

Module

NMemory Unit [502]

7.1.2.27.1.2 NAlignedOffsetAlloc Function

Allocates an aligned memory block.

C++

```
NResult N_API NAlignedOffsetAlloc(NSizeType size, NSizeType alignment, NSizeType offset,
void * * ppBlock);
```

Parameters

Parameters	Description
NSizeType size	[in] The size in bytes of memory to be allocated.
NSizeType alignment	[in] The alignment value, which must be an integer power of 2.
NSizeType offset	[in] The offset into the memory allocation to force the alignment.
void * * ppBlock	[out] A pointer to a newly allocated memory block.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemory Unit [502]

7.1.2.27.1.3 NAlignedOffsetCAlloc Function

Allocates an aligned memory block with all bytes set to zero.

C++

```
NResult N_API NAlignedOffsetCAlloc(NSizeType size, NSizeType alignment, NSizeType offset,
void * * ppBlock);
```

Parameters

Parameters	Description
NSizeType size	[in] The size in bytes of memory to be allocated.
NSizeType alignment	[in] The alignment value, which must be an integer power of 2.
NSizeType offset	[in] The offset into the memory allocation to force the alignment.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

void * * ppBlock	[out] A pointer to a newly allocated memory block.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemory Unit [502]

7.1.2.27.1.4 NAlignedOffsetReAlloc Function

Reallocates an aligned memory block.

C++

```
NResult N_API NAlignedOffsetReAlloc(void * * ppBlock, NSizeType size, NSizeType alignment,
NSizeType offset);
```

Parameters

Parameters	Description
void * * ppBlock	[in] A pointer to a previously allocated memory block which should be reallocated.
NSizeType size	[in] The size in bytes of a new memory block.
NSizeType alignment	[in] The alignment value, which must be an integer power of 2.
NSizeType offset	[in] The offset into the memory allocation to force the alignment.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemory Unit [502]

7.1.2.27.1.5 NAlloc Function

Allocates memory block.

C++

```
NResult N_API NAlloc(NSizeType size, void * * ppBlock);
```

Parameters

Parameters	Description
NSizeType size	[in] The size in bytes of memory to be allocated.
void * * ppBlock	[out] A pointer to a newly allocated memory block.

Returns

Return value	Description
N_OK [433]	If function succeeds it returns N_OK [433].
N_E_ARGUMENT_NULL [427]	If provided *ppBlock* value is NULL [1075], function fails and returns this error.
N_E_OUT_OF_MEMORY [432]	If there was nor enough memory to allocate *size* bytes this error is returned.

Module

NMemory Unit [502]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.27.1.6 NCAlloc Function

Allocates memory block with all bytes set to zero.

C++

```
NResult N_API NCAlloc(NSizeType size, void * * ppBlock);
```

Parameters

Parameters	Description
NSizeType size	[in] The size in bytes of memory to be allocated.
void * * ppBlock	[out] A pointer to a newly allocated memory block.

Returns

Return value	Description
N_OK [433]	If function succeeds it returns N_OK [433].
N_E_ARGUMENT_NULL [427]	If provided *ppBlock* value is NULL [1075], function fails and returns this error.
N_E_OUT_OF_MEMORY [432]	If there was nor enough memory to allocate *size* bytes this error is returned.

Module

NMemory Unit [502]

7.1.2.27.1.7 NCompare Function

Compares bytes in two memory blocks.

C++

```
NResult N_API NCompare(const void * pBlock1, const void * pBlock2, NSizeType size, NInt *
pResult);
```

Parameters

Parameters	Description
const void * pBlock1	[in] A pointer to the first memory block.
const void * pBlock2	[in] A pointer to the second memory block.
NSizeType size	[in] Number of bytes to be compared.
NInt * pResult	[out] Compare result.

Returns

Return value	Description
N_OK [433]	If function succeeds it returns N_OK [433].
N_E_ARGUMENT_NULL [427]	If provided *ppBlock1* or *ppBlock2* value is NULL [1075], function fails and returns this error.

Remarks

If *pResult* value is less than zero then a block specified by *pBlock1* is less than a block specified by *pBlock2*.

If *pResult* has a zero value then a block specified by *pBlock1* is identical to a block specified by *pBlock2*.

If *pResult* is greater than zero then A block specified by *pBlock1* is greater than a block specified by *pBlock2*.

Module

NMemory Unit [502]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.27.1.8 NCopy Function

Copies data between memory blocks.

C++

```
NResult N_API NCopy(void * pDstBlock, const void * pSrcBlock, NSizeType size);
```

Parameters

Parameters	Description
void * pDstBlock	[in] A destination memory block (memory block to copy from).
const void * pSrcBlock	[in] A source memory block (memory block to copy to).
NSizeType size	[in] The size of memory block to be copied (the size is in bytes).

Returns

Return value	Description
N_OK [433]	If function succeeds it returns N_OK [433].
N_E_ARGUMENT_NULL [427]	If provided *pDstBlock* or *pDstBlock* value is NULL [1075], function fails and returns this error.

Module

NMemory Unit [502]

7.1.2.27.1.9 NFill Function

Sets bytes of memory block to specified value.

C++

```
NResult N_API NFill(void * pBlock, NByte value, NSizeType size);
```

Parameters

Parameters	Description
void * pBlock	[in] A pointer to a destination memory block that contains data.
NByte value	[in] A value to set.
NSizeType size	[in] The size in bytes of memory block to be set.

Returns

Return value	Description
N_OK [433]	If function succeeds it returns N_OK [433].
N_E_ARGUMENT_NULL [427]	If provided *pBlock* value is NULL [1075], function fails and returns this error.

Module

NMemory Unit [502]

7.1.2.27.1.10 NFree Function

Deallocates memory block.

C++

```
void N_API NFree(void * pBlock);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
void * pBlock	[in] A pointer to memory block to be deallocated.

Returns

If function succeeds the return value is N_OK [433].

Module

NMemory Unit [502]

7.1.2.27.1.11 NFreeEx Function

Deallocates memory block of a specified type.

C++

```
void N_API NFreeEx(void * pBlock, NMemoryType memoryType);
```

Parameters

Parameters	Description
void * pBlock	[in] A pointer to memory block to be deallocated.
NMemoryType memoryType	[in] Type of the memory block to be dealloacted.

Module

NMemory Unit [502]

7.1.2.27.1.12 NMemoryTypeIsValid Function

Checks if enumeration value is correct.

C++

```
NBool N_API NMemoryTypeIsValid(NMemoryType value);
```

Parameters

Parameters	Description
NMemoryType value	[in] Type of the memory block.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemory Unit [502]

7.1.2.27.1.13 NMemoryTypeIsValidForFree Function

Checks if enumeration value for NFreeEx [507] is correct.

C++

```
NBool N_API NMemoryTypeIsValidForFree(NMemoryType value);
```

Parameters

Parameters	Description
NMemoryType value	[in] Type of the memory block.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NMemory Unit [502]

7.1.2.27.1.14 NMove Function

Move data from one memory block to another.

C++

```
NResult N_API NMove(void * pDstBlock, const void * pSrcBlock, NSizeType size);
```

Parameters

Parameters	Description
void * pDstBlock	[in] A pointer to destination memory block.
const void * pSrcBlock	[in] A pointer to source memory block.
NSizeType size	[in] Size in bytes to be copied.

Returns

If the function succeeds, the return value is N_OK [433].

Remarks

Copies *size* bytes from *pDstBlock* to *pSrcBlock*.

Make sure that the destination buffer is the same size or larger than the source buffer.

Module

NMemory Unit [502]

7.1.2.27.1.15 NReAlloc Function

Reallocates memory block.

C++

```
NResult N_API NReAlloc(void * * ppBlock, NSizeType size);
```

Parameters

Parameters	Description
void * * ppBlock	[in] Pointer to previously allocated memory block which should be reallocated.
NSizeType size	[in] The size in bytes of a new memory block.

Returns

Return value	Description
N_OK [433]	If function succeeds it returns N_OK [433].
N_E_ARGUMENT_NULL [427]	If provided *ppBlock* value is NULL [1075], function fails and returns this error.
N_E_OUT_OF_MEMORY [432]	If there was nor enough memory to allocate *size* bytes this error is returned.

Module

NMemory Unit [502]

7.1.2.27.2 Structs, Records, Enums

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.27.2.1 NMemoryType Enumeration

Enumerates memory types.

C++

```
typedef enum NMemoryType_ {
  nmmtNone = 0,
  nmmtDefault = 1,
  nmmtAligned = 2,
  nmmtStandard = 3,
  nmmtWin32Local = 4,
  nmmtWin32Global = 5,
  nmmtCustom = 15
} NMemoryType;
```

Members

Members	Description
nmmtNone = 0	Indicates no memory.
nmmtDefault = 1	Indicates default memory.
nmmtAligned = 2	Indicates aligned memory.
nmmtStandard = 3	Indicates standard memory.
nmmtWin32Local = 4	Indicates Win32 Local memory.
nmmtWin32Global = 5	Indicates Win32 Global memory.
nmmtCustom = 15	Indicates custom memory.

Module

NMemory Unit [502]

7.1.2.27.3 Macros

7.1.2.27.3.1 NClear Macro

Clears memory block.

C++

```
#define NClear(pBlock, size) NFill(pBlock, 0, size)
```

Module

NMemory Unit [502]

7.1.2.27.4 Files

7.1.2.27.4.1 NMemory.h

Header file for the NMemory [502] module. Provides memory management for Neurotechnology components.

Enumerations

	Name	Description
	NMemoryType_ [509]	Enumerates memory types.
	NMemoryType [509]	Enumerates memory types.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NAlignedFree [503]	Frees an aligned memory resources.
	NAlignedOffsetAlloc [503]	Allocates an aligned memory block.
	NAlignedOffsetCAlloc [503]	Allocates an aligned memory block with all bytes set to zero.
	NAlignedOffsetReAlloc [504]	Reallocates an aligned memory block.
	NAlloc [504]	Allocates memory block.
	NCAlloc [505]	Allocates memory block with all bytes set to zero.
	NCompare [505]	Compares bytes in two memory blocks.
	NCopy [506]	Copies data between memory blocks.
	NFill [506]	Sets bytes of memory block to specified value.
	NFree [506]	Deallocates memory block.
	NFreeEx [507]	Deallocates memory block of a specified type.
	NMemoryTypeIsValid [507]	Checks if enumeration value is correct.
	NMemoryTypeIsValidForFree [507]	Checks if enumeration value for NFreeEx [507] is correct.
	NMove [508]	Move data from one memory block to another.
	NReAlloc [508]	Reallocates memory block.

Macros

Name	Description
NClear [509]	Clears memory block.

Module

NMemory Unit [502]

7.1.2.28 NMemoryStream Unit

Provides functions for working with memory streams.

Functions

	Name	Description
	NMemoryStreamCreate [511]	Creates a stream with an expandable capacity initialized to zero
	NMemoryStreamCreateFromBuffer [511]	Creates a new NMemoryStream based on the specified byte array, with the access other NMemoryStream (s) can have to the same file, and the buffer size.
	NMemoryStreamCreateFromBufferN [512]	Creates a new NMemoryStream with a the specified byte array and an access other NMemoryStreams can have to the same file.
	NMemoryStreamCreateWithCapacity [512]	Creates a new NMemoryStream with an expandable capacity initialized as specified.
	NMemoryStreamDetachBuffer [512]	Returns the memory contained in the buffer without copying it.
	NMemoryStreamGetBuffer [513]	Returns NBuffer [313] from which this stream was created.
	NMemoryStreamGetCapacity [513]	Returns the number of bytes allocated for this stream.
	NMemoryStreamGetPositionPtr [513]	Returns the current position within the stream.
	NMemoryStreamSetCapacity [514]	Sets the length of the usable portion of the buffer for the stream.
	NMemoryStreamSetPositionPtr [514]	Sets the current position within the stream.
	NMemoryStreamToPtr [514]	Copies the memory contained in the buffer and returns it as a pointer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NMemoryStreamWriteTo [515]	Writes the entire contents of NMemoryStream to another stream.

Macros

Name	Description
NMemoryStreamCreateFromBufferForRead [515]	Creates a new NMemoryStream based on the specified byte array.

7.1.2.28.1 Functions

7.1.2.28.1.1 NMemoryStreamCreate Function

Creates a stream with an expandable capacity initialized to zero

C++

```
NResult N_API NMemoryStreamCreate(HNMemoryStream * phStream);
```

Parameters

Parameters	Description
HNMemoryStream * phStream	[out] A pointer to the created NMemoryStream [510].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

7.1.2.28.1.2 NMemoryStreamCreateFromBuffer Function

Creates a new NMemoryStream [510] based on the specified byte array, with the access other NMemoryStream [510] (s) can have to the same file, and the buffer size.

C++

```
NResult N_API NMemoryStreamCreateFromBuffer(void * pBuffer, NSizeType bufferSize,
NFileAccess access, NBool bufferExposable, HNMemoryStream * phStream);
```

Parameters

Parameters	Description
void * pBuffer	[out] A pointer to an array of unsigned bytes from which to create this stream.
NSizeType bufferSize	[in] Size of memory buffer to store NMemoryStream [510]. Can be zero.
NFileAccess access	[in] Value determining how the stream can be accessed by the NMemoryStream [510]. NStreamCanRead [661] and NStreamCanWrite [662] function can be used to check stream access.
NBool bufferExposable	[in] NTrue [1064] to enable NStreamGetBuffer, which returns the unsigned byte array from which the stream was created; otherwise, NFalse [1041].
HNMemoryStream * phStream	[out] A pointer to the created NMemoryStream [510].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.28.1.3 NMemoryStreamCreateFromBufferN Function

Creates a new NMemoryStream [510] with a the specified byte array and an access other NMemoryStreams can have to the same file.

C++

```
NResult N_API NMemoryStreamCreateFromBufferN(HNBuffer hBuffer, NFileAccess access, NBool
bufferExposable, HNMemoryStream * phStream);
```

Parameters

Parameters	Description
HNBuffer hBuffer	[in] A handle to an array of unsigned bytes from which to create this stream.
NFileAccess access	[in] File access value determining how the file can be accessed by the NMemoryStream [510]. This gets the NStreamCanRead [661] and NStreamCanWrite [662] properties of the NMemoryStream [510] object.
NBool bufferExposable	[in] NTrue [1064] to enable NMemoryStreamGetBuffer [513], which returns the NBuffer [313] from which the stream was created; otherwise, NFalse [1041].
HNMemoryStream * phStream	[out] A pointer to the created NMemoryStream [510].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

7.1.2.28.1.4 NMemoryStreamCreateWithCapacity Function

Creates a new NMemoryStream [510] with an expandable capacity initialized as specified.

C++

```
NResult N_API NMemoryStreamCreateWithCapacity(NSizeType capacity, HNMemoryStream *
phStream);
```

Parameters

Parameters	Description
NSizeType capacity	[in] Gets or sets the number of bytes allocated for this stream.
HNMemoryStream * phStream	[out] A pointer to the created NMemoryStream [510].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

7.1.2.28.1.5 NMemoryStreamDetachBuffer Function

Returns the memory contained in the buffer without copying it.

C++

```
NResult N_API NMemoryStreamDetachBuffer(HNMemoryStream hStream, void * * ppBuffer,
NSizeType * pBufferSize);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNMemoryStream hStream	[in] A handle to NMemoryStream [510].
void * * ppBuffer	[out] A pointer to a pointer to an array of unsigned bytes from which this stream was created.
NSizeType * pBufferSize	[out] A pointer to NSizeType [1008] value greater than 0 indicating the buffer size.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

7.1.2.28.1.6 NMemoryStreamGetBuffer Function

Returns NBuffer [313] from which this stream was created.

C++

```
NResult N_API NMemoryStreamGetBuffer(HNMemoryStream hStream, HNBuffer * phValue);
```

Parameters

Parameters	Description
HNMemoryStream hStream	[in] A handle to NMemoryStream [510].
HNBuffer * phValue	[out] A pointer NBuffer [313] from which to create this stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

7.1.2.28.1.7 NMemoryStreamGetCapacity Function

Returns the number of bytes allocated for this stream.

C++

```
NResult N_API NMemoryStreamGetCapacity(HNMemoryStream hStream, NSizeType * pValue);
```

Parameters

Parameters	Description
HNMemoryStream hStream	[in] A handle to NMemoryStream [510].
NSizeType * pValue	[out] A pointer to the number of bytes allocated for this stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

7.1.2.28.1.8 NMemoryStreamGetPositionPtr Function

Returns the current position within the stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NMemoryStreamGetPositionPtr(HNMemoryStream hStream, void * * pValue);
```

Parameters

Parameters	Description
HNMemoryStream hStream	[in] A handle to NMemoryStream [510].
void * * pValue	[out] A pointer to the current position within the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

7.1.2.28.1.9 NMemoryStreamSetCapacity Function

Sets the length of the usable portion of the buffer for the stream.

C++

```
NResult N_API NMemoryStreamSetCapacity(HNMemoryStream hStream, NSizeType value);
```

Parameters

Parameters	Description
HNMemoryStream hStream	[in] A handle to NMemoryStream [510]. value
NSizeType value	[in] The length of the usable portion of the buffer for the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

7.1.2.28.1.10 NMemoryStreamSetPositionPtr Function

Sets the current position within the stream.

C++

```
NResult N_API NMemoryStreamSetPositionPtr(HNMemoryStream hStream, void * value);
```

Parameters

Parameters	Description
HNMemoryStream hStream	[in] A handle to NMemoryStream [510].
void * value	[out] A pointer to the current position within the stream to set.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

7.1.2.28.1.11 NMemoryStreamToPtr Function

Copies the memory contained in the buffer and returns it as a pointer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NMemoryStreamToPtr(HNMemoryStream hStream, NSizeType * pBufferSize, void * *
ppBuffer);
```

Parameters

Parameters	Description
HNMemoryStream hStream	[in] A handle to NMemoryStream [510].
NSizeType * pBufferSize	[out] A NSizeType [1008] indicating the buffer size.
void * * ppBuffer	[out] A pointer to a pointer to an array of unsigned bytes from which this stream was created.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

7.1.2.28.1.12 NMemoryStreamWriteTo Function

Writes the entire contents of NMemoryStream [510] to another stream.

C++

```
NResult N_API NMemoryStreamWriteTo(HNMemoryStream hStream, HNStream hDstStream);
```

Parameters

Parameters	Description
HNMemoryStream hStream	[in] A handle to NMemoryStream [510].
HNStream hDstStream	[in] A handle to the destination stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

7.1.2.28.2 Macros

7.1.2.28.2.1 NMemoryStreamCreateFromBufferForRead Macro

Creates a new NMemoryStream [510] based on the specified byte array.

C++

```
#define NMemoryStreamCreateFromBufferForRead(pBuffer, bufferSize, phStream)
NMemoryStreamCreateFromBuffer((void *)pBuffer, bufferSize, nfaRead, NFalse, phStream)
```

Parameters

Parameters	Description
pBuffer	[out] A pointer to an array of unsigned bytes from which to create this stream.
bufferSize	[in] Size of memory buffer to store NMemoryStream [510]. Can be zero.
phStream	[out] Pointer to a handle to the created memory stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMemoryStream Unit [510]

7.1.2.29 NModule Unit

Each separate piece of code/functionality is a module. The functions in this unit allow to get the information about the module it's name, version, etc..

Functions

	Name	Description
	NModuleCheckInit [517]	Checks whether the module is initialized and throws and exception if it is not.
	NModuleCreate [517]	Creates a new NModule.
	NModuleGetActivated [518]	Get the activation status of a protected module.
	NModuleGetCompany [518]	Get the company name providing the component.
	NModuleGetCopyright [519]	Get the copyright string for the module.
	NModuleGetDependences [519]	Retrieves the module's dependencies.
	NModuleGetName [519]	Get the short name of the given module.
	NModuleGetOptions [520]	Get the set of component's flags.
	NModuleGetProduct [520]	Get the product name of the given module.
	NModuleGetTitle [521]	Get the title string (long name with version) for the given module.
	NModuleGetVersionBuild [521]	Get the build number part of the version (Major.Minor.Revision.Build).
	NModuleGetVersionMajor [521]	Get the major number part of the version (Major.Minor.Revision.Build).
	NModuleGetVersionMinor [522]	Get the minor number part of the version (Major.Minor.Revision.Build).
	NModuleGetVersionRevision [522]	Get the revision number part of the version (Major.Minor.Revision.Build).
	NModuleSetCompanyN [522]	Sets the company for the specified NModule.
	NModuleSetCopyrightN [523]	Sets the copyright for the specified NModule.
	NModuleSetNameN [523]	Sets the name for the specified NModule.
	NModuleSetOptions [523]	Sets the options for the specified module.
	NModuleSetProductN [524]	Sets the product of the specified module.
	NModuleSetTitleN [524]	Sets the title for the specified NModule.
	NModuleSetVersionBuild [524]	Sets the version build value for the specified module.
	NModuleSetVersionMajor [525]	Sets the version major value for the specified module.
	NModuleSetVersionMinor [525]	Sets the version minor value for the specified module.
	NModuleSetVersionRevision [525]	Sets the version revision value for the specified module.

Macros

Name	Description
N_DECLARE_MODULE [526]	Declares the module.
N_MODULE_OF [526]	Used to obtain a pointer to the function that retrieves NModule object for the specified module name.
NModuleGetActivated [527]	Picks a Unicode (W) or ANSCII (A) version of the NModuleGetActivated function.
NModuleGetCompany [527]	Picks a Unicode (W) or ANSCII (A) version of the NModuleGetCompany function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NModuleGetCopyright [527]	Picks a Unicode (W) or ANSCII (A) version of the NModuleGetCopyright function.
NModuleGetName [527]	Picks a Unicode (W) or ANSCII (A) version of the NModuleGetName function.
NModuleGetProduct [527]	Picks a Unicode (W) or ANSCII (A) version of the NModuleGetProduct function.
NModuleGetTitle [527]	Picks a Unicode (W) or ANSCII (A) version of the NModuleGetTitle function.
NModuleSetCompany [528]	Picks a Unicode (W) or ASCII (A) version of the NModuleSetCompany function.
NModuleSetCopyright [528]	Picks a Unicode (W) or ASCII (A) version of the NModuleSetCopyright function.
NModuleSetName [528]	Picks a Unicode (W) or ASCII (A) version of the NModuleSetName function.
NModuleSetProduct [528]	Picks a Unicode (W) or ASCII (A) version of the NModuleSetProduct function.
NModuleSetTitle [528]	Picks a Unicode (W) or ASCII (A) version of the NModuleSetTitle function.

Structs, Records, Enums

	Name	Description
	NModuleOptions_ [526]	Enumeration of possible module flags giving additional information about the module.
	NModuleOptions [526]	Enumeration of possible module flags giving additional information about the module.

7.1.2.29.1 Functions

7.1.2.29.1.1 NModuleCheckInit Function

Checks whether the module is initialized and throws and exception if it is not.

C++

```
NResult N_API NModuleCheckInit(HNModule hModule, NBool lazy);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NBool lazy	[in] NTrue [1064] if the initialization is lazy; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.2 NModuleCreate Function

Creates a new NModule [516].

C++

```
NResult N_API NModuleCreate(HNModule * phModule);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNModule * phModule	[out] Pointer to a handle to the new NModule [516].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.3 NModuleGetActivated Function

Get the activation status of a protected module.

C++

```
NResult N_API NModuleGetActivated(HNModule hModule, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle of the module.
NChar * szValue	[out] Pointer to memory buffer that receives result as zero terminated string.
NInt valueSize	[in] Size available in szValue memory buffer (including zero terminator).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

The format of the string returned through szValue:

- "" (empty string) – for unprotected module;
- "Yes" or "No" – in case the module provides single protected component;
- "Component1:Yes/No, Component1:Yes/No, ..., ComponentN:Yes/No" – in case the module provides multiple protected components, the comma separated list of component names with "Yes" or "No" value for each component.

Module

NModule Unit [516]

7.1.2.29.1.4 NModuleGetCompany Function

Get the company name providing the component.

C++

```
NResult N_API NModuleGetCompany(HNModule hModule, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle of the module.
NChar * szValue	[out] Pointer to memory buffer that receives result as zero terminated string.
NInt valueSize	[in] Size available in szValue memory buffer (including zero terminator).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NModule Unit [516]

7.1.2.29.1.5 NModuleGetCopyright Function

Get the copyright string for the module.

C++

```
NResult N_API NModuleGetCopyright(HNModule hModule, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle of the module.
NChar * szValue	[out] Pointer to memory buffer that receives result as zero terminated string.
NInt valueSize	[in] Size available in szValue memory buffer (including zero terminator).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.6 NModuleGetDependences Function

Retrieves the module's dependencies.

C++

```
NResult N_API NModuleGetDependences(HNModule hModule, HNModule * arhValues, NInt
valuesLength);
```

Parameters

Parameters	Description
HNModule hModule	[in] A handle to the NModule [516].
HNModule * arhValues	[out] A pointer to an array of dependencies.
NInt valuesLength	[in] Length of the array of dependencies.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NModule Unit [516]

7.1.2.29.1.7 NModuleGetName Function

Get the short name of the given module.

C++

```
NResult N_API NModuleGetName(HNModule hModule, NChar * szValue, NInt valueSize);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNModule hModule	[in] Handle of the module.
NChar * szValue	[out] Pointer to memory buffer that receives result as zero terminated string.
NInt valueSize	[in] Size available in szValue memory buffer (including zero terminator).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.8 NModuleGetOptions Function

Get the set of component's flags.

C++

```
NResult N_API NModuleGetOptions(HNModule hModule, NModuleOptions * pValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle of the module.
NModuleOptions * pValue	[out] Pointer to variable that receives options.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

The returned value is bitwise OR composition of individual NModuleOptions [526] flags.

Module

NModule Unit [516]

7.1.2.29.1.9 NModuleGetProduct Function

Get the product name of the given module.

C++

```
NResult N_API NModuleGetProduct(HNModule hModule, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle of the module.
NChar * szValue	[out] Pointer to memory buffer that receives result as zero terminated string.
NInt valueSize	[in] Size available in szValue memory buffer (including zero terminator).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.29.1.10 NModuleGetTitle Function

Get the title string (long name with version) for the given module.

C++

```
NResult N_API NModuleGetTitle(HNModule hModule, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle of the module.
NChar * szValue	[out] Pointer to memory buffer that receives result as zero terminated string.
NInt valueSize	[in] Size available in szValue memory buffer (including zero terminator).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.11 NModuleGetVersionBuild Function

Get the build number part of the version (Major.Minor.Revision.Build).

C++

```
NResult N_API NModuleGetVersionBuild(HNModule hModule, NInt * pValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle of the module.
NInt * pValue	[out] Pointer to variable that receives requested value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.12 NModuleGetVersionMajor Function

Get the major number part of the version (Major.Minor.Revision.Build).

C++

```
NResult N_API NModuleGetVersionMajor(HNModule hModule, NInt * pValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle of the module.
NInt * pValue	[out] Pointer to variable that receives requested value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.29.1.13 NModuleGetVersionMinor Function

Get the minor number part of the version (Major.Minor.Revision.Build).

C++

```
NResult N_API NModuleGetVersionMinor(HNModule hModule, NInt * pValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle of the module.
NInt * pValue	[out] Pointer to variable that receives requested value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.14 NModuleGetVersionRevision Function

Get the revision number part of the version (Major.Minor.Revision.Build).

C++

```
NResult N_API NModuleGetVersionRevision(HNModule hModule, NInt * pValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle of the module.
NInt * pValue	[out] Pointer to variable that receives requested value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.15 NModuleSetCompanyN Function

Sets the company for the specified NModule [516].

C++

```
NResult N_API NModuleSetCompanyN(HNModule hModule, HNString hValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
HNString hValue	[in] Handle to the new company string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.29.1.16 NModuleSetCopyrightN Function

Sets the copyright for the specified NModule [516].

C++

```
NResult N_API NModuleSetCopyrightN(HNModule hModule, HNString hValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
HNString hValue	[in] Handle to the new copyright string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.17 NModuleSetNameN Function

Sets the name for the specified NModule [516].

C++

```
NResult N_API NModuleSetNameN(HNModule hModule, HNString hValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
HNString hValue	[in] Handle to the new name string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.18 NModuleSetOptions Function

Sets the options for the specified module.

C++

```
NResult N_API NModuleSetOptions(HNModule hModule, NModuleOptions value);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NModuleOptions value	[in] Module options.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.29.1.19 NModuleSetProductN Function

Sets the product of the specified module.

C++

```
NResult N_API NModuleSetProductN(HNModule hModule, HNString hValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
HNString hValue	[in] Handle to the new product string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.20 NModuleSetTitleN Function

Sets the title for the specified NModule [516].

C++

```
NResult N_API NModuleSetTitleN(HNModule hModule, HNString hValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
HNString hValue	[in] Handle to the new title string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.21 NModuleSetVersionBuild Function

Sets the version build value for the specified module.

C++

```
NResult N_API NModuleSetVersionBuild(HNModule hModule, NInt value);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NInt value	[in] New version build value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.29.1.22 NModuleSetVersionMajor Function

Sets the version major value for the specified module.

C++

```
NResult N_API NModuleSetVersionMajor(HNModule hModule, NInt value);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NInt value	[in] New version major value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.23 NModuleSetVersionMinor Function

Sets the version minor value for the specified module.

C++

```
NResult N_API NModuleSetVersionMinor(HNModule hModule, NInt value);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NInt value	[in] New version minor value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

7.1.2.29.1.24 NModuleSetVersionRevision Function

Sets the version revision value for the specified module.

C++

```
NResult N_API NModuleSetVersionRevision(HNModule hModule, NInt value);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NInt value	[in] New version revision value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModule Unit [516]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.29.2 Structs, Records, Enums

7.1.2.29.2.1 NModuleOptions Enumeration

Enumeration of possible module flags giving additional information about the module.

C++

```
typedef enum NModuleOptions_ {
  nmoNone = 0,
  nmoDebug = 0x01,
  nmoProtected = 0x02,
  nmoUnicode = 0x04,
  nmoNoAnsiFunc = 0x08,
  nmoNoUnicode = 0x10,
  nmoLib = 0x20,
  nmoExe = 0x40
} NModuleOptions;
```

Members

Members	Description
nmoNone = 0	Module has no flags set.
nmoDebug = 0x01	Module includes debug information.
nmoProtected = 0x02	Module is protected and requires activation.
nmoUnicode = 0x04	Module functions support Unicode strings.
nmoNoAnsiFunc = 0x08	Module doesn't provide functions for ANSI encoded strings.
nmoNoUnicode = 0x10	Module doesn't provide functions for Unicode encoded strings.
nmoLib = 0x20	Module is compiled as a library.
nmoExe = 0x40	Module is compiled into the executable.

Module

NModule Unit [516]

7.1.2.29.3 Macros

7.1.2.29.3.1 N_DECLARE_MODULE Macro

Declares the module.

C++

```
#define N_DECLARE_MODULE(name) \
    NResult N_API name##ModuleOf(HNModule * phValue);
```

Module

NModule Unit [516]

7.1.2.29.3.2 N_MODULE_OF Macro

Used to obtain a pointer to the function that retrieves NModule [516] object for the specified module name.

C++

```
#define N_MODULE_OF(name) N_JOIN_SYMBOLS(name, ModuleOf)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NModule Unit [516]

7.1.2.29.3.3 NModuleGetActivated Macro

Picks a Unicode (W) or ANSCII (A) version of the NModuleGetActivated function.

C++

```
#define NModuleGetActivated N_FUNC_AW(NModuleGetActivated)
```

Module

NModule Unit [516]

7.1.2.29.3.4 NModuleGetCompany Macro

Picks a Unicode (W) or ANSCII (A) version of the NModuleGetCompany function.

C++

```
#define NModuleGetCompany N_FUNC_AW(NModuleGetCompany)
```

Module

NModule Unit [516]

7.1.2.29.3.5 NModuleGetCopyright Macro

Picks a Unicode (W) or ANSCII (A) version of the NModuleGetCopyright function.

C++

```
#define NModuleGetCopyright N_FUNC_AW(NModuleGetCopyright)
```

Module

NModule Unit [516]

7.1.2.29.3.6 NModuleGetName Macro

Picks a Unicode (W) or ANSCII (A) version of the NModuleGetName function.

C++

```
#define NModuleGetName N_FUNC_AW(NModuleGetName)
```

Module

NModule Unit [516]

7.1.2.29.3.7 NModuleGetProduct Macro

Picks a Unicode (W) or ANSCII (A) version of the NModuleGetProduct function.

C++

```
#define NModuleGetProduct N_FUNC_AW(NModuleGetProduct)
```

Module

NModule Unit [516]

7.1.2.29.3.8 NModuleGetTitle Macro

Picks a Unicode (W) or ANSCII (A) version of the NModuleGetTitle function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NModuleGetTitle N_FUNC_AW(NModuleGetTitle)
```

Module

NModule Unit [516]

7.1.2.29.3.9 NModuleSetCompany Macro

Picks a Unicode (W) or ASCII (A) version of the NModuleSetCompany function.

C++

```
#define NModuleSetCompany N_FUNC_AW(NModuleSetCompany)
```

Module

NModule Unit [516]

7.1.2.29.3.10 NModuleSetCopyright Macro

Picks a Unicode (W) or ASCII (A) version of the NModuleSetCopyright function.

C++

```
#define NModuleSetCopyright N_FUNC_AW(NModuleSetCopyright)
```

Module

NModule Unit [516]

7.1.2.29.3.11 NModuleSetName Macro

Picks a Unicode (W) or ASCII (A) version of the NModuleSetName function.

C++

```
#define NModuleSetName N_FUNC_AW(NModuleSetName)
```

Module

NModule Unit [516]

7.1.2.29.3.12 NModuleSetProduct Macro

Picks a Unicode (W) or ASCII (A) version of the NModuleSetProduct function.

C++

```
#define NModuleSetProduct N_FUNC_AW(NModuleSetProduct)
```

Module

NModule Unit [516]

7.1.2.29.3.13 NModuleSetTitle Macro

Picks a Unicode (W) or ASCII (A) version of the NModuleSetTitle function.

C++

```
#define NModuleSetTitle N_FUNC_AW(NModuleSetTitle)
```

Module

NModule Unit [516]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.30 NModuleInternal Unit

Each separate piece of code/functionality is a module. The functions in this unit allow to set the information about the module it's name, version, etc..

Functions

	Name	Description
	NModuleRegister [529]	Register a created NModule [516].
	NModuleSetActivatedProc [529]	Sets the activated procedure for the specified module.
	NModuleSetDependences [530]	Sets the dependencies for the specified module.
	NModuleSetInit [530]	Sets the initialization procedure for the specified module.
	NModuleSetLazyInit [530]	Sets the lazy initialization procedure for the specified module.
	NModuleSetThreadInit [531]	Sets the thread initialization procedure for the specified module.
	NModuleSetThreadUninit [531]	Sets the thread uninitialization procedure for the specified module.
	NModuleSetUninit [531]	Sets the uninitialization procedure for the specified module.
	NModuleUnregister [532]	Unregisters a module.

7.1.2.30.1 Functions

7.1.2.30.1.1 NModuleRegister Function

Register a created NModule [516].

C++

```
NResult N_API NModuleRegister(NModuleCreateProc pCreate, HNModule * phModule);
```

Parameters

Parameters	Description
NModuleCreateProc pCreate	[in] Pointer to a NModule [516] creating function.
HNModule * phModule	[out] Pointer to a handle to the registered NModuel.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModuleInternal Unit [529]

7.1.2.30.1.2 NModuleSetActivatedProc Function

Sets the activated procedure for the specified module.

C++

```
NResult N_API NModuleSetActivatedProc(HNModule hModule, NModuleGetActivatedProc pValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NModuleGetActivatedProc pValue	[in] Pointer to the activated procedure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NModuleInternal Unit [529]

7.1.2.30.1.3 NModuleSetDependences Function

Sets the dependencies for the specified module.

C++

```
NResult N_API NModuleSetDependences(HNModule hModule, NModuleOfProc * arpValues, NInt count);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NModuleOfProc * arpValues	[out] A pointer to an array of dependencies.
NInt count	[in] Length of the array of dependencies.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModuleInternal Unit [529]

7.1.2.30.1.4 NModuleSetInit Function

Sets the initialization procedure for the specified module.

C++

```
NResult N_API NModuleSetInit(HNModule hModule, NModuleInitProc pValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NModuleInitProc pValue	[in] Pointer to the initialization procedure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModuleInternal Unit [529]

7.1.2.30.1.5 NModuleSetLazyInit Function

Sets the lazy initialization procedure for the specified module.

C++

```
NResult N_API NModuleSetLazyInit(HNModule hModule, NModuleLazyInitProc pValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NModuleLazyInitProc pValue	[in] Pointer to the lazy initialization procedure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NModuleInternal Unit [529]

7.1.2.30.1.6 NModuleSetThreadInit Function

Sets the thread initialization procedure for the specified module.

C++

```
NResult N_API NModuleSetThreadInit(HNModule hModule, NModuleThreadInitProc pValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NModuleThreadInitProc pValue	[in] Pointer to the thread initialization procedure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModuleInternal Unit [529]

7.1.2.30.1.7 NModuleSetThreadUninit Function

Sets the thread uninitialization procedure for the specified module.

C++

```
NResult N_API NModuleSetThreadUninit(HNModule hModule, NModuleThreadUninitProc pValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NModuleThreadUninitProc pValue	[in] Pointer to the thread uninitialization procedure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModuleInternal Unit [529]

7.1.2.30.1.8 NModuleSetUninit Function

Sets the uninitialization procedure for the specified module.

C++

```
NResult N_API NModuleSetUninit(HNModule hModule, NModuleUninitProc pValue);
```

Parameters

Parameters	Description
HNModule hModule	[in] Handle to the NModule [516].
NModuleUninitProc pValue	[in] Pointer to the uninitialization procedure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NModuleInternal Unit [529]

7.1.2.30.1.9 NModuleUnregister Function

Unregisters a module.

C++

```
NResult N_API NModuleUnregister(HNModule * phModule);
```

Parameters

Parameters	Description
HNModule * phModule	[out] Pointer to a handle to the unregistered NModule [516].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NModuleInternal Unit [529]

7.1.2.31 NMonitor Unit

Lightweight, cross platform primitives for critical section.

See Also

NMutex [534] for alternative implementation.

Functions

	Name	Description
	NMonitorDispose [532]	Releases all resources used by a monitor.
	NMonitorEnter [533]	Waits for ownership of the specified monitor. The function returns when the calling thread is granted ownership (after thread currently owning the monitor calls NMonitorExit [533]).
	NMonitorExit [533]	Releases ownership of the specified monitor.
	NMonitorInit [533]	Initializes a monitor.
	NMonitorTryEnter [534]	Gets ownership of the specified monitor without waiting if possible.

Macros

Name	Description
N_MONITOR_SIZE [534]	Defines the monitor size.

7.1.2.31.1 Functions

7.1.2.31.1.1 NMonitorDispose Function

Releases all resources used by a monitor.

C++

```
void N_API NMonitorDispose(NMonitor * pMonitor);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NMonitor * pMonitor	[in] Pointer to the monitor.

Remarks

The given monitor should have been initialized and must be not owned by any thread.

Module

NMonitor Unit [532]

7.1.2.31.1.2 NMonitorEnter Function

Waits for ownership of the specified monitor. The function returns when the calling thread is granted ownership (after thread currently owning the monitor calls NMonitorExit [533]).

C++

```
void N_API NMonitorEnter(NMonitor * pMonitor);
```

Parameters

Parameters	Description
NMonitor * pMonitor	[in] Pointer to the monitor.

Notes

- The monitor primitive is recursive, i.e. the owning thread can call NMonitorEnter multiple times. It is however required that NMonitorExit [533] is called for each successful NMonitorEnter/NMonitorTryEnter [534].

Module

NMonitor Unit [532]

7.1.2.31.1.3 NMonitorExit Function

Releases ownership of the specified monitor.

C++

```
void N_API NMonitorExit(NMonitor * pMonitor);
```

Parameters

Parameters	Description
NMonitor * pMonitor	[in] Pointer to the monitor.

Notes

- The monitor primitive is recursive, i.e. the owning thread must call NMonitorExit for each successfull NMonitorEnter [533]/NMonitorTryEnter [534] call.
- It is an error to call NMonitorExit on not owned monitor, and will result in undefined behaviour. Consider using NMutex [534] if checking against such misuses is needed.

Module

NMonitor Unit [532]

7.1.2.31.1.4 NMonitorInit Function

Initializes a monitor.

C++

```
void N_API NMonitorInit(NMonitor * pMonitor);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NMonitor * pMonitor	[in,out] Pointer to the monitor that will be initialized.

Module

NMonitor Unit [532]

7.1.2.31.1.5 NMonitorTryEnter Function

Gets ownership of the specified monitor without waiting if possible.

C++

```
NBool N_API NMonitorTryEnter(NMonitor * pMonitor);
```

Parameters

Parameters	Description
NMonitor * pMonitor	[in] Pointer to the monitor.

Returns

NFalse [1041] is returned if another thread currently owns the monitor. The non zero value is returned when the ownership was successfully acquired.

Notes

- The monitor primitive is recursive, i.e. the owning thread can successfully call NMonitorTryEnter multiple times. It is however required that NMonitorExit [533] is called for each successful NMonitorEnter [533]/NMonitorTryEnter.

Module

NMonitor Unit [532]

7.1.2.31.2 Macros

7.1.2.31.2.1 N_MONITOR_SIZE Macro

Defines the monitor size.

C++

```
#define N_MONITOR_SIZE 24
```

Module

NMonitor Unit [532]

7.1.2.32 NMutex Unit

Cross platform implementation of critical sections.

See Also

- NWaitObject [1123] for NMutex functionality shared with other synchronization objects.
- NMonitor for alternative, more lightweight implementation.

Functions

	Name	Description
	NMutexCreate [535]	Creates and initializes a mutex object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NMutexCreateFromOSHandle [535]	Creates and initializes a mutex object with an OS handle.
	NMutexGetOSHandle [535]	Gets an OS handle to the specified mutex.
	NMutexRelease [536]	Releases ownership of the specified mutex.

7.1.2.32.1 Functions

7.1.2.32.1.1 NMutexCreate Function

Creates and initializes a mutex object.

C++

```
NResult N_API NMutexCreate(NBool initiallyOwned, HNMutex * phMutex);
```

Parameters

Parameters	Description
NBool initiallyOwned	[in] Should the mutex be initially owned by creating thread.
HNMutex * phMutex	[out] Pointer to the variable that will receive the handle of the created object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This is usual recursive mutex. It is owned by at most single thread at any point of time. The mutex can be acquired by NWaitObjectWaitFor [1123]/NWaitObjectTimedWaitFor and released by NMutexRelease [536] functions.

Module

NMutex Unit [534]

7.1.2.32.1.2 NMutexCreateFromOSHandle Function

Creates and initializes a mutex object with an OS handle.

C++

```
NResult N_API NMutexCreateFromOSHandle(NHandle handle, NBool ownsHandle, HNMutex * phMutex);
```

Parameters

Parameters	Description
NHandle handle	[in] A file handle for the file that the current NMutex [534] will encapsulate.
NBool ownsHandle	[in] NTrue [1064] if the file handle will be owned by this NMutex [534]; otherwise, NFalse [1041].
HNMutex * phMutex	[out] Pointer to the variable that will receive the handle of the created object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMutex Unit [534]

7.1.2.32.1.3 NMutexGetOSHandle Function

Gets an OS handle to the specified mutex.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NMutexGetOSHandle(HNMutex hMutex, NHandle * phValue);
```

Parameters

Parameters	Description
HNMutex hMutex	[in] Handle to NMutex [534] object.
NHandle * phValue	[out] Pointer to the OS handle.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMutex Unit [534]

7.1.2.32.1.4 NMutexRelease Function

Releases ownership of the specified mutex.

C++

```
NResult N_API NMutexRelease(HNMutex hMutex);
```

Parameters

Parameters	Description
HNMutex hMutex	[in] Handle of the mutex.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

- The NMutex [534] is derived from NWaitObject [1123]. The NWaitObjectWaitFor [1123] or NWaitObjectTimedWaitFor should be used with the mutex handle to obtain ownership of the mutex.
- The NMutex [534] is recursive, i.e. the owning thread must call NMutexRelease for each successfull NWaitObjectTimedWaitFor or NWaitObjectWaitFor [1123] call.
- It is an error to call NMutexRelease on not owned mutex, the misuse will be checked and reported to the caller.

Module

NMutex Unit [534]

7.1.2.33 NNoDeprecate Unit

7.1.2.34 NObject Unit

Provides functionality for retrieving information about specified object.

Files

Name	Description
NObject.h [566]	Header file for NObject module. Provides functionality for retrieving information about specified object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NObjectAddDisposed [539]	Adds a "Disposed" callback to NObject.
	NObjectAddDisposedCallback [539]	Adds a "Disposed" callback to NObject.
	NObjectAddOwnerChanged [539]	Adds an "Owner changed" callback to NObject.
	NObjectAddOwnerChangedCallback [540]	Adds an "Owner changed" callback to NObject.
	NObjectAddPropertyChanged [540]	Adds a "Property changed" callback to NObject.
	NObjectAddPropertyChangedCallback [540]	Adds a "Property changed" callback to NObject.
	NObjectClone [541]	Creates a copy of specified object.
	NObjectCompareTo [541]	Compares the specified object with another specified object and returns an indication of their relative values.
	NObjectCopyProperties [542]	Copies property values from one object to another.
	NObjectEquals [542]	Compares the objects.
	NObjectFree [542]	Deletes the object specified by handle. After the object is deleted the specified handle is no longer valid.
	NObjectFreeArray [543]	Deletes array of objects specified by handles. After the objects is deleted the specified object handles are no longer valid.
	NObjectFreeElements [543]	Deletes objects specified by handles in array. After the objects is deleted the specified object handles are no longer valid.
	NObjectGet [543]	Retrieves the value of NObject.
	NObjectGetArray [543]	Retrieves an array of values of an array of objects.
	NObjectGetConcurrent [544]	Retrieves the value of NObject. This function is thread-safe.
	NObjectGetFlags [544]	Gets any flags of specified object.
	NObjectGetHashCode [545]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	NObjectGetOwner [545]	Gets owner of specified object.
	NObjectGetOwnerEx [545]	Gets owner of specified object.
	NObjectGetParameterWithPartEx [546]	Retrieves value of the parameter specified by Id.
	NObjectGetParameterWithPartExN [546]	Retrieves value of the parameter specified by Id.
	NObjectGetPropertyN [547]	Retrieves the value of the specified property of the specified object.
	NObjectGetPropertyNN [547]	Retrieves the value of the specified property of the specified object.
	NObjectGetPropertyP [548]	Retrieves the value of the specified property of the specified object.
	NObjectGetSize [548]	Retrieves the size of specified object.
	NObjectGetSizeMany [549]	Retrieves the size of the array of objects.
	NObjectGetType [549]	Retrieves type of specified object.
	NObjectRef [549]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	NObjectRemoveDisposed [550]	Removes the "Disposed" callback from NObject.
	NObjectRemoveDisposedCallback [550]	Removes the "Disposed" callback from NObject.
	NObjectRemoveOwnerChanged [550]	Removes the "Owner changed" callback from NObject.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NObjectRemoveOwnerChangedCallback [551]	Removes the "Owner changed" callback from NObject.
	NObjectRemovePropertyChanged [551]	Removes the "Property changed" callback from NObject.
	NObjectRemovePropertyChangedCallback [551]	Removes the "Property changed" callback from NObject.
	NObjectReset [552]	Sets default values for all parameters of the specified object.
	NObjectResetProperty [552]	Sets default values for the specified property of the specified object.
	NObjectSaveManyToMemoryDst [552]	Saves an array of objects to a memory location with a specified size.
	NObjectSaveManyToMemoryDstN [553]	Saves an array of objects to a specified memory location.
	NObjectSaveManyToMemoryN [553]	Saves an array of objects to a memory location.
	NObjectSaveManyToStream [554]	Saves a NObject array in a stream.
	NObjectSaveToMemoryDst [554]	Saves an objects to a memory location with specified size.
	NObjectSaveToMemoryDstN [554]	Save an object to a specified memory location.
	NObjectSaveToMemoryN [555]	Saves an object to a memory location.
	NObjectSaveToStream [555]	Saves NObject to a stream.
	NObjectSet [556]	Set the value of NObject.
	NObjectSetConcurrent [556]	Set the value of NObject. This function is tread-safe.
	NObjectSetFlags [556]	Sets the flags for the specified object.
	NObjectSetParameterWithPartEx [557]	Sets value of the parameter specified by Id.
	NObjectSetParameterWithPartExN [557]	Sets value of the parameter specified by Id.
	NObjectSetPropertyN [558]	Sets the specified property of the specified object to the specified value.
	NObjectSetPropertyNN [558]	Sets the specified property of the specified object to the specified value.
	NObjectSetPropertyP [559]	Sets the specified property of the specified object to the specified value.
	NObjectToString [559]	Returns string representation of the object.
	NObjectUnref [560]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
	NObjectUnrefArray [560]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
	NObjectUnrefElements [561]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

Macros

Name	Description
N_OBJECT_REF_RET [561]	Parameter used when object should be returned with incremented reference count. This parameter can be used with all *Add* functions accepting flags.
NObjectCopyParameters [561]	Copies parameter values from one object to another.
NObjectGetParameter [562]	Retrieves value of the parameter specified by Id.
NObjectGetParameterEx [562]	Retrieves value of the parameter specified by Id.
NObjectGetParameterWithPart [563]	Retrieves value of the parameter specified by Id.
NObjectGetPropertyP [563]	Picks a Unicode (W) or ASCII (A) version of the NObjectGetPropertyP function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObjectResetProperty [564]	Picks a Unicode (W) or ASCII (A) version of the NObjectResetProperty function.
NObjectSetParameter [564]	Sets value of the specified parameter of the specified object.
NObjectSetParameterEx [564]	Sets value of the specified parameter of the specified object.
NObjectSetParameterWithPart [565]	Sets value of the parameter specified by Id.
NObjectSetPropertyP [566]	Picks a Unicode (W) or ASCII (A) version of the NObjectSetPropertyP function.
NObjectToString [566]	Picks a Unicode (W) or ASCII (A) version of the NObjectToString function.

7.1.2.34.1 Functions

7.1.2.34.1.1 NObjectAddDisposed Function

Adds a "Disposed" callback to NObject [536].

C++

```
NResult N_API NObjectAddDisposed(HNObject hObject, HNCallback hCallback);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
HNCallback hCallback	[in] The callback to be added.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.2 NObjectAddDisposedCallback Function

Adds a "Disposed" callback to NObject [536].

C++

```
NResult N_API NObjectAddDisposedCallback(HNObject hObject, NObjectCallback pCallback, void
* pParam);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
NObjectCallback pCallback	[in] The callback to be added.
void * pParam	[in/out] A pointer to data provided for the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.3 NObjectAddOwnerChanged Function

Adds an "Owner changed" callback to NObject [536].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NObjectAddOwnerChanged(HNObject hObject, HNCallback hCallback);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
HNCallback hCallback	[in] The callback to be added.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.4 NObjectAddOwnerChangedCallback Function

Adds an "Owner changed" callback to NObject [536].

C++

```
NResult N_API NObjectAddOwnerChangedCallback(HNObject hObject, NObjectCallback pCallback,
void * pParam);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
NObjectCallback pCallback	[in] The callback to be added.
void * pParam	[in/out] A pointer to data provided for the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.5 NObjectAddPropertyChanged Function

Adds a "Property changed" callback to NObject [536].

C++

```
NResult N_API NObjectAddPropertyChanged(HNObject hObject, HNCallback hCallback);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
HNCallback hCallback	[in] The callback to be added.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.6 NObjectAddPropertyChangedCallback Function

Adds a "Property changed" callback to NObject [536].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NObjectAddPropertyChangedCallback(HNObject hObject,
NObjectPropertyChangedCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
NObjectPropertyChangedCallback pCallback	[in] The callback to be added.
void * pParam	[in/out] A pointer to data provided for the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.7 NObjectClone Function

Creates a copy of specified object.

C++

```
NResult N_API NObjectClone(HNObject hObject, HNObject * phClonedObject);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to object to clone.
HNObject * phClonedObject	[out] Pointer to cloned object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

Cloned object always should be disposed by calling NObjectFree [542] function.

Module

NObject Unit [536]

7.1.2.34.1.8 NObjectCompareTo Function

Compares the specified object with another specified object and returns an indication of their relative values.

C++

```
NResult N_API NObjectCompareTo(HNObject hObject, HNObject hOtherObject, NInt * pResult);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the first object.
HNObject hOtherObject	[in] Handle to the second object.
NInt * pResult	[out] Pointer to value indicating the relative values of the objects.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NObject Unit [536]

7.1.2.34.1.9 NObjectCopyProperties Function

Copies property values from one object to another.

C++

```
NResult N_API NObjectCopyProperties(HNObject hDstObject, HNObject hSrcObject);
```

Parameters

Parameters	Description
HNObject hDstObject	[in] Destination object. Source object will be copied to this object.
HNObject hSrcObject	[in] Source object to copy.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise, one of the following error codes [17] is returned.

Module

NObject Unit [536]

7.1.2.34.1.10 NObjectEquals Function

Compares the objects.

C++

```
NResult N_API NObjectEquals(HNObject hObject, HNObject hOtherObject, NBool * pResult);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to first object to be compared.
HNObject hOtherObject	[in] Handle to second object to be compared.
NBool * pResult	[out] pointer to NBool [1004] variable where result of comparison will be saved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.11 NObjectFree Function

Deletes the object specified by handle. After the object is deleted the specified handle is no longer valid.

C++

```
void N_API NObjectFree(HNObject hObject);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to object to be deleted.

Module

NObject Unit [536]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.34.1.12 NObjectFreeArray Function

Deletes array of objects specified by handles. After the objects is deleted the specified object handles are no longer valid.

C++

```
void N_API NObjectFreeArray(HNObject * arhObjects, NInt objectCount);
```

Parameters

Parameters	Description
HNObject * arhObjects	[in] Handle to objects array to be deleted.
NInt objectCount	[in] Objects array count.

Module

NObject Unit [536]

7.1.2.34.1.13 NObjectFreeElements Function

Deletes objects specified by handles in array. After the objects is deleted the specified object handles are no longer valid.

C++

```
void N_API NObjectFreeElements(HNObject * arhObjects, NInt objectCount);
```

Parameters

Parameters	Description
HNObject * arhObjects	[in] Handle to objects array to be deleted.
NInt objectCount	[in] Objects array count.

Remarks

This function does not free array where objects are saved. If you need to dispose all array use NObjectFreeArray [543].

Module

NObject Unit [536]

7.1.2.34.1.14 NObjectGet Function

Retrieves the value of NObject [536].

C++

```
NResult N_API NObjectGet(HNObject hObject, HNObject * phValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
HNObject * phValue	[out] A pointer to the value of NObject [536].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.15 NObjectGetArray Function

Retrieves an array of values of an array of objects.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NObjectGetArray(const HNObject * arhObjects, NInt objectCount, HNObject * *
parhValues, NInt * pValueCount);
```

Parameters

Parameters	Description
const HNObject * arhObjects	[in] A pointer to an array of NObject [536] handles.
NInt objectCount	[in] NObject [536] array count.
HNObject * * parhValues	[out] A pointer to a pointer to an array of values of NObject [536] array.
NInt * pValueCount	[out] *parhValues* values count.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.16 NObjectGetConcurrent Function

Retrieves the value of NObject [536]. This function is thread-safe.

C++

```
NResult N_API NObjectGetConcurrent(volatile HNObject * phVariable, HNObject * phValue);
```

Parameters

Parameters	Description
volatile HNObject * phVariable	[in] A handle to NObject [536].
HNObject * phValue	[out] A pointer to the value of NObject [536].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.17 NObjectGetFlags Function

Gets any flags of specified object.

C++

```
NResult N_API NObjectGetFlags(HNObject hObject, NUInt * pValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to object to retrieve its flags.
NUInt * pValue	[out] Pointer to memory block which contains flags from specified object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.34.1.18 NObjectGetHashCode Function

Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.

C++

```
NResult N_API NObjectGetHashCode(HNObject hObject, NInt * pValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to object to retrieve the value.
NInt * pValue	[out] Pointer to variable where value will be stored.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.19 NObjectGetOwner Function

Gets owner of specified object.

C++

```
NResult N_API NObjectGetOwner(HNObject hObject, HNObject * phValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to object to retrieve it's owner.
HNObject * phValue	[out] Pointer to owner object.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise, one of the following error codes [17] is returned.

Module

NObject Unit [536]

7.1.2.34.1.20 NObjectGetOwnerEx Function

Gets owner of specified object.

C++

```
NResult N_API NObjectGetOwnerEx(HNObject hObject, HNObject * phValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to object to retrieve it's owner.
HNObject * phValue	[out] Pointer to owner object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.34.1.21 NObjectGetParameterWithPartEx Function

Retrieves value of the parameter specified by Id.

C++

```
NResult N_API NObjectGetParameterWithPartEx(HNObject hObject, NUShort partId, NUInt
parameterId, NInt typeId, void * pValue, NSizeType valueSize);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to object to retrieve it's parameter value.
NUShort partId	[in] Specifies the part of the specified NObject [536].
NUInt parameterId	[in] Identifier of the parameter to retrieve.
NInt typeId	[in] Parameter type Id (one of N_TYPE_* values).
void * pValue	[in] Pointer to variable that receives parameter value.
NSizeType valueSize	[in] Size of parameter value. Typically sizeof(value).

Returns

If the function succeeds and *parameterId* specifies a N_TYPE_STRING [578] type parameter, and *pValue [870]* is NULL [1075], the return value is length of the string (not including the NULL [1075]-terminator) *pValue [870]* should point to. If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].

NResult [1007] value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

- Parameters allow to alter configurable behavior of the units. This function can be used to check the active settings. The list of configurable parameters supported by the unit can be found in *Macros* subsection of it's API reference (for example see NLExtractor parameters [119]). Note that parameter defines can be distinguished from other defines by the **P** character in the name (like in `NLEP_MIN_IOD` parameter for minimal inter ocular distance [124] of NLExtractor [104]).
- To learn the type of the parameter pass value obtained with NParameterMakeId [578] macro using N_PC_TYPE_ID [576] code and the parameter id via *parameterId* parameter and pointer to NInt [1006] that will receive one of N_TYPE_XXX via *pValue [870]* parameter. *hObject* can be NULL [1075] in this case.

Notes

It is recommended to use function NObjectGetParameterEx [562] instead of this one.

Module

NObject Unit [536]

7.1.2.34.1.22 NObjectGetParameterWithPartExN Function

Retrieves value of the parameter specified by Id.

C++

```
NResult N_API NObjectGetParameterWithPartExN(HNObject hObject, NUShort partId, NUInt
parameterId, HNValue * phValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to object to retrieve it's parameter value.
NUShort partId	[in] Specifies the part of the specified NObject [536].
NUInt parameterId	[in] Identifier of the parameter to retrieve.
HNValue * phValue	[in] Pointer to variable that receives parameter value.

Returns

If the function succeeds and *parameterId* specifies a N_TYPE_STRING [578] type parameter, and *pValue [870]* is NULL

Copyright © 2009-2014 Neurotechnology. All rights reserved.

[1075], the return value is length of the string (not including the NULL [1075]-terminator) *pValue [870]* should point to. If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].

NResult [1007] value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

- Parameters allow to alter configurable behavior of the units. This function can be used to check the active settings. The list of configurable parameters supported by the unit can be found in *Macros* subsection of it's API reference (for example see NLExtractor parameters [119]). Note that parameter defines can be distinguished from other defines by the **P** character in the name (like in `NLEP_MIN_IOD` parameter for minimal inter ocular distance [124] of NLExtractor [104]).
- To learn the type of the parameter pass value obtained with NParameterMakeId [578] macro using N_PC_TYPE_ID [576] code and the parameter id via *parameterId* parameter and pointer to NInt [1006] that will receive one of N_TYPE_XXX via *pValue [870]* parameter. *hObject* can be NULL [1075] in this case.

Module

NObject Unit [536]

7.1.2.34.1.23 NObjectGetPropertyN Function

Retrieves the value of the specified property of the specified object.

C++

```
NResult N_API NObjectGetPropertyN(HNObject hObject, HNString hName, HNValue * phValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the object.
HNString hName	[in] Handle to the name of the property.
HNValue * phValue	[in] Pointer to a handle to the new value of the property.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.24 NObjectGetPropertyNN Function

Retrieves the value of the specified property of the specified object.

C++

```
NResult N_API NObjectGetPropertyNN(HNObject hObject, HNString hName, HNType hValueType,
NAttributes attributes, void * arValues, NSizeType valuesSize, NInt valuesLength, NBool *
pHasValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the object.
HNString hName	[in] Handle to the name of the property.
HNType hValueType	[in] Handle to the type of the value.
NAttributes attributes	[in] Property attributes.
void * arValues	[out] Pointer to an array of values.
NSizeType valuesSize	[in] Size of the value.
NInt valuesLength	[in] Length of the value array.
NBool * pHasValue	[out] NTrue [1064] if the array contains a value; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.25 NObjectGetPropertyP Function

Retrieves the value of the specified property of the specified object.

C++

```
NResult N_API NObjectGetPropertyP(HNObject hObject, const NChar * szName, NTypeOfProc
pValueTypeOf, NAttributes attributes, void * arValues, NSizeType valuesSize, NInt
valuesLength, NBool * pHasValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the object.
const NChar * szName	[in] Pointer to the name of the property.
NTypeOfProc pValueTypeOf	[in] Pointer to the process that retrieves the type of the value.
NAttributes attributes	[in] Property attributes.
void * arValues	[out] Pointer to an array of values.
NSizeType valuesSize	[in] Size of the value.
NInt valuesLength	[in] Length of the value array.
NBool * pHasValue	[out] NTrue [1064] if the array contains a value; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.26 NObjectGetSize Function

Retrieves the size of specified object.

C++

```
NResult N_API NObjectGetSize(HNObject hObject, NUInt flags, NSizeType * pValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
NUInt flags	[in] Flags specific to NObject [536].
NSizeType * pValue	[out] A pointer to the size value of the NObject [536].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.34.1.27 NObjectGetSizeMany Function

Retrieves the size of the array of objects.

C++

```
NResult N_API NObjectGetSizeMany(const HNObject * arhObjects, NInt objectCount, NUInt
flags, NSizeType * pValue);
```

Parameters

Parameters	Description
const HNObject * arhObjects	[in] A pointer to an array of NObject [536] handles.
NInt objectCount	[in] NObject [536] array count.
NUInt flags	[in] Flags specific to NObject [536] array.
NSizeType * pValue	[out] A pointer to the size of value of NObject [536] array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.28 NObjectGetType Function

Retrieves type of specified object.

C++

```
NResult N_API NObjectGetType(HNObject hObject, HNType * phValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to object to retrieve it's type.
pValue	[out] Object's type value.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise, one of the following error codes [17] is returned.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NObject Unit [536]

7.1.2.34.1.29 NObjectRef Function

Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.

C++

```
NResult N_API NObjectRef(HNObject hObject);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to an object to increment its reference count.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

NResult [1007] > 0 is new reference count.

Module

NObject Unit [536]

7.1.2.34.1.30 NObjectRemoveDisposed Function

Removes the "Disposed" callback from NObject [536].

C++

```
NResult N_API NObjectRemoveDisposed(HNObject hObject, HNCallback hCallback);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
HNCallback hCallback	[in] The callback to be removed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.31 NObjectRemoveDisposedCallback Function

Removes the "Disposed" callback from NObject [536].

C++

```
NResult N_API NObjectRemoveDisposedCallback(HNObject hObject, NObjectCallback pCallback,
void * pParam);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
NObjectCallback pCallback	[in] The callback to be removed.
void * pParam	[in/out] A pointer to data provided for the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.32 NObjectRemoveOwnerChanged Function

Removes the "Owner changed" callback from NObject [536].

C++

```
NResult N_API NObjectRemoveOwnerChanged(HNObject hObject, HNCallback hCallback);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
HNCallback hCallback	[in] The callback to be removed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.33 NObjectRemoveOwnerChangedCallback Function

Removes the "Owner changed" callback from NObject [536].

C++

```
NResult N_API NObjectRemoveOwnerChangedCallback(HNObject hObject, NObjectCallback
pCallback, void * pParam);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
NObjectCallback pCallback	[in] The callback to be removed.
void * pParam	[in/out] A pointer to data provided for the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.34 NObjectRemovePropertyChanged Function

Removes the "Property changed" callback from NObject [536].

C++

```
NResult N_API NObjectRemovePropertyChanged(HNObject hObject, HNCallback hCallback);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
HNCallback hCallback	[in] The callback to be removed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.35 NObjectRemovePropertyChangedCallback Function

Removes the "Property changed" callback from NObject [536].

C++

```
NResult N_API NObjectRemovePropertyChangedCallback(HNObject hObject,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
NObjectPropertyChangedCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
NObjectPropertyChangedCallback pCallback	[in] The callback to be removed.
void * pParam	[in/out] A pointer to data provided for the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.36 NObjectReset Function

Sets default values for all parameters of the specified object.

C++

```
NResult N_API NObjectReset(HNObject hObject);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to object to reset values.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise, one of the following error codes [17] is returned.

Module

NObject Unit [536]

7.1.2.34.1.37 NObjectResetProperty Function

Sets default values for the specified property of the specified object.

C++

```
NResult N_API NObjectResetProperty(HNObject hObject, const NChar * szName);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to object.
const NChar * szName	[in] Pointer to the name of the property to reset values.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.38 NObjectSaveManyToMemoryDst Function

Saves an array of objects to a memory location with a specified size.

C++

```
NResult N_API NObjectSaveManyToMemoryDst(const HNObject * arhObjects, NInt objectCount,
void * pBuffer, NSizeType bufferSize, NUInt flags, NSizeType * pSize);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const HNObject * arhObjects	[in] A pointer to an array of NObject [536] handles.
NInt objectCount	[in] NObject [536] array count.
void * pBuffer	[in] A pointer to the memory location.
NSizeType bufferSize	[in] The size of the memory location.
NUInt flags	[in] Flags specific to NObject [536] array.
NSizeType * pSize	[out] A pointer to the size value of the saved array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.39 NObjectSaveManyToMemoryDstN Function

Saves an array of objects to a specified memory location.

C++

```
NResult N_API NObjectSaveManyToMemoryDstN(const HNObject * arhObjects, NInt objectCount,
HNBuffer hBuffer, NUInt flags, NSizeType * pSize);
```

Parameters

Parameters	Description
const HNObject * arhObjects	[in] A pointer to an array of NObject [536] handles.
NInt objectCount	[in] NObject [536] array count.
HNBuffer hBuffer	[in] A handle to the memory location.
NUInt flags	[in] Flags specific to NObject [536] array.
NSizeType * pSize	[out] A pointer to the size value of the saved array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.40 NObjectSaveManyToMemoryN Function

Saves an array of objects to a memory location.

C++

```
NResult N_API NObjectSaveManyToMemoryN(const HNObject * arhObjects, NInt objectCount, NUInt
flags, HNBuffer * phBuffer);
```

Parameters

Parameters	Description
const HNObject * arhObjects	[in] A pointer to an array of NObject [536] handles.
NInt objectCount	[in] NObject [536] array count.
NUInt flags	[in] Flags specific to NObject [536] array.
HNBuffer * phBuffer	[out] A pointer to a memory location.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.41 NObjectSaveManyToStream Function

Saves a NObject [536] array in a stream.

C++

```
NResult N_API NObjectSaveManyToStream(const HNObject * arhObjects, NInt objectCount,
HNStream hStream, NUInt flags);
```

Parameters

Parameters	Description
const HNObject * arhObjects	[in] A pointer to an array of NObject [536] handles.
NInt objectCount	[in] NObject [536] array count.
HNStream hStream	[in] A handle to the target stream.
NUInt flags	[in] Flags specific to NObject [536] array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.42 NObjectSaveToMemoryDst Function

Saves an objects to a memory location with specified size.

C++

```
NResult N_API NObjectSaveToMemoryDst(HNObject hObject, void * pBuffer, NSizeType
bufferSize, NUInt flags, NSizeType * pSize);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
void * pBuffer	[in] A pointer to the memory location.
NSizeType bufferSize	[in] The size of the memory location.
NUInt flags	[in] Flags specific to NObject [536].
NSizeType * pSize	[out] A pointer to the size value of the saved object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.43 NObjectSaveToMemoryDstN Function

Save an object to a specified memory location.

C++

```
NResult N_API NObjectSaveToMemoryDstN(HNObject hObject, HNBuffer hBuffer, NUInt flags,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
NSizeType * pSize);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
HNBuffer hBuffer	[in] A handle to the memory location.
NUInt flags	[in] Flags specific to NObject [536].
NSizeType * pSize	[out] A pointer to the size value of the saved object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.44 NObjectSaveToMemoryN Function

Saves an object to a memory location.

C++

```
NResult N_API NObjectSaveToMemoryN(HNObject hObject, NUInt flags, HNBuffer * phBuffer);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
NUInt flags	[in] Flags specific to NObject [536].
HNBuffer * phBuffer	[out] A pointer to a memory location.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.45 NObjectSaveToStream Function

Saves NObject [536] to a stream.

C++

```
NResult N_API NObjectSaveToStream(HNObject hObject, HNStream hStream, NUInt flags);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
HNStream hStream	[in] A handle to the target stream.
NUInt flags	[in] Flags specific to NObject [536].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.34.1.46 NObjectSet Function

Set the value of NObject [536].

C++

```
NResult N_API NObjectSet(HNObject hObject, HNObject * phVariable);
```

Parameters

Parameters	Description
HNObject hObject	[in] A handle to NObject [536].
HNObject * phVariable	[in] A pointer to the value of NObject [536] to set.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.47 NObjectSetConcurrent Function

Set the value of NObject [536]. This function is tread-safe.

C++

```
NResult N_API NObjectSetConcurrent(HNObject hValue, volatile HNObject * phVariable);
```

Parameters

Parameters	Description
HNObject hValue	[in] A handle to NObject [536].
volatile HNObject * phVariable	[in] A pointer to the value of NObject [536] to set.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.48 NObjectSetFlags Function

Sets the flags for the specified object.

C++

```
NResult N_API NObjectSetFlags(HNObject hObject, NUInt value);
```

Parameters

Parameters	Description
NUInt value	[in] The flags to set.
hModule	[in] Handle to NObject [536].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.34.1.49 NObjectSetParameterWithPartEx Function

Sets value of the parameter specified by Id.

C++

```
NResult N_API NObjectSetParameterWithPartEx(HNObject hObject, NUShort partId, NUInt
parameterId, NInt typeId, const void * pValue, NSizeType valueSize);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the object. Can be NULL [1075] if setting static parameter value.
NUShort partId	[in] Specifies the part of the specified NObject [536].
NUInt parameterId	[in] Identifier of the parameter to set.
NInt typeId	[in] Parameter type Id (one of N_TYPE_* values).
const void * pValue	[in] Pointer to the parameter value to set.
NSizeType valueSize	[in] Size of parameter value. Typically sizeof(value).

Returns

If the function succeeds and *parameterId* specifies a N_TYPE_STRING [578] type parameter, and *pValue [870]* is NULL [1075], the return value is length of the string (not including the NULL [1075]-terminator) *pValue [870]* should point to. If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].

NResult [1007] value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

The supported *partId* and *parameterId* parameters list for each object is different. See handle description for each object.

- Parameters allow to alter configurable behavior of the units. This function can be used to check the active settings. The list of configurable parameters supported by the unit can be found in *Macros* subsection of it's API reference (for example see NLExtractor parameters [119]). Note that parameter defines can be distinguished from other defines by the **P** character in the name (like in `NLEP_MIN_IOD` parameter for minimal inter ocular distance [124] of NLExtractor [104]).
- To learn the type of the parameter pass value obtained with NParameterMakeId [578] macro using N_PC_TYPE_ID [576] code and the parameter id via *parameterId* parameter and pointer to NInt [1006] that will receive one of N_TYPE_XXX via *pValue [870]* parameter. *hObject* can be NULL [1075] in this case.

Notes

It is recommended to use function NObjectSetParameterEx [564] instead of this one.

Module

NObject Unit [536]

7.1.2.34.1.50 NObjectSetParameterWithPartExN Function

Sets value of the parameter specified by Id.

C++

```
NResult N_API NObjectSetParameterWithPartExN(HNObject hObject, NUShort partId, NUInt
parameterId, HNValue hValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the object. Can be NULL [1075] if setting static parameter value.
NUShort partId	[in] Specifies the part of the specified NObject [536].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt parameterId	[in] Identifier of the parameter to set.
HNValue hValue	[in] Handle to the parameter value to set.

Returns

If the function succeeds and *parameterId* specifies a N_TYPE_STRING [578] type parameter, and *pValue [870]* is NULL [1075], the return value is length of the string (not including the NULL [1075]-terminator) *pValue [870]* should point to. If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].

NResult [1007] value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

The supported *partId* and *parameterId* parameters list for each object is different. See handle description for each object.

- Parameters allow to alter configurable behavior of the units. This function can be used to check the active settings. The list of configurable parameters supported by the unit can be found in *Macros* subsection of it's API reference (for example see NLExtractor parameters [119]). Note that parameter defines can be distinguished from other defines by the **P** character in the name (like in `NLE**P**_MIN_IOD` parameter for minimal inter ocular distance [124] of NLExtractor [104]).
- To learn the type of the parameter pass value obtained with NParameterMakeId [578] macro using N_PC_TYPE_ID [576] code and the parameter id via *parameterId* parameter and pointer to NInt [1006] that will receive one of N_TYPE_XXX via *pValue [870]* parameter. *hObject* can be NULL [1075] in this case.

Module

NObject Unit [536]

7.1.2.34.1.51 NObjectSetPropertyN Function

Sets the specified property of the specified object to the specified value.

C++

```
NResult N_API NObjectSetPropertyN(HNObject hObject, HNString hName, HNValue hValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the object.
HNString hName	[in] Handle to the name of the property.
HNValue hValue	[in] Handle to the new value of the property.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.52 NObjectSetPropertyNN Function

Sets the specified property of the specified object to the specified value.

C++

```
NResult N_API NObjectSetPropertyNN(HNObject hObject, HNString hName, HNType hValueType,
NAttributes attributes, const void * arValues, NSizeType valuesSize, NInt valuesLength,
NBool hasValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNString hName	[in] Handle to the name of the property.
HNType hValueType	[in] Handle to the type of the value.
NAttributes attributes	[in] Property attributes.
const void * arValues	[in] Pointer to an array of values.
NSizeType valuesSize	[in] Size of the value.
NInt valuesLength	[in] Length of the value array.
NBool hasValue	[in] NTrue [1064] if the array contains a value; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.53 NObjectSetPropertyP Function

Sets the specified property of the specified object to the specified value.

C++

```
NResult N_API NObjectSetPropertyP(HNObject hObject, const NChar * szName, NTypeOfProc
pValueTypeOf, NAttributes attributes, const void * arValues, NSizeType valuesSize, NInt
valuesLength, NBool hasValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the object.
const NChar * szName	[in] Pointer to the name of the property.
NTypeOfProc pValueTypeOf	[in] Pointer to the process that retrieves the type of the value.
NAttributes attributes	[in] Property attributes.
const void * arValues	[in] Pointer to an array of values.
NSizeType valuesSize	[in] Size of the value.
NInt valuesLength	[in] Length of the value array.
NBool hasValue	[in] NTrue [1064] if the array contains a value; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.1.54 NObjectToString Function

Returns string representation of the object.

C++

```
NResult N_API NObjectToString(HNObject hObject, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to object to perform operation on.
const NChar * szFormat	[in] Pointer to a format string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNString * phValue	[out] Pointer to handle which contains string representation of NObject [536].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NObject Unit [536]

7.1.2.34.1.55 NObjectUnref Function

Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.

C++

```
NResult N_API NObjectUnref(HNObject hObject);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to an object to decrement its reference count.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

NResult [1007] > 0 is new reference count.

Module

NObject Unit [536]

7.1.2.34.1.56 NObjectUnrefArray Function

Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

C++

```
NResult N_API NObjectUnrefArray(HNObject * arhObjects, NInt objectCount);
```

Parameters

Parameters	Description
HNObject * arhObjects	[in] Handle to array of objects which reference count should be decremented.
NInt objectCount	[in] Objects in *arhObjects* count.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

After calling this function the memory used by *arhObjects* is deallocated.

Module

NObject Unit [536]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.34.1.57 NObjectUnrefElements Function

Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

C++

```
NResult N_API NObjectUnrefElements(HNObject * arhObjects, NInt objectCount);
```

Parameters

Parameters	Description
HNObject * arhObjects	[in] Handle to array of objects which reference count should be decremented.
NInt objectCount	[in] Objects in *arhObjects* count.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.2 Macros

7.1.2.34.2.1 N_OBJECT_REF_RET Macro

Parameter used when object should be returned with incremented reference count. This parameter can be used with all *Add* functions accepting flags.

C++

```
#define N_OBJECT_REF_RET 0x00000010
```

Module

NObject Unit [536]

7.1.2.34.2.2 NObjectCopyParameters Macro

Copies parameter values from one object to another.

C++

```
#define NObjectCopyParameters(hDstObject, hSrcObject) NObjectCopyProperties(hDstObject,
hSrcObject)
```

Parameters

Parameters	Description
hDstObject	[in] Destination object. Source object will be copied to this object.
hSrcObject	[in] Source object to copy.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise, one of the following error codes [17] is returned.

Module

NObject Unit [536]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.34.2.3 NObjectGetParameter Macro

Retrieves value of the parameter specified by Id.

C++

```
#define NObjectGetParameter(hObject, parameterId, pValue) NObjectGetParameterEx(hObject,
parameterId, -1, pValue, N_SIZE_TYPE_MAX)
```

Parameters

Parameters	Description
hObject	[in] Handle to object to retrieve it's parameter value.
parameterId	[in] Identifier of the parameter to retrieve.
pValue	[in] Pointer to variable that receives parameter value.

Returns

If the function succeeds and *parameterId* specifies a N_TYPE_STRING [578] type parameter, and *pValue [870]* is NULL [1075], the return value is length of the string (not including the NULL [1075]-terminator) *pValue [870]* should point to. If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].

NResult [1007] value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.2.4 NObjectGetParameterEx Macro

Retrieves value of the parameter specified by Id.

C++

```
#define NObjectGetParameterEx(hObject, parameterId, typeId, pValue, valueSize)
NObjectGetParameterWithPartEx(hObject, 0, parameterId, typeId, pValue, valueSize)
```

Parameters

Parameters	Description
hObject	[in] Handle to object to retrieve it's parameter value.
parameterId	[in] Identifier of the parameter to retrieve.
typeId	[in] Parameter type Id (one of N_TYPE_* values).
pValue	[in] Pointer to variable that receives parameter value.
valueSize	[in] Size of parameter value. Typically sizeof(value).

Returns

If the function succeeds and *parameterId* specifies a N_TYPE_STRING [578] type parameter, and *pValue [870]* is NULL [1075], the return value is length of the string (not including the NULL [1075]-terminator) *pValue [870]* should point to. If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].

NResult [1007] value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

- Parameters allow to alter configurable behavior of the units. This function can be used to check the active settings. The list of configurable parameters supported by the unit can be found in *Macros* subsection of it's API reference (for example see NLExtractor parameters [119]). Note that parameter defines can be distinguished from other defines by the **P** character in the name (like in `NLE**P**_MIN_IOD` parameter for minimal inter ocular distance [124] of NLExtractor [104]).
- To learn the type of the parameter pass value obtained with NParameterMakeId [578] macro using N_PC_TYPE_ID [576] code and the parameter id via *parameterId* parameter and pointer to NInt [1006] that will receive one of N_TYPE_XXX via *pValue [870]* parameter. *hObject* can be NULL [1075] in this case.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Example

Following code snippet demonstrates how to check the default template size of face extractor:

```
    NResult result = N_OK;
    HNLExtractor extractor = NULL;
    NleTemplateSize templSize;

    // Create extractor
    result = NleCreate(&extractor);
    if (NFailed(result)) printf(N_T("Failed to create extractor (result = %d)!\n"), result);

    // Set template size to large
    result = NObjectGetParameterEx(extractor, NLEP_TEMPLATE_SIZE, N_TYPE_INT, &templSize,
sizeof(templSize));
    if (NFailed(result)) printf(N_T("Failed to get NLEP_TEMPLATE_SIZE parameter (result =
%d)!\n"), result);

    switch(templSize) {
        case nletsSmall: printf(N_T("NLEP_TEMPLATE_SIZE is set to nletsSmall\n")); break;
        case nletsCompact: printf(N_T("NLEP_TEMPLATE_SIZE is set to nletsCompact\n"));
break;
        case nletsMedium: printf(N_T("NLEP_TEMPLATE_SIZE is set to nletsMedium\n")); break;
        case nletsLarge: printf(N_T("NLEP_TEMPLATE_SIZE is set to nletsLarge\n")); break;
        default: printf(N_T("Unexpected NLEP_TEMPLATE_SIZE value\n"));
    }
```

Module

NObject Unit [536]

7.1.2.34.2.5 NObjectGetParameterWithPart Macro

Retrieves value of the parameter specified by Id.

C++

```
#define NObjectGetParameterWithPart(hObject, partId, parameterId, pValue)
NObjectGetParameterWithPartEx(hObject, partId, parameterId, -1, pValue, N_SIZE_TYPE_MAX)
```

Parameters

Parameters	Description
hObject	[in] Handle to object to retrieve it's parameter value.
partId	[in] Specifies the part of the specified NObject [536].
parameterId	[in] Identifier of the parameter to retrieve.
pValue	[in] Pointer to variable that receives parameter value.

Returns

If the function succeeds and *parameterId* specifies a N_TYPE_STRING [578] type parameter, and *pValue [870]* is NULL [1075], the return value is length of the string (not including the NULL [1075]-terminator) *pValue [870]* should point to. If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].

NResult [1007] value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NObject Unit [536]

7.1.2.34.2.6 NObjectGetPropertyP Macro

Picks a Unicode (W) or ASCII (A) version of the NObjectGetPropertyP function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NObjectGetPropertyP N_FUNC_AW(NObjectGetPropertyP)
```

Module

NObject Unit [536]

7.1.2.34.2.7 NObjectResetProperty Macro

Picks a Unicode (W) or ASCII (A) version of the NObjectResetProperty function.

C++

```
#define NObjectResetProperty N_FUNC_AW(NObjectResetProperty)
```

Module

NObject Unit [536]

7.1.2.34.2.8 NObjectSetParameter Macro

Sets value of the specified parameter of the specified object.

C++

```
#define NObjectSetParameter(hObject, parameterId, pValue) NObjectSetParameterEx(hObject,
parameterId, -1, pValue, N_SIZE_TYPE_MAX)
```

Parameters

Parameters	Description
hObject	[in] Handle to the object. Can be NULL [1075] if setting static parameter value.
parameterId	[in] Identifier of the parameter to set.
pValue	[in] Pointer to the parameter value to set.

Returns

If the function succeeds and *parameterId* specifies a N_TYPE_STRING [578] type parameter, and *pValue [870]* is NULL [1075], the return value is length of the string (not including the NULL [1075]-terminator) *pValue [870]* should point to. If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].

NResult [1007] value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

- Parameters set by this function are used to configure behavior of some unit, for example inter ocular distance range accepted by NLExtractor [104]. The list of configurable parameters supported by the unit can be found in *Macros* subsection of it's API reference (for example see NLExtractor parameters [119]). Note that parameter defines can be distinguished from other defines by the **P** character in the name (like in `NLE**P**_MIN_IOD` parameter for minimal inter ocular distance [124] of NLExtractor [104]).
- To learn the type of the parameter pass value obtained with NParameterMakeId [578] macro using N_PC_TYPE_ID [576] code and the parameter id via *parameterId* parameter and pointer to NInt [1006] that will receive one of N_TYPE_XXX via *pValue [870]* parameter. *hObject* can be NULL [1075] in this case.

Module

NObject Unit [536]

7.1.2.34.2.9 NObjectSetParameterEx Macro

Sets value of the specified parameter of the specified object.

C++

```
#define NObjectSetParameterEx(hObject, parameterId, typeId, pValue, valueSize)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
NObjectSetParameterWithPartEx(hObject, 0, parameterId, typeId, pValue, valueSize)
```

Parameters

Parameters	Description
hObject	[in] Handle to the object. Can be NULL [1075] if setting static parameter value.
parameterId	[in] Identifier of the parameter to set.
typeId	[in] Parameter type Id (one of N_TYPE_* values).
pValue	[in] Pointer to the parameter value to set.
valueSize	[in] Size of parameter value. Typically sizeof(value).

Returns

If the function succeeds and *parameterId* specifies a N_TYPE_STRING [578] type parameter, and *pValue [870]* is NULL [1075], the return value is length of the string (not including the NULL [1075]-terminator) *pValue [870]* should point to. If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].

NResult [1007] value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

- Parameters set by this function are used to configure behavior of some unit, for example inter ocular distance range accepted by NLExtractor [104]. The list of configurable parameters supported by the unit can be found in *Macros* subsection of it's API reference (for example see NLExtractor parameters [119]). Note that parameter defines can be distinguished from other defines by the **P** character in the name (like in `NLE**P**_MIN_IOD` parameter for minimal inter ocular distance [124] of NLExtractor [104]).
- To learn the type of the parameter pass value obtained with NParameterMakeId [578] macro using N_PC_TYPE_ID [576] code and the parameter id via *parameterId* parameter and pointer to NInt [1006] that will receive one of N_TYPE_XXX via *pValue [870]* parameter. *hObject* can be NULL [1075] in this case.

Example

Following code snippet demonstrates how to set the template size of face extractor to large:

```
    NResult result = N_OK;
    HNLExtractor extractor = NULL;
    NleTemplateSize templSize = nletsLarge;

    // Create extractor
    result = NleCreate(&extractor);
    if (NFailed(result)) printErrorMsg(N_T("Failed to create extractor (result = %d)!"),
result);

    // Set template size to large
    result = NObjectSetParameterEx(extractor, NLEP_TEMPLATE_SIZE, N_TYPE_INT, &templSize,
sizeof(templSize));
    if (NFailed(result)) printErrorMsg(N_T("Failed to set NLEP_TEMPLATE_SIZE parameter
(result = %d)!"), result);

    // Now NfeExtract(extractor, ...) function returned templates will be more precise
```

Module

NObject Unit [536]

7.1.2.34.2.10 NObjectSetParameterWithPart Macro

Sets value of the parameter specified by Id.

C++

```
#define NObjectSetParameterWithPart(hObject, partId, parameterId, pValue)
NObjectSetParameterWithPartEx(hObject, partId, parameterId, -1, pValue, N_SIZE_TYPE_MAX)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
hObject	[in] Handle to the object. Can be NULL [1075] if setting static parameter value.
partId	[in] Specifies the part of the specified NObject [536].
parameterId	[in] Identifier of the parameter to set.
pValue	[in] Pointer to the parameter value to set.

Returns

If the function succeeds and *parameterId* specifies a N_TYPE_STRING [578] type parameter, and *pValue [870]* is NULL [1075], the return value is length of the string (not including the NULL [1075]-terminator) *pValue [870]* should point to. If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].

NResult [1007] value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

The supported *partId* and *parameterId* parameters list for each object is different. See handle description for each object.

Module

NObject Unit [536]

7.1.2.34.2.11 NObjectSetPropertyP Macro

Picks a Unicode (W) or ASCII (A) version of the NObjectSetPropertyP function.

C++

```
#define NObjectSetPropertyP N_FUNC_AW(NObjectSetPropertyP)
```

Module

NObject Unit [536]

7.1.2.34.2.12 NObjectToString Macro

Picks a Unicode (W) or ASCII (A) version of the NObjectToString function.

C++

```
#define NObjectToString N_FUNC_AW(NObjectToString)
```

Module

NObject Unit [536]

7.1.2.34.3 Files

7.1.2.34.3.1 NObject.h

Header file for NObject [536] module. Provides functionality for retrieving information about specified object.

Functions

	Name	Description
	NObjectAddDisposed [539]	Adds a "Disposed" callback to NObject [536].
	NObjectAddDisposedCallback [539]	Adds a "Disposed" callback to NObject [536].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NObjectAddOwnerChanged [539]	Adds an "Owner changed" callback to NObject [536].
	NObjectAddOwnerChangedCallback [540]	Adds an "Owner changed" callback to NObject [536].
	NObjectAddPropertyChanged [540]	Adds a "Property changed" callback to NObject [536].
	NObjectAddPropertyChangedCallback [540]	Adds a "Property changed" callback to NObject [536].
	NObjectClone [541]	Creates a copy of specified object.
	NObjectCompareTo [541]	Compares the specified object with another specified object and returns an indication of their relative values.
	NObjectCopyProperties [542]	Copies property values from one object to another.
	NObjectEquals [542]	Compares the objects.
	NObjectFree [542]	Deletes the object specified by handle. After the object is deleted the specified handle is no longer valid.
	NObjectFreeArray [543]	Deletes array of objects specified by handles. After the objects is deleted the specified object handles are no longer valid.
	NObjectFreeElements [543]	Deletes objects specified by handles in array. After the objects is deleted the specified object handles are no longer valid.
	NObjectGet [543]	Retrieves the value of NObject [536].
	NObjectGetArray [543]	Retrieves an array of values of an array of objects.
	NObjectGetConcurrent [544]	Retrieves the value of NObject [536]. This function is thread-safe.
	NObjectGetFlags [544]	Gets any flags of specified object.
	NObjectGetHashCode [545]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	NObjectGetOwner [545]	Gets owner of specified object.
	NObjectGetOwnerEx [545]	Gets owner of specified object.
	NObjectGetParameterWithPartEx [546]	Retrieves value of the parameter specified by Id.
	NObjectGetParameterWithPartExN [546]	Retrieves value of the parameter specified by Id.
	NObjectGetPropertyN [547]	Retrieves the value of the specified property of the specified object.
	NObjectGetPropertyNN [547]	Retrieves the value of the specified property of the specified object.
	NObjectGetPropertyP [548]	Retrieves the value of the specified property of the specified object.
	NObjectGetSize [548]	Retrieves the size of specified object.
	NObjectGetSizeMany [549]	Retrieves the size of the array of objects.
	NObjectGetType [549]	Retrieves type of specified object.
	NObjectRef [549]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	NObjectRemoveDisposed [550]	Removes the "Disposed" callback from NObject [536].
	NObjectRemoveDisposedCallback [550]	Removes the "Disposed" callback from NObject [536].
	NObjectRemoveOwnerChanged [550]	Removes the "Owner changed" callback from NObject [536].
	NObjectRemoveOwnerChangedCallback [551]	Removes the "Owner changed" callback from NObject [536].
	NObjectRemovePropertyChanged [551]	Removes the "Property changed" callback from NObject [536].
	NObjectRemovePropertyChangedCallback [551]	Removes the "Property changed" callback from NObject [536].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NObjectReset [552]	Sets default values for all parameters of the specified object.
	NObjectResetProperty [552]	Sets default values for the specified property of the specified object.
	NObjectSaveManyToMemoryDst [552]	Saves an array of objects to a memory location with a specified size.
	NObjectSaveManyToMemoryDstN [553]	Saves an array of objects to a specified memory location.
	NObjectSaveManyToMemoryN [553]	Saves an array of objects to a memory location.
	NObjectSaveManyToStream [554]	Saves a NObject [536] array in a stream.
	NObjectSaveToMemoryDst [554]	Saves an objects to a memory location with specified size.
	NObjectSaveToMemoryDstN [554]	Save an object to a specified memory location.
	NObjectSaveToMemoryN [555]	Saves an object to a memory location.
	NObjectSaveToStream [555]	Saves NObject [536] to a stream.
	NObjectSet [556]	Set the value of NObject [536].
	NObjectSetConcurrent [556]	Set the value of NObject [536]. This function is tread-safe.
	NObjectSetFlags [556]	Sets the flags for the specified object.
	NObjectSetParameterWithPartEx [557]	Sets value of the parameter specified by Id.
	NObjectSetParameterWithPartExN [557]	Sets value of the parameter specified by Id.
	NObjectSetPropertyN [558]	Sets the specified property of the specified object to the specified value.
	NObjectSetPropertyNN [558]	Sets the specified property of the specified object to the specified value.
	NObjectSetPropertyP [559]	Sets the specified property of the specified object to the specified value.
	NObjectToString [559]	Returns string representation of the object.
	NObjectUnref [560]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
	NObjectUnrefArray [560]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
	NObjectUnrefElements [561]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

Macros

Name	Description
N_OBJECT_REF_RET [561]	Parameter used when object should be returned with incremented reference count. This parameter can be used with all *Add* functions accepting flags.
NObjectCopyParameters [561]	Copies parameter values from one object to another.
NObjectGetParameter [562]	Retrieves value of the parameter specified by Id.
NObjectGetParameterEx [562]	Retrieves value of the parameter specified by Id.
NObjectGetParameterWithPart [563]	Retrieves value of the parameter specified by Id.
NObjectGetPropertyP [563]	Picks a Unicode (W) or ASCII (A) version of the NObjectGetPropertyP function.
NObjectResetProperty [564]	Picks a Unicode (W) or ASCII (A) version of the NObjectResetProperty function.
NObjectSetParameter [564]	Sets value of the specified parameter of the specified object.
NObjectSetParameterEx [564]	Sets value of the specified parameter of the specified object.
NObjectSetParameterWithPart [565]	Sets value of the parameter specified by Id.
NObjectSetPropertyP [566]	Picks a Unicode (W) or ASCII (A) version of the NObjectSetPropertyP function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObjectToString [566]	Picks a Unicode (W) or ASCII (A) version of the NObjectToString function.

Module

NObject Unit [536]

7.1.2.35 NParameterBag Unit

A helper class that aids in implementation of PropertyBag for ParameterDescriptor class.

Files

Name	Description
NParameterBag.h [575]	Header file for the NParameterBag unit.

Functions

	Name	Description
	NParameterBagApplyPropertyBag [569]	Applies the specified parameter bag to the specified property bag.
	NParameterBagContains [570]	Checks whether the specified parameter bag contains the parameter with specified key.
	NParameterBagCreate [570]	Creates a new parameter bag from an array of parameters.
	NParameterBagGet [571]	Retrieves the parameter specified by the key.
	NParameterBagGetAt [571]	Retrieves the parameter specified by an index.
	NParameterBagGetCount [571]	Retrieves the number of parameters in the specified parameter bag.
	NParameterBagGetKey [572]	Retrieves the key to the parameter specified by index.
	NParameterBagGetValue [572]	Retrieves the value of the parameter specified by an index.
	NParameterBagSet [572]	Sets the value of the parameter specified by the key.
	NParameterBagSetValue [573]	Sets the value of the parameter specified by an index.
	NParameterBagToPropertyBag [573]	Converts a parameter bag to a property bag.
	NParameterBagTryGet [573]	Retrieves the parameter specified by the key. Returns a Boolean value indicating whether the operation succeeded.

Macros

Name	Description
NParameterBagContains [574]	Picks a Unicode (W) or ASCII (A) version of the NParameterBagContains function.
NParameterBagGet [574]	Picks a Unicode (W) or ASCII (A) version of the NParameterBagGet function.
NParameterBagSet [574]	Picks a Unicode (W) or ASCII (A) version of the NParameterBag function.
NParameterBagTryGet [574]	Picks a Unicode (W) or ASCII (A) version of the NParameterBagTryGet function.

7.1.2.35.1 Functions

7.1.2.35.1.1 NParameterBagApplyPropertyBag Function

Applies the specified parameter bag to the specified property bag.

C++

```
NResult N_API NParameterBagApplyPropertyBag(HNParameterBag hParameterBag, HNPropertyBag
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
hValue, NBool ignoreUknown);
```

Parameters

Parameters	Description
HNParameterBag hParameterBag	[in] Handle to the parameter bag.
HNPropertyBag hValue	[in] Handle to the property bag.
NBool ignoreUknown	[in] NTrue [1064] if unknown values should be ignored; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterBag Unit [569]

7.1.2.35.1.2 NParameterBagContains Function

Checks whether the specified parameter bag contains the parameter with specified key.

C++

```
NResult N_API NParameterBagContains(HNParameterBag hParameterBag, const NChar * szKey,
NBool * pResult);
```

Parameters

Parameters	Description
HNParameterBag hParameterBag	[in] Handle to the parameter bag.
const NChar * szKey	[in] Pointer to the key to the parameter.
NBool * pResult	[out] NTrue [1064] if the bag contains the specified parameter; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterBag Unit [569]

7.1.2.35.1.3 NParameterBagCreate Function

Creates a new parameter bag from an array of parameters.

C++

```
NResult N_API NParameterBagCreate(const HNParameterDescriptor * arhParameters, NInt
parameterCount, HNParameterBag * phParameterBag);
```

Parameters

Parameters	Description
const HNParameterDescriptor * arhParameters	[in] Pointer to an array of parameter descriptor handles.
NInt parameterCount	[in] Number of parameters.
HNParameterBag * phParameterBag	[out] Pointer to a handle to the new parameter bag.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterBag Unit [569]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.35.1.4 NParameterBagGet Function

Retrieves the parameter specified by the key.

C++

```
NResult N_API NParameterBagGet(HNParameterBag hParameterBag, const NChar * szKey, HNValue *
phValue);
```

Parameters

Parameters	Description
HNParameterBag hParameterBag	[in] Handle to the parameter bag.
const NChar * szKey	[in] Pointer to the parameter key.
HNValue * phValue	[out] Pointer to a handle to the parameter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterBag Unit [569]

7.1.2.35.1.5 NParameterBagGetAt Function

Retrieves the parameter specified by an index.

C++

```
NResult N_API NParameterBagGetAt(HNParameterBag hParameterBag, NInt index, struct
NNameValuePair_ * pValue);
```

Parameters

Parameters	Description
HNParameterBag hParameterBag	[in] Handle to the parameter bag.
NInt index	[in] Index of the parameter.
struct NNameValuePair_ * pValue	[out] Pointer to a handle to the parameter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterBag Unit [569]

7.1.2.35.1.6 NParameterBagGetCount Function

Retrieves the number of parameters in the specified parameter bag.

C++

```
NResult N_API NParameterBagGetCount(HNParameterBag hParameterBag, NInt * pValue);
```

Parameters

Parameters	Description
HNParameterBag hParameterBag	[in] Handle to the parameter bag.
NInt * pValue	[out] Pointer to the number of parameters in the bag.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NParameterBag Unit [569]

7.1.2.35.1.7 NParameterBagGetKey Function

Retrieves the key to the parameter specified by index.

C++

```
NResult N_API NParameterBagGetKey(HNParameterBag hParameterBag, NInt index, HNString *
phValue);
```

Parameters

Parameters	Description
HNParameterBag hParameterBag	[in] Handle to the parameter bag.
NInt index	[in] Index of the parameter.
HNString * phValue	[out] Pointer to a handle to the string representing the key of the parameter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterBag Unit [569]

7.1.2.35.1.8 NParameterBagGetValue Function

Retrieves the value of the parameter specified by an index.

C++

```
NResult N_API NParameterBagGetValue(HNParameterBag hParameterBag, NInt index, HNValue *
phValue);
```

Parameters

Parameters	Description
HNParameterBag hParameterBag	[in] Handle to the parameter bag.
NInt index	[in] Index of the parameter whose value is to be retrieved.
HNValue * phValue	[out] Pointer to a handle to the value of the parameter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterBag Unit [569]

7.1.2.35.1.9 NParameterBagSet Function

Sets the value of the parameter specified by the key.

C++

```
NResult N_API NParameterBagSet(HNParameterBag hParameterBag, const NChar * szKey, HNValue
hValue);
```

Parameters

Parameters	Description
HNParameterBag hParameterBag	[in] Handle to the parameter bag.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const NChar * szKey	[in] Pointer to the parameter key.
HNValue hValue	[in] Handle to the new parameter value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterBag Unit [569]

7.1.2.35.1.10 NParameterBagSetValue Function

Sets the value of the parameter specified by an index.

C++

```
NResult N_API NParameterBagSetValue(HNParameterBag hParameterBag, NInt index, HNValue
hValue);
```

Parameters

Parameters	Description
HNParameterBag hParameterBag	[in] Handle to the parameter bag.
NInt index	[in] Index of the parameter.
HNValue hValue	[in] Handle to the new parameter value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterBag Unit [569]

7.1.2.35.1.11 NParameterBagToPropertyBag Function

Converts a parameter bag to a property bag.

C++

```
NResult N_API NParameterBagToPropertyBag(HNParameterBag hParameterBag, HNPropertyBag *
phValue);
```

Parameters

Parameters	Description
HNParameterBag hParameterBag	[in] Handle to the parameter bag.
HNPropertyBag * phValue	[out] Pointer to a the handle to the converted property bag.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterBag Unit [569]

7.1.2.35.1.12 NParameterBagTryGet Function

Retrieves the parameter specified by the key. Returns a Boolean value indicating whether the operation succeeded.

C++

```
NResult N_API NParameterBagTryGet(HNParameterBag hParameterBag, const NChar * szKey,
HNValue * phValue, NBool * pResult);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNParameterBag hParameterBag	[in] Handle to the parameter bag.
const NChar * szKey	[in] Pointer to the parameter key.
HNValue * phValue	[out] Pointer to a handle to the parameter.
NBool * pResult	[out] NTrue [1064] if the operation was successful, else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterBag Unit [569]

7.1.2.35.2 Macros

7.1.2.35.2.1 NParameterBagContains Macro

Picks a Unicode (W) or ASCII (A) version of the NParameterBagContains function.

C++

```
#define NParameterBagContains N_FUNC_AW(NParameterBagContains)
```

Module

NParameterBag Unit [569]

7.1.2.35.2.2 NParameterBagGet Macro

Picks a Unicode (W) or ASCII (A) version of the NParameterBagGet function.

C++

```
#define NParameterBagGet N_FUNC_AW(NParameterBagGet)
```

Module

NParameterBag Unit [569]

7.1.2.35.2.3 NParameterBagSet Macro

Picks a Unicode (W) or ASCII (A) version of the NParameterBag [569] function.

C++

```
#define NParameterBagSet N_FUNC_AW(NParameterBag)
```

Module

NParameterBag Unit [569]

7.1.2.35.2.4 NParameterBagTryGet Macro

Picks a Unicode (W) or ASCII (A) version of the NParameterBagTryGet function.

C++

```
#define NParameterBagTryGet N_FUNC_AW(NParameterBagTryGet)
```

Module

NParameterBag Unit [569]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.35.3 Files

7.1.2.35.3.1 NParameterBag.h

Header file for the NParameterBag [569] unit.

Functions

	Name	Description
	NParameterBagApplyPropertyBag [569]	Applies the specified parameter bag to the specified property bag.
	NParameterBagContains [570]	Checks whether the specified parameter bag contains the parameter with specified key.
	NParameterBagCreate [570]	Creates a new parameter bag from an array of parameters.
	NParameterBagGet [571]	Retrieves the parameter specified by the key.
	NParameterBagGetAt [571]	Retrieves the parameter specified by an index.
	NParameterBagGetCount [571]	Retrieves the number of parameters in the specified parameter bag.
	NParameterBagGetKey [572]	Retrieves the key to the parameter specified by index.
	NParameterBagGetValue [572]	Retrieves the value of the parameter specified by an index.
	NParameterBagSet [572]	Sets the value of the parameter specified by the key.
	NParameterBagSetValue [573]	Sets the value of the parameter specified by an index.
	NParameterBagToPropertyBag [573]	Converts a parameter bag to a property bag.
	NParameterBagTryGet [573]	Retrieves the parameter specified by the key. Returns a Boolean value indicating whether the operation succeeded.

Macros

Name	Description
NParameterBagContains [574]	Picks a Unicode (W) or ASCII (A) version of the NParameterBagContains function.
NParameterBagGet [574]	Picks a Unicode (W) or ASCII (A) version of the NParameterBagGet function.
NParameterBagSet [574]	Picks a Unicode (W) or ASCII (A) version of the NParameterBag [569] function.
NParameterBagTryGet [574]	Picks a Unicode (W) or ASCII (A) version of the NParameterBagTryGet function.

Module

NParameterBag Unit [569]

7.1.2.36 NParameters Unit

Provides functionality for working with parameters for Neurotechnology components.

Files

Name	Description
NParameters.h [579]	Header file for the NParameters module. Provides functionality for working with parameters for Neurotechnology components.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Macros

Name	Description
N_TYPE_BOOL [576]	Specifies that parameter type is NBool [1004].
N_PC_TYPE_ID [576]	Specifies that type id (NInt [1006] value, one of N_TYPE_XXX) of the parameter should be retrieved.
N_TYPE_BYTE [576]	Specifies that parameter type is NByte [1005].
N_TYPE_DOUBLE [576]	Specifies that parameter type is NDouble [1005].
N_TYPE_FLOAT [577]	Specifies that parameter type is NFloat [1005].
N_TYPE_INT [577]	Specifies that parameter type is NInt [1006].
N_TYPE_LONG [577]	Specifies that parameter type is NLong [1007].
N_TYPE_SBYTE [577]	Specifies that parameter type is NSByte [1008].
N_TYPE_SHORT [577]	Specifies that parameter type is NShort [1008].
N_TYPE_STRING [578]	Specifies that parameter type is null-terminated string of NChar [1005].
N_TYPE_UINT [578]	Specifies that parameter type is NUInt [1009].
N_TYPE_ULONG [578]	Specifies that parameter type is NULong [1010].
N_TYPE_USHORT [578]	Specifies that parameter type is NUShort [1010].
NParameterMakeId [578]	Makes parameter id.

7.1.2.36.1 Macros

7.1.2.36.1.1 N_TYPE_BOOL Macro

Specifies that parameter type is NBool [1004].

C++

```
#define N_TYPE_BOOL 10
```

Module

NParameters Unit [575]

7.1.2.36.1.2 N_PC_TYPE_ID Macro

Specifies that type id (NInt [1006] value, one of N_TYPE_XXX) of the parameter should be retrieved.

C++

```
#define N_PC_TYPE_ID 1
```

Module

NParameters Unit [575]

7.1.2.36.1.3 N_TYPE_BYTE Macro

Specifies that parameter type is NByte [1005].

C++

```
#define N_TYPE_BYTE 1
```

Module

NParameters Unit [575]

7.1.2.36.1.4 N_TYPE_DOUBLE Macro

Specifies that parameter type is NDouble [1005].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_TYPE_DOUBLE 31
```

Module

NParameters Unit [575]

7.1.2.36.1.5 N_TYPE_FLOAT Macro

Specifies that parameter type is NFloat [1005].

C++

```
#define N_TYPE_FLOAT 30
```

Module

NParameters Unit [575]

7.1.2.36.1.6 N_TYPE_INT Macro

Specifies that parameter type is NInt [1006].

C++

```
#define N_TYPE_INT 6
```

Module

NParameters Unit [575]

7.1.2.36.1.7 N_TYPE_LONG Macro

Specifies that parameter type is NLong [1007].

C++

```
#define N_TYPE_LONG 8
```

Module

NParameters Unit [575]

7.1.2.36.1.8 N_TYPE_SBYTE Macro

Specifies that parameter type is NSByte [1008].

C++

```
#define N_TYPE_SBYTE 2
```

Module

NParameters Unit [575]

7.1.2.36.1.9 N_TYPE_SHORT Macro

Specifies that parameter type is NShort [1008].

C++

```
#define N_TYPE_SHORT 4
```

Module

NParameters Unit [575]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.36.1.10 N_TYPE_STRING Macro

Specifies that parameter type is null-terminated string of NChar [1005].

C++

```
#define N_TYPE_STRING 100
```

Module

NParameters Unit [575]

7.1.2.36.1.11 N_TYPE_UINT Macro

Specifies that parameter type is NUInt [1009].

C++

```
#define N_TYPE_UINT 5
```

Module

NParameters Unit [575]

7.1.2.36.1.12 N_TYPE_ULONG Macro

Specifies that parameter type is NULong [1010].

C++

```
#define N_TYPE_ULONG 7
```

Module

NParameters Unit [575]

7.1.2.36.1.13 N_TYPE_USHORT Macro

Specifies that parameter type is NUShort [1010].

C++

```
#define N_TYPE_USHORT 3
```

Module

NParameters Unit [575]

7.1.2.36.1.14 NParameterMakeId Macro

Makes parameter id.

C++

```
#define NParameterMakeId(code, index, id) ((NUInt)(((code) << 24) | (((index) & 0xFF) <<
16) | ((id) & 0xFFFF)))
```

Parameters

Parameters	Description
code	One of N_PC_XXX.
index	Reserved, must be zero.
id	One of the parameter ids provided by a Neurotechnology module.

Module

NParameters Unit [575]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.36.2 Files

7.1.2.36.2.1 NParameters.h

Header file for the NParameters [575] module. Provides functionality for working with parameters for Neurotechnology components.

Macros

Name	Description
N_PC_TYPE_ID [576]	Specifies that type id (NInt [1006] value, one of N_TYPE_XXX) of the parameter should be retrieved.
N_TYPE_BOOL [576]	Specifies that parameter type is NBool [1004].
N_TYPE_BYTE [576]	Specifies that parameter type is NByte [1005].
N_TYPE_DOUBLE [576]	Specifies that parameter type is NDouble [1005].
N_TYPE_FLOAT [577]	Specifies that parameter type is NFloat [1005].
N_TYPE_INT [577]	Specifies that parameter type is NInt [1006].
N_TYPE_LONG [577]	Specifies that parameter type is NLong [1007].
N_TYPE_SBYTE [577]	Specifies that parameter type is NSByte [1008].
N_TYPE_SHORT [577]	Specifies that parameter type is NShort [1008].
N_TYPE_STRING [578]	Specifies that parameter type is null-terminated string of NChar [1005].
N_TYPE_UINT [578]	Specifies that parameter type is NUInt [1009].
N_TYPE_ULONG [578]	Specifies that parameter type is NULong [1010].
N_TYPE_USHORT [578]	Specifies that parameter type is NUShort [1010].
NParameterMakeId [578]	Makes parameter id.

Module

NParameters Unit [575]

7.1.2.37 NParameterDescriptor Unit

Provides functions for getting information about parameter.

Files

Name	Description
NParameterDescriptor.h [582]	Header file for the NParameterDescriptor module.

Functions

	Name	Description
	NParameterDescriptorGetDefaultValue [580]	Retrieves the default value of the specified parameter descriptor.
	NParameterDescriptorGetFormat [580]	Retrieves the format value of the specified parameter descriptor.
	NParameterDescriptorGetMaxValue [580]	Retrieves the maximum value of the specified parameter descriptor.
	NParameterDescriptorGetMinValue [581]	Retrieves the minimum value of the specified parameter descriptor.
	NParameterDescriptorGetParameterType [581]	Retrieves the parameter type of the specified parameter descriptor.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NParameterDescriptorGetStdValue [581]	Retrieves the specified standard value of the specified parameter descriptor.
	NParameterDescriptorGetStdValueCount [582]	Retrieves the number of standard parameter descriptor values.

7.1.2.37.1 Functions

7.1.2.37.1.1 NParameterDescriptorGetDefaultValue Function

Retrieves the default value of the specified parameter descriptor.

C++

```
NResult N_API NParameterDescriptorGetDefaultValue(HNParameterDescriptor
hParameterDescriptor, HNValue * phValue);
```

Parameters

Parameters	Description
HNParameterDescriptor hParameterDescriptor	[in] Handle to the parameter.
HNValue * phValue	[out] Pointer to a handle to the default value of the parameter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterDescriptor Unit [579]

7.1.2.37.1.2 NParameterDescriptorGetFormat Function

Retrieves the format value of the specified parameter descriptor.

C++

```
NResult N_API NParameterDescriptorGetFormat(HNParameterDescriptor hParameterDescriptor,
HNString * phValue);
```

Parameters

Parameters	Description
HNParameterDescriptor hParameterDescriptor	[in] Handle to the parameter descriptor.
HNString * phValue	[out] Pointer to a handle to a string representing the format of the parameter descriptor.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterDescriptor Unit [579]

7.1.2.37.1.3 NParameterDescriptorGetMaxValue Function

Retrieves the maximum value of the specified parameter descriptor.

C++

```
NResult N_API NParameterDescriptorGetMaxValue(HNParameterDescriptor hParameterDescriptor,
HNValue * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNParameterDescriptor hParameterDescriptor	[in] Handle to the parameter descriptor.
HNValue * phValue	[out] Pointer to a handle to the maximum value of the parameter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterDescriptor Unit [579]

7.1.2.37.1.4 NParameterDescriptorGetMinValue Function

Retrieves the minimum value of the specified parameter descriptor.

C++

```
NResult N_API NParameterDescriptorGetMinValue(HNParameterDescriptor hParameterDescriptor,
HNValue * phValue);
```

Parameters

Parameters	Description
HNParameterDescriptor hParameterDescriptor	[in] Handle to the parameter descriptor.
HNValue * phValue	[out] Pointer to a handle to the minimum value of the parameter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterDescriptor Unit [579]

7.1.2.37.1.5 NParameterDescriptorGetParameterType Function

Retrieves the parameter type of the specified parameter descriptor.

C++

```
NResult N_API NParameterDescriptorGetParameterType(HNParameterDescriptor
hParameterDescriptor, HNType * phValue);
```

Parameters

Parameters	Description
HNParameterDescriptor hParameterDescriptor	[in] Handle to the parameter descriptor.
HNType * phValue	[out] Pointer to a handle to the parameter type.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterDescriptor Unit [579]

7.1.2.37.1.6 NParameterDescriptorGetStdValue Function

Retrieves the specified standard value of the specified parameter descriptor.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NParameterDescriptorGetStdValue(HNParameterDescriptor hParameterDescriptor,
NInt index, struct NNameValuePair_ * pValue);
```

Parameters

Parameters	Description
HNParameterDescriptor hParameterDescriptor	[in] Handle to the parameter descriptor.
NInt index	[in] Index of the standard value.
struct NNameValuePair_ * pValue	[out] Pointer to the standard value of the parameter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterDescriptor Unit [579]

7.1.2.37.1.7 NParameterDescriptorGetStdValueCount Function

Retrieves the number of standard parameter descriptor values.

C++

```
NResult N_API NParameterDescriptorGetStdValueCount(HNParameterDescriptor
hParameterDescriptor, NInt * pValue);
```

Parameters

Parameters	Description
HNParameterDescriptor hParameterDescriptor	[in] Handle to the parameter descriptor.
NInt * pValue	[out] Number of standard parameter descriptor values.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NParameterDescriptor Unit [579]

7.1.2.37.2 Files

7.1.2.37.2.1 NParameterDescriptor.h

Header file for the NParameterDescriptor [579] module.

Functions

	Name	Description
	NParameterDescriptorGetDefaultValue [580]	Retrieves the default value of the specified parameter descriptor.
	NParameterDescriptorGetFormat [580]	Retrieves the format value of the specified parameter descriptor.
	NParameterDescriptorGetMaxValue [580]	Retrieves the maximum value of the specified parameter descriptor.
	NParameterDescriptorGetMinValue [581]	Retrieves the minimum value of the specified parameter descriptor.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NParameterDescriptorGetParameterType [581]	Retrieves the parameter type of the specified parameter descriptor.
	NParameterDescriptorGetStdValue [581]	Retrieves the specified standard value of the specified parameter descriptor.
	NParameterDescriptorGetStdValueCount [582]	Retrieves the number of standard parameter descriptor values.

Module

NParameterDescriptor Unit [579]

7.1.2.38 NPath Unit

Performs operations on String instances that contain file or directory path information. These operations are performed in a cross-platform manner.

Files

Name	Description
NPath.h [593]	Header file for the NPath unit.

Functions

	Name	Description
	NPathChangeExtension [584]	Changes the extension of a path string.
	NPathChangeExtensionPN [585]	Changes the extension of a path string.
	NPathCheck [585]	Checks whether the specified path is valid and throws an exception if the path is not valid.
	NPathCheckN [585]	Checks whether the specified path is valid and throws an exception if the path is not valid.
	NPathCombine [586]	Combines two strings into a path.
	NPathCombineP1N [586]	Combines two strings into a path.
	NPathCombineP2N [586]	Combines two strings into a path.
	NPathGetDirectoryName [587]	Returns the directory information for the specified path string.
	NPathGetExtension [587]	Returns the extension of the specified path string.
	NPathGetFileName [588]	Returns the file name and extension of the specified path string.
	NPathGetFileNameWithoutExtension [588]	Returns the file name of the specified path string without the extension.
	NPathGetTempFileName [588]	Creates a uniquely named, zero-byte temporary file on disk and returns the full path of that file.
	NPathGetTempPath [589]	Returns the path of the current user's temporary folder.
	NPathHasExtension [589]	Determines whether a path includes a file name extension.
	NPathIsRooted [589]	Gets a value indicating whether the specified path string contains a root.

Macros

Name	Description
N_MAX_FILE_NAME [590]	Specifies the maximum file name length.
N_PATH_ALT_DIRECTORY_SEPARATOR [590]	Picks a Unicode (W) or ANSCII (A) version of the N_PATH_PATH_SEPARATOR [590] macro which specifies an alternate character used to separate directory levels in a path string that reflects a hierarchical file system organization.

7

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_PATH_DIRECTORY_SEPARATOR [590]	Picks a Unicode (W) or ANSCII (A) version of the N_PATH_PATH_SEPARATOR [590] macro which specifies a character used to separate directory levels in a path string that reflects a hierarchical file system organization.
N_PATH_IS_CASE_SENSITIVE [590]	Specifies whether the path is case sensitive.
N_PATH_PATH_SEPARATOR [590]	Picks a Unicode (W) or ANSCII (A) version of the N_PATH_PATH_SEPARATOR macro which specifies a separator character used to separate path strings in environment variables.
N_PATH_VOLUME_SEPARATOR [591]	Picks a Unicode (W) or ANSCII (A) version of the N_PATH_PATH_SEPARATOR [590] macro which specifies a volume separator character.
NPathChangeExtension [591]	Picks a Unicode (W) or ASCII (A) version of the NPathChangeExtension function.
NPathChangeExtensionPN [591]	Picks a Unicode (W) or ASCII (A) version of the NPathChangeExtensionPN function.
NPathCheck [591]	Picks a Unicode (W) or ASCII (A) version of the NPathCheck function.
NPathCombine [591]	Picks a Unicode (W) or ASCII (A) version of the NPathCombine function.
NPathCombineP1N [591]	Picks a Unicode (W) or ASCII (A) version of the NPathCombineP1N function.
NPathCombineP2N [592]	Picks a Unicode (W) or ASCII (A) version of the NPathCombineP2N function.
NPathGetDirectoryName [592]	Picks a Unicode (W) or ASCII (A) version of the NPathGetDirectoryName function.
NPathGetExtension [592]	Picks a Unicode (W) or ASCII (A) version of the NPathGetExtension function.
NPathGetFileName [592]	Picks a Unicode (W) or ASCII (A) version of the NPathGetFileName function.
NPathGetFileNameWithoutExtension [592]	Picks a Unicode (W) or ASCII (A) version of the NPathGetFileNameWithoutExtension function.
NPathHasExtension [593]	Picks a Unicode (W) or ASCII (A) version of the NPathHasExtension function.
NPathIsRooted [593]	Picks a Unicode (W) or ASCII (A) version of the NPathIsRooted function.

7.1.2.38.1 Functions

7.1.2.38.1.1 NPathChangeExtension Function

Changes the extension of a path string.

C++

```
NResult N_API NPathChangeExtension(const NChar * szPath, const NChar * szExtension,
HNString * phValue);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path string.
const NChar * szExtension	[in] Pointer to the new extension (with or without a leading period).
HNString * phValue	[out] Pointer to a handle to the modified path information.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NPath Unit [583]

7.1.2.38.1.2 NPathChangeExtensionPN Function

Changes the extension of a path string.

C++

```
NResult N_API NPathChangeExtensionPN(HNString hPath, const NChar * szExtension, HNString *
phValue);
```

Parameters

Parameters	Description
HNString hPath	[in] Handle to the path string.
const NChar * szExtension	[in] Pointer to the new extension (with or without a leading period).
HNString * phValue	[out] Pointer to a handle to the modified path information.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPath Unit [583]

7.1.2.38.1.3 NPathCheck Function

Checks whether the specified path is valid and throws an exception if the path is not valid.

C++

```
NResult N_API NPathCheck(const NChar * szValue);
```

Parameters

Parameters	Description
const NChar * szValue	[in] Pointer to path to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPath Unit [583]

7.1.2.38.1.4 NPathCheckN Function

Checks whether the specified path is valid and throws an exception if the path is not valid.

C++

```
NResult N_API NPathCheckN(HNString hValue);
```

Parameters

Parameters	Description
HNString hValue	[in] Handle to path to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NPath Unit [583]

7.1.2.38.1.5 NPathCombine Function

Combines two strings into a path.

C++

```
NResult N_API NPathCombine(const NChar * szPath1, const NChar * szPath2, HNString *
phValue);
```

Parameters

Parameters	Description
const NChar * szPath1	[in] Pointer to the first path to combine.
const NChar * szPath2	[in] Pointer to the second path to combine.
HNString * phValue	[out] Pointer to a handle to the combined paths.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If one of the specified paths is a zero-length string, this method returns the other path. If szPath2 contains an absolute path, this method returns szPath2.

Module

NPath Unit [583]

7.1.2.38.1.6 NPathCombineP1N Function

Combines two strings into a path.

C++

```
NResult N_API NPathCombineP1N(HNString hPath1, const NChar * szPath2, HNString * phValue);
```

Parameters

Parameters	Description
HNString hPath1	[in] Handle to the first path to combine.
const NChar * szPath2	[in] Pointer to the second path to combine.
HNString * phValue	[out] Pointer to a handle to the combined paths.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If one of the specified paths is a zero-length string, this method returns the other path. If szPath2 contains an absolute path, this method returns szPath2.

Module

NPath Unit [583]

7.1.2.38.1.7 NPathCombineP2N Function

Combines two strings into a path.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NPathCombineP2N(const NChar * szPath1, HNString hPath2, HNString * phValue);
```

Parameters

Parameters	Description
const NChar * szPath1	[in] Pointer to the first path to combine.
HNString hPath2	[in] Handle to the second path to combine.
HNString * phValue	[out] Pointer to a handle to the combined paths.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If one of the specified paths is a zero-length string, this method returns the other path. If szPath2 contains an absolute path, this method returns szPath2.

Module

NPath Unit [583]

7.1.2.38.1.8 NPathGetDirectoryName Function

Returns the directory information for the specified path string.

C++

```
NResult N_API NPathGetDirectoryName(const NChar * szPath, HNString * phValue);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer to the path of a file or directory.
HNString * phValue	[out] Pointer to a handle to the directory information for path.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPath Unit [583]

7.1.2.38.1.9 NPathGetExtension Function

Returns the extension of the specified path string.

C++

```
NResult N_API NPathGetExtension(const NChar * szPath, HNString * phValue);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer the path string from which to get the extension.
HNString * phValue	[out] Pointer to a handle to the extension of the specified path (including the period ".").

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPath Unit [583]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.38.1.10 NPathGetFileName Function

Returns the file name and extension of the specified path string.

C++

```
NResult N_API NPathGetFileName(const NChar * szPath, HNString * phValue);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer the path string from which to get the file name.
HNString * phValue	[out] Pointer to a handle to the file name and extension of the specified path (including the period ".").

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPath Unit [583]

7.1.2.38.1.11 NPathGetFileNameWithoutExtension Function

Returns the file name of the specified path string without the extension.

C++

```
NResult N_API NPathGetFileNameWithoutExtension(const NChar * szPath, HNString * phValue);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer the path string from which to get the file name.
HNString * phValue	[out] Pointer to a handle to the file name.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPath Unit [583]

7.1.2.38.1.12 NPathGetTempFileName Function

Creates a uniquely named, zero-byte temporary file on disk and returns the full path of that file.

C++

```
NResult N_API NPathGetTempFileName(HNString * phFileName);
```

Parameters

Parameters	Description
HNString * phFileName	[out] Pointer to a handle to the path string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPath Unit [583]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.38.1.13 NPathGetTempPath Function

Returns the path of the current user's temporary folder.

C++

```
NResult N_API NPathGetTempPath(HNString * phPath);
```

Parameters

Parameters	Description
HNString * phPath	[out] Pointer to a handle to the path string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPath Unit [583]

7.1.2.38.1.14 NPathHasExtension Function

Determines whether a path includes a file name extension.

C++

```
NResult N_API NPathHasExtension(const NChar * szPath, NBool * pValue);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer the path string to check.
NBool * pValue	[out] NTrue [1064] if the path includes a file name extension; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPath Unit [583]

7.1.2.38.1.15 NPathIsRooted Function

Gets a value indicating whether the specified path string contains a root.

C++

```
NResult N_API NPathIsRooted(const NChar * szPath, NBool * pValue);
```

Parameters

Parameters	Description
const NChar * szPath	[in] Pointer the path string to check.
NBool * pValue	[out] NTrue [1064] if the path contains a root; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPath Unit [583]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.38.2 Macros

7.1.2.38.2.1 N_MAX_FILE_NAME Macro

Specifies the maximum file name length.

C++

```
#define N_MAX_FILE_NAME 256
```

Module

NPath Unit [583]

7.1.2.38.2.2 N_PATH_ALT_DIRECTORY_SEPARATOR Macro

Picks a Unicode (W) or ANSCII (A) version of the N_PATH_PATH_SEPARATOR [590] macro which specifies an alternate character used to separate directory levels in a path string that reflects a hierarchical file system organization.

C++

```
#define N_PATH_ALT_DIRECTORY_SEPARATOR N_MACRO_AW(N_PATH_ALT_DIRECTORY_SEPARATOR)
```

Module

NPath Unit [583]

7.1.2.38.2.3 N_PATH_DIRECTORY_SEPARATOR Macro

Picks a Unicode (W) or ANSCII (A) version of the N_PATH_PATH_SEPARATOR [590] macro which specifies a character used to separate directory levels in a path string that reflects a hierarchical file system organization.

C++

```
#define N_PATH_DIRECTORY_SEPARATOR N_MACRO_AW(N_PATH_DIRECTORY_SEPARATOR)
```

Module

NPath Unit [583]

7.1.2.38.2.4 N_PATH_IS_CASE_SENSITIVE Macro

Specifies whether the path is case sensitive.

C++

```
#define N_PATH_IS_CASE_SENSITIVE NTrue
```

Module

NPath Unit [583]

7.1.2.38.2.5 N_PATH_PATH_SEPARATOR Macro

Picks a Unicode (W) or ANSCII (A) version of the N_PATH_PATH_SEPARATOR macro which specifies a separator character used to separate path strings in environment variables.

C++

```
#define N_PATH_PATH_SEPARATOR N_MACRO_AW(N_PATH_PATH_SEPARATOR)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NPath Unit [583]

7.1.2.38.2.6 N_PATH_VOLUME_SEPARATOR Macro

Picks a Unicode (W) or ANSCII (A) version of the N_PATH_PATH_SEPARATOR [590] macro which specifies a volume separator character.

C++

```
#define N_PATH_VOLUME_SEPARATOR N_MACRO_AW(N_PATH_VOLUME_SEPARATOR)
```

Module

NPath Unit [583]

7.1.2.38.2.7 NPathChangeExtension Macro

Picks a Unicode (W) or ASCII (A) version of the NPathChangeExtension function.

C++

```
#define NPathChangeExtension N_FUNC_AW(NPathChangeExtension)
```

Module

NPath Unit [583]

7.1.2.38.2.8 NPathChangeExtensionPN Macro

Picks a Unicode (W) or ASCII (A) version of the NPathChangeExtensionPN function.

C++

```
#define NPathChangeExtensionPN N_FUNC_AW(NPathChangeExtensionPN)
```

Module

NPath Unit [583]

7.1.2.38.2.9 NPathCheck Macro

Picks a Unicode (W) or ASCII (A) version of the NPathCheck function.

C++

```
#define NPathCheck N_FUNC_AW(NPathCheck)
```

Module

NPath Unit [583]

7.1.2.38.2.10 NPathCombine Macro

Picks a Unicode (W) or ASCII (A) version of the NPathCombine function.

C++

```
#define NPathCombine N_FUNC_AW(NPathCombine)
```

Module

NPath Unit [583]

7.1.2.38.2.11 NPathCombineP1N Macro

Picks a Unicode (W) or ASCII (A) version of the NPathCombineP1N function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPathCombineP1N N_FUNC_AW(NPathCombineP1N)
```

Module

NPath Unit [583]

7.1.2.38.2.12 NPathCombineP2N Macro

Picks a Unicode (W) or ASCII (A) version of the NPathCombineP2N function.

C++

```
#define NPathCombineP2N N_FUNC_AW(NPathCombineP2N)
```

Module

NPath Unit [583]

7.1.2.38.2.13 NPathGetDirectoryName Macro

Picks a Unicode (W) or ASCII (A) version of the NPathGetDirectoryName function.

C++

```
#define NPathGetDirectoryName N_FUNC_AW(NPathGetDirectoryName)
```

Module

NPath Unit [583]

7.1.2.38.2.14 NPathGetExtension Macro

Picks a Unicode (W) or ASCII (A) version of the NPathGetExtension function.

C++

```
#define NPathGetExtension N_FUNC_AW(NPathGetExtension)
```

Module

NPath Unit [583]

7.1.2.38.2.15 NPathGetFileName Macro

Picks a Unicode (W) or ASCII (A) version of the NPathGetFileName function.

C++

```
#define NPathGetFileName N_FUNC_AW(NPathGetFileName)
```

Module

NPath Unit [583]

7.1.2.38.2.16 NPathGetFileNameWithoutExtension Macro

Picks a Unicode (W) or ASCII (A) version of the NPathGetFileNameWithoutExtension function.

C++

```
#define NPathGetFileNameWithoutExtension N_FUNC_AW(NPathGetFileNameWithoutExtension)
```

Module

NPath Unit [583]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.38.2.17 NPathHasExtension Macro

Picks a Unicode (W) or ASCII (A) version of the NPathHasExtension function.

C++

```
#define NPathHasExtension N_FUNC_AW(NPathHasExtension)
```

Module

NPath Unit [583]

7.1.2.38.2.18 NPathIsRooted Macro

Picks a Unicode (W) or ASCII (A) version of the NPathIsRooted function.

C++

```
#define NPathIsRooted N_FUNC_AW(NPathIsRooted)
```

Module

NPath Unit [583]

7.1.2.38.3 Files

7.1.2.38.3.1 NPath.h

Header file for the NPath [583] unit.

Functions

	Name	Description
	NPathChangeExtension [584]	Changes the extension of a path string.
	NPathChangeExtensionPN [585]	Changes the extension of a path string.
	NPathCheck [585]	Checks whether the specified path is valid and throws an exception if the path is not valid.
	NPathCheckN [585]	Checks whether the specified path is valid and throws an exception if the path is not valid.
	NPathCombine [586]	Combines two strings into a path.
	NPathCombineP1N [586]	Combines two strings into a path.
	NPathCombineP2N [586]	Combines two strings into a path.
	NPathGetDirectoryName [587]	Returns the directory information for the specified path string.
	NPathGetExtension [587]	Returns the extension of the specified path string.
	NPathGetFileName [588]	Returns the file name and extension of the specified path string.
	NPathGetFileNameWithoutExtension [588]	Returns the file name of the specified path string without the extension.
	NPathGetTempFileName [588]	Creates a uniquely named, zero-byte temporary file on disk and returns the full path of that file.
	NPathGetTempPath [589]	Returns the path of the current user's temporary folder.
	NPathHasExtension [589]	Determines whether a path includes a file name extension.
	NPathIsRooted [589]	Gets a value indicating whether the specified path string contains a root.

Macros

Name	Description
N_MAX_FILE_NAME [590]	Specifies the maximum file name length.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_PATH_ALT_DIRECTORY_SEPARATOR [590]	Picks a Unicode (W) or ANSCII (A) version of the N_PATH_PATH_SEPARATOR [590] macro which specifies an alternate character used to separate directory levels in a path string that reflects a hierarchical file system organization.
N_PATH_DIRECTORY_SEPARATOR [590]	Picks a Unicode (W) or ANSCII (A) version of the N_PATH_PATH_SEPARATOR [590] macro which specifies a character used to separate directory levels in a path string that reflects a hierarchical file system organization.
N_PATH_IS_CASE_SENSITIVE [590]	Specifies whether the path is case sensitive.
N_PATH_PATH_SEPARATOR [590]	Picks a Unicode (W) or ANSCII (A) version of the N_PATH_PATH_SEPARATOR macro which specifies a separator character used to separate path strings in environment variables.
N_PATH_VOLUME_SEPARATOR [591]	Picks a Unicode (W) or ANSCII (A) version of the N_PATH_PATH_SEPARATOR [590] macro which specifies a volume separator character.
NPathChangeExtension [591]	Picks a Unicode (W) or ASCII (A) version of the NPathChangeExtension function.
NPathChangeExtensionPN [591]	Picks a Unicode (W) or ASCII (A) version of the NPathChangeExtensionPN function.
NPathCheck [591]	Picks a Unicode (W) or ASCII (A) version of the NPathCheck function.
NPathCombine [591]	Picks a Unicode (W) or ASCII (A) version of the NPathCombine function.
NPathCombineP1N [591]	Picks a Unicode (W) or ASCII (A) version of the NPathCombineP1N function.
NPathCombineP2N [592]	Picks a Unicode (W) or ASCII (A) version of the NPathCombineP2N function.
NPathGetDirectoryName [592]	Picks a Unicode (W) or ASCII (A) version of the NPathGetDirectoryName function.
NPathGetExtension [592]	Picks a Unicode (W) or ASCII (A) version of the NPathGetExtension function.
NPathGetFileName [592]	Picks a Unicode (W) or ASCII (A) version of the NPathGetFileName function.
NPathGetFileNameWithoutExtension [592]	Picks a Unicode (W) or ASCII (A) version of the NPathGetFileNameWithoutExtension function.
NPathHasExtension [593]	Picks a Unicode (W) or ASCII (A) version of the NPathHasExtension function.
NPathIsRooted [593]	Picks a Unicode (W) or ASCII (A) version of the NPathIsRooted function.

Module

NPath Unit [583]

7.1.2.39 NPlugin Unit

Plug-in management framework for use in other Neurotechnology components.

Files

Name	Description
NPlugin.h [604]	Header file for plugins management framework.

Functions

	Name	Description
	NPluginAddChanged [595]	Adds a changed callback to NPlugin.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NPluginAddChangedCallback [596]	Adds a changed callback to NPlugin.
	NPluginAddChanging [596]	Adds a changing callback to NPlugin.
	NPluginAddChangingCallback [596]	Adds a changing callback to NPlugin.
	NPluginDisable [597]	Adds plugin's name to disabled plugins collection [616].
	NPluginEnable [597]	Removes plugin's name from disabled plugins collection [616].
	NPluginGetError [597]	Returns the most recent error for the specified NPlugin.
	NPluginGetErrorCode [598]	Returns the most recent error code for the specified NPlugin.
	NPluginGetFileName [598]	Returns plug-in library name.
	NPluginGetLoadTime [598]	Returns plug-in load time.
	NPluginGetModule [599]	Gets module for specified NPlugin.
	NPluginGetPlugTime [599]	Returns plug-in plug time (nanoseconds).
	NPluginGetSelectedInterfaceVersion [600]	Returns current version of selected plugin interface.
	NPluginGetState [600]	Returns the state of specified plugin.
	NPluginPlug [600]	Plugs specified plug-in to the system.
	NPluginRemoveChanged [601]	Removes a changed callback from NPlugin.
	NPluginRemoveChangedCallback [601]	Removes a changed callback from NPlugin.
	NPluginRemoveChanging [601]	Removes a changing callback from NPlugin.
	NPluginRemoveChangingCallback [602]	Removes a changing callback from NPlugin.
	NPluginUnplug [602]	Unplugs specified plug-in from the system.

Macros

Name	Description
NPluginGetFileName [604]	Picks a Unicode (W) or ASCII (A) version of the NPluginGetFileName function.

Structs, Records, Enums

	Name	Description
	NPluginState_ [603]	Enumerates NPlugin state values.
	NPluginState [603]	Enumerates NPlugin state values.

7.1.2.39.1 Functions

7.1.2.39.1.1 NPluginAddChanged Function

Adds a changed callback to NPlugin [594].

C++

```
NResult N_API NPluginAddChanged(HNPlugin hPlugin, HNCallback hCallback);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] A handle to NPlugin [594].
HNCallback hCallback	[in] The callback to be added.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NPlugin Unit [594]

7.1.2.39.1.2 NPluginAddChangedCallback Function

Adds a changed callback to NPlugin [594].

C++

```
NResult N_API NPluginAddChangedCallback(HNPlugin hPlugin, NObjectCallback pCallback, void *
pParam);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] A handle to NPlugin [594].
NObjectCallback pCallback	[in] The callback to be added.
void * pParam	[in] A pointer to the data provided for the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPlugin Unit [594]

7.1.2.39.1.3 NPluginAddChanging Function

Adds a changing callback to NPlugin [594].

C++

```
NResult N_API NPluginAddChanging(HNPlugin hPlugin, HNCallback hCallback);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] A handle to NPlugin [594].
HNCallback hCallback	[in] The callback to be added.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPlugin Unit [594]

7.1.2.39.1.4 NPluginAddChangingCallback Function

Adds a changing callback to NPlugin [594].

C++

```
NResult N_API NPluginAddChangingCallback(HNPlugin hPlugin, NObjectCallback pCallback, void
* pParam);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] A handle to NPlugin [594].
NObjectCallback pCallback	[in] The callback to be added.
void * pParam	[in] A pointer to the data provided for the callback.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPlugin Unit [594]

7.1.2.39.1.5 NPluginDisable Function

Adds plugin's name to disabled plugins collection [616].

C++

```
NResult N_API NPluginDisable(HNPlugin hPlugin);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] Handle to NPlugin [594] to disable.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

NPlugin Unit [594]

7.1.2.39.1.6 NPluginEnable Function

Removes plugin's name from disabled plugins collection [616].

C++

```
NResult N_API NPluginEnable(HNPlugin hPlugin);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] Handle to NPlugin [594] to enable.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

NPlugin Unit [594]

7.1.2.39.1.7 NPluginGetError Function

Returns the most recent error for the specified NPlugin [594].

C++

```
NResult N_API NPluginGetError(HNPlugin hPlugin, HNError * phValue);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] Handle to NPlugin [594] to retrieve its error.
pValue	[out] Pointer to NError [398].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NPlugin Unit [594]

7.1.2.39.1.8 NPluginGetErrorCode Function

Returns the most recent error code for the specified NPlugin [594].

C++

```
NResult N_API NPluginGetErrorCode(HNPlugin hPlugin, NResult * pValue);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] Handle to NPlugin [594] to retrieve its error code.
NResult * pValue	[out] Pointer to NResult [1007] which contains error code.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

NPlugin Unit [594]

7.1.2.39.1.9 NPluginGetFileName Function

Returns plug-in library name.

C++

```
NResult N_API NPluginGetFileName(HNPlugin hPlugin, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] Handle to NPlugin [594] to retrieve its library name.
NChar * szValue	[out] Pointer to memory block which contains library file name.
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

NPlugin Unit [594]

7.1.2.39.1.10 NPluginGetLoadTime Function

Returns plug-in load time.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NPluginGetLoadTime(HNPlugin hPlugin, NTimeSpan_ * pValue);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] Handle to NPlugin [594] to retrieve its plugin load time.
NTimeSpan_ * pValue	[out] Plugin load time.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

NPlugin Unit [594]

7.1.2.39.1.11 NPluginGetModule Function

Gets module for specified NPlugin [594].

C++

```
NResult N_API NPluginGetModule(HNPlugin hPlugin, HNPluginModule * phValue);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] A handle to NPlugin [594].
HNPluginModule * phValue	[out] A pointer to NPluginModule [629].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NPlugin Unit [594]

7.1.2.39.1.12 NPluginGetPlugTime Function

Returns plug-in plug time (nanoseconds).

C++

```
NResult N_API NPluginGetPlugTime(HNPlugin hPlugin, NTimeSpan_ * pValue);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] Handle to NPlugin [594].
NTimeSpan_ * pValue	[out] Plug time in nanoseconds.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

NPlugin Unit [594]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.39.1.13 NPluginGetSelectedInterfaceVersion Function

Returns current version of selected plugin interface.

C++

```
NResult N_API NPluginGetSelectedInterfaceVersion(HNPlugin hPlugin, NVersion_ * pValue);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] Handle to plugin to retrieve its version.
NVersion_ * pValue	[out] Interface version.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Notes

NVersionRange [1011]* and NVersion [1010]* macros can be used to parse version range.

Module

NPlugin Unit [594]

7.1.2.39.1.14 NPluginGetState Function

Returns the state of specified plugin.

C++

```
NResult N_API NPluginGetState(HNPlugin hPlugin, NPluginState * pValue);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] Handle to NPlugin [594] to retrieve its state.
NPluginState * pValue	[out] Pointer to NPluginState [603] enumeration.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

NPlugin Unit [594]

7.1.2.39.1.15 NPluginPlug Function

Plugs specified plug-in to the system.

C++

```
NResult N_API NPluginPlug(HNPlugin hPlugin);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] Handle to NPlugin [594].

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Notes

The requested plugin might not be plugged into the system immediately. If there is no potential users (device managers) for provided device types, the plugin's state will be set to npsUnused [603]. The plugin will be automatically plugged later, once device manager of appropriate type is created.

Module

NPlugin Unit [594]

7.1.2.39.1.16 NPluginRemoveChanged Function

Removes a changed callback from NPlugin [594].

C++

```
NResult N_API NPluginRemoveChanged(HNPlugin hPlugin, HNCallback hCallback);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] A handle to NPlugin [594].
HNCallback hCallback	[in] The callback to be removed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPlugin Unit [594]

7.1.2.39.1.17 NPluginRemoveChangedCallback Function

Removes a changed callback from NPlugin [594].

C++

```
NResult N_API NPluginRemoveChangedCallback(HNPlugin hPlugin, NObjectCallback pCallback,
void * pParam);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] A handle to NPlugin [594].
NObjectCallback pCallback	[in] The callback to be removed.
void * pParam	[in] A pointer to the data provided for the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPlugin Unit [594]

7.1.2.39.1.18 NPluginRemoveChanging Function

Removes a changing callback from NPlugin [594].

C++

```
NResult N_API NPluginRemoveChanging(HNPlugin hPlugin, HNCallback hCallback);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNPlugin hPlugin	[in] A handle to NPlugin [594].
HNCallback hCallback	[in] The callback to be removed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPlugin Unit [594]

7.1.2.39.1.19 NPluginRemoveChangingCallback Function

Removes a changing callback from NPlugin [594].

C++

```
NResult N_API NPluginRemoveChangingCallback(HNPlugin hPlugin, NObjectCallback pCallback,
void * pParam);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] A handle to NPlugin [594].
NObjectCallback pCallback	[in] The callback to be removed.
void * pParam	[in] A pointer to the data provided for the callback.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPlugin Unit [594]

7.1.2.39.1.20 NPluginUnplug Function

Unplugs specified plug-in from the system.

C++

```
NResult N_API NPluginUnplug(HNPlugin hPlugin);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] Handle to NPlugin [594].

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Notes

- The function is not mandatory as plugin will be unplugged automatically when it becomes unused. Not calling a function would have effect that plugin would be later plugged automatically when needed.
- If the function is called the plugin's state will be set to npsUnplugged [603] and plugin will not be automatically plugged later.
- The function can be used to force unplug even if devices provided by the plugin are still in use.

Module

NPlugin Unit [594]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.39.2 Structs, Records, Enums

7.1.2.39.2.1 NPluginState Enumeration

Enumerates NPlugin [594] state values.

C++

```
typedef enum NPluginState_ {
  npsNone = 0,
  npsLoadError = 1,
  npsNotRecognized = 2,
  npsInvalidModule = 3,
  npsInterfaceTypeMismatch = 4,
  npsInterfaceVersionMismatch = 5,
  npsInvalidInterface = 6,
  npsUnplugged = 64,
  npsUnused = 65,
  npsDisabled = 66,
  npsDuplicate = 67,
  npsIncompatibleWithOtherPlugins = 68,
  npsPluggingError = 69,
  npsPlugged = 128
} NPluginState;
```

Members

Members	Description
npsNone = 0	Indicates that plugin has no state.
npsLoadError = 1	Indicates that plugin's dynamic library (dll) failed to load.
npsNotRecognized = 2	Indicates that function returning plugin module handle was not found. If the file name of plugin library is "<*Name*>`.dll`", the expected name of the function is "<*Name*>`ModuleOf`".
npsInvalidModule = 3	Indicates invalid plugin module. The plugin's ModuleOf function (see: npsNotRecognized) was found, but the returned handle was wrong.
npsInterfaceTypeMismatch = 4	Indicates that plugin's interface type differs from expected type. This means that the plugin is for other product/application. Interface type can be retrieved using NPluginModuleGetInterfaceType [631] function.
npsInterfaceVersionMismatch = 5	Indicates that plug-in interface version mismatches. This means that plugin is of the right type, but has version incompatible with versions supported by current plugin loader. Interface version can be retrieved using NPluginGetInterfaceVersions function.
npsInvalidInterface = 6	Indicates that the plugin's interface didn't pass the validation check. This happens when some mandatory functions required for plugin are missing or return wrong value.
npsUnplugged = 64	Indicates that plugin is not connected into the system. The plugin will not be used by the system unless activated with NPluginPlug [600] or NPluginManagerPlugAll [620]/NPluginManagerRefresh [620] functions.
npsUnused = 65	Indicates that plugin is activated (with NPluginPlug [600] or NPluginManagerPlugAll [620]/NPluginManagerRefresh [620]) but not currently used (not plugged into the system because there are no potential users of the plugin). The plugin will be automatically plugged once device manager of appropriate type is created (see: npsPlugged).
npsDisabled = 66	Indicates that plugin is disabled, i.e. it's name is present in disabled plugins collection [616].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

npsDuplicate = 67	Indicates that plugin is unusable because the NPluginManager [607] already contains plugin with the same name.
npsIncompatibleWithOtherPlugins = 68	Indicates that the current plugin is incompatible with other plugin in the system. Each plugin has a priority value which determines which of the incompatible plugins will be loaded. The plugin with the highest priority will be loaded, and other plugins will fail with npsIncompatibleWithOtherPlugins state.
npsPluggingError = 69	Indicates that an error occurred when plugging the plugin. NPluginGetErrorCode [598] and NPluginGetError [597] can be used to retrieve error code and error message.
npsPlugged = 128	Indicates that plugin is activated and the provided functionality can be used by the system. If all the device managers enumerating the devices provided by the current plugin are terminated, the plugin will be automatically unplugged and it's state changed to npsUnused.

Module

NPlugin Unit [594]

7.1.2.39.3 Macros

7.1.2.39.3.1 NPluginGetFileName Macro

Picks a Unicode (W) or ASCII (A) version of the NPluginGetFileName function.

C++

```
#define NPluginGetFileName N_FUNC_AW(NPluginGetFileName)
```

Module

NPlugin Unit [594]

7.1.2.39.4 Files

7.1.2.39.4.1 NPlugin.h

Header file for plugins management framework.

Enumerations

	Name	Description
	NPluginState_ [603]	Enumerates NPlugin [594] state values.
	NPluginState [603]	Enumerates NPlugin [594] state values.

Functions

	Name	Description
	NPluginAddChanged [595]	Adds a changed callback to NPlugin [594].
	NPluginAddChangedCallback [596]	Adds a changed callback to NPlugin [594].
	NPluginAddChanging [596]	Adds a changing callback to NPlugin [594].
	NPluginAddChangingCallback [596]	Adds a changing callback to NPlugin [594].
	NPluginDisable [597]	Adds plugin's name to disabled plugins collection [616].
	NPluginEnable [597]	Removes plugin's name from disabled plugins collection [616].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NPluginGetError [597]	Returns the most recent error for the specified NPlugin [594].
	NPluginGetErrorCode [598]	Returns the most recent error code for the specified NPlugin [594].
	NPluginGetFileName [598]	Returns plug-in library name.
	NPluginGetLoadTime [598]	Returns plug-in load time.
	NPluginGetModule [599]	Gets module for specified NPlugin [594].
	NPluginGetPlugTime [599]	Returns plug-in plug time (nanoseconds).
	NPluginGetSelectedInterfaceVersion [600]	Returns current version of selected plugin interface.
	NPluginGetState [600]	Returns the state of specified plugin.
	NPluginPlug [600]	Plugs specified plug-in to the system.
	NPluginRemoveChanged [601]	Removes a changed callback from NPlugin [594].
	NPluginRemoveChangedCallback [601]	Removes a changed callback from NPlugin [594].
	NPluginRemoveChanging [601]	Removes a changing callback from NPlugin [594].
	NPluginRemoveChangingCallback [602]	Removes a changing callback from NPlugin [594].
	NPluginUnplug [602]	Unplugs specified plug-in from the system.

Macros

Name	Description
NPluginGetFileName [604]	Picks a Unicode (W) or ASCII (A) version of the NPluginGetFileName function.

Module

NPlugin Unit [594]

7.1.2.40 NPluginInternal Unit

Functions

	Name	Description
	NPluginGetSelectedInterface [605]	Retrieves the selected plugin interface.

Macros

Name	Description
NPluginInterfaceVersionInfoConst [606]	Defines the plugin interface info constant.

Structs, Records, Enums

	Name	Description
	NPluginInterfaceVersionInfo_ [606]	Provides plugin interface version information.
	NPluginInterfaceVersionInfo [606]	Provides plugin interface version information.

7.1.2.40.1 Functions

7.1.2.40.1.1 NPluginGetSelectedInterface Function

Retrieves the selected plugin interface.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NPluginGetSelectedInterface(HNPlugin hPlugin, const void * * ppValue);
```

Parameters

Parameters	Description
HNPlugin hPlugin	[in] Handle to the plugin.
const void * * ppValue	[out] Double pointer to the selected interface.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginInternal Unit [605]

7.1.2.40.2 Structs, Records, Enums

7.1.2.40.2.1 NPluginInterfaceVersionInfo Structure

Provides plugin interface version information.

C++

```
typedef struct NPluginInterfaceVersionInfo_ {
  NVersionRange_ versionRange;
  const void * pInterface;
} NPluginInterfaceVersionInfo;
```

Members

Members	Description
NVersionRange_ versionRange;	Plugin interface version range.
const void * pInterface;	Plugin interface.

Module

NPluginInternal Unit [605]

7.1.2.40.3 Macros

7.1.2.40.3.1 NPluginInterfaceVersionInfoConst Macro

Defines the plugin interface info constant.

C++

```
#define NPluginInterfaceVersionInfoConst(majorVersion, minorVersionFrom, minorVersionTo,
functionTable) { NVersionRangeMake(NVersionMake(majorVersion, minorVersionFrom),
NVersionMake(majorVersion, minorVersionTo)), &(functionTable) }
```

Parameters

Parameters	Description
majorVersion	[in] Major version value.
minorVersionFrom	[in] Minor version lower value.
minorVersionTo	[in] Minor version upper value.
functionTable	[in] The function table.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginInternal Unit [605]

7.1.2.41 NPluginManager Unit

Provides functions for managing plugins.

Remarks

All NDeviceManager instances have one NPluginManager. One NPlugin [594] instance can have more than one NDevice [1159] which will be visible in one or more NDeviceManagers.

Files

Name	Description
NPluginManager.h [627]	Header file for NPluginManager. Provides functions for working with NPlugin [594].

Functions

	Name	Description
	NPluginManagerAddDisabledPlugin [609]	Disables plugin specified by name and adds to disabled plugins collection.
	NPluginManagerAddDisabledPluginsChanged [609]	Adds user specified callback for "disabled plugins collection changed" event. This callback is called when disabled plugins collection was changed.
	NPluginManagerAddDisabledPluginsChangedCallback [610]	Adds user specified callback for "disabled plugins collection changed" event. This callback is called when disabled plugins collection was changed.
	NPluginManagerAddDisabledPluginsChanging [610]	Adds user specified callback for "disabled plugins collection changing" event. This callback is called when disabled plugins collection is changing.
	NPluginManagerAddDisabledPluginsChangingCallback [611]	Adds user specified callback for "disabled plugins collection changing" event. This callback is called when disabled plugins collection is changing.
	NPluginManagerAddPlugin [611]	Loads plugin. It's dependencies are searched relative to specified directory.
	NPluginManagerAddPluginAdded [612]	Adds user specified callback for "plugin added" event. This callback is called when plugin was loaded into the system.
	NPluginManagerAddPluginAddedCallback [612]	Adds user specified callback for "plugin added" event. This callback is called when plugin was loaded into the system.
	NPluginManagerAddPluginFromFile [612]	Loads plugin from user specified file.
	NPluginManagerAddPluginsChanged [613]	Adds user specified callback for "plugins changed" event.
	NPluginManagerAddPluginsChangedCallback [613]	Adds user specified callback for "plugins changed" event.
	NPluginManagerAddPluginsChanging [613]	Adds user specified callback for "adding plugins changing" event. This callback is called when user started adding new plugins
	NPluginManagerAddPluginsChangingCallback [614]	Adds user specified callback for "adding plugins changing" event. This callback is called when user started adding new plugins

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NPluginManagerAllowsUnplug [614]	Checks whether NPluginPlug [600] and NPluginUnplug [602] functions work
	NPluginManagerClearDisabledPlugins [615]	Clears the collection of disabled plugins.
	NPluginManagerEnsurePluginSearchPerformed [615]	Trigger plugin search in plugin directory [619] if search hasn't been performed yet.
	NPluginManagerGetDisabledPlugin [616]	Gets disabled plugin specified by index.
	NPluginManagerGetDisabledPluginCount [616]	Gets disabled plugins count.
	NPluginManagerGetInstances [617]	Retrieves the instances of the plugin manager.
	NPluginManagerGetInterfaceType [617]	Returns NPluginManager interface type name.
	NPluginManagerGetInterfaceVersions [618]	Returns the range of supported interface versions by specified NPluginManager.
	NPluginManagerGetPlugin [618]	Returns NPlugin [594] specified by index.
	NPluginManagerGetPluginByName [619]	Gets plugin by specified name.
	NPluginManagerGetPluginCount [619]	Returns plugins count in specified NPluginManager.
	NPluginManagerGetPluginSearchPath [619]	Gets plugin dependencies search path.
	NPluginManagerPlugAll [620]	Plugs all enabled plugins into the system.
	NPluginManagerRefresh [620]	Refreshes plugins list (search for added/removed plugin files).
	NPluginManagerRemoveDisabledPlugin [621]	Removes plugin specified by name from disabled plugins collection.
	NPluginManagerRemoveDisabledPluginAt [621]	Removes plugin specified by index from disabled plugins collection.
	NPluginManagerRemoveDisabledPluginsChanged [621]	Removes user specified callback for "disabled plugins collection changed" event.
	NPluginManagerRemoveDisabledPluginsChangedCallback [622]	Removes user specified callback for "disabled plugins collection changed" event.
	NPluginManagerRemoveDisabledPluginsChanging [622]	Removes user specified callback for "disabled plugins collection changing" event.
	NPluginManagerRemoveDisabledPluginsChangingCallback [622]	Removes user specified callback for "disabled plugins collection changing" event.
	NPluginManagerRemovePluginAdded [623]	Removes user specified callback for "plugin added" event.
	NPluginManagerRemovePluginAddedCallback [623]	Removes user specified callback for "plugin added" event.
	NPluginManagerRemovePluginsChanged [624]	Removes user specified callback for "plugins changed" event.
	NPluginManagerRemovePluginsChangedCallback [624]	Removes user specified callback for "plugins changed" event.
	NPluginManagerRemovePluginsChanging [624]	Removes user specified callback for "adding plugins changing" event.
	NPluginManagerRemovePluginsChangingCallback [625]	Removes user specified callback for "adding plugins changing" event.
	NPluginManagerSetPluginSearchPath [625]	Sets directory which will be automatically searched for plugins.
	NPluginManagerUnplugAll [625]	Unplugs all plugins.

Macros

Name	Description
NPluginManagerAddDisabledPlugin [626]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerAddDisabledPlugin function.
NPluginManagerAddPlugin [626]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerAddPlugin function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPluginManagerAddPluginFromFile [626]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerAddPluginFromFile function.
NPluginManagerGetDisabledPlugin [626]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerGetDisabledPlugin function.
NPluginManagerGetInterfaceType [627]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerGetInterfaceType function.
NPluginManagerGetPluginSearchPath [627]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerGetPluginSearchPath function.
NPluginManagerRemoveDisabledPlugin [627]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerRemoveDisabledPlugin function.
NPluginManagerSetPluginSearchPath [627]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerSetPluginSearchPath function.

7.1.2.41.1 Functions

7.1.2.41.1.1 NPluginManagerAddDisabledPlugin Function

Disables plugin specified by name and adds to disabled plugins collection.

C++

```
NResult N_API NPluginManagerAddDisabledPlugin(HNPluginManager hPluginManager, const NChar *
szValue, NInt * pIndex);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
const NChar * szValue	[in] Plugin name.
NInt * pIndex	[out] Disabled plugin index in disabled plugins collection.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

If plugin is not added to the framework, it will be disabled when it will be added.

See Also

NPluginManagerPlugAll [620]

NPluginManagerClearDisabledPlugin

NPluginManagerGetDisabledPluginCount [616]

NPluginManagerGetDisabledPlugin [616]

Module

NPluginManager Unit [607]

7.1.2.41.1.2 NPluginManagerAddDisabledPluginsChanged Function

Adds user specified callback for "disabled plugins collection changed" event. This callback is called when disabled plugins collection was changed.

C++

```
NResult N_API NPluginManagerAddDisabledPluginsChanged(HNPluginManager hPluginManager,
HNCallback hCallback);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
HNCallback hCallback	[in] Handle to user defined function (callback).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NPluginManagerRemoveDisabledPluginsChanged [621]

Module

NPluginManager Unit [607]

7.1.2.41.1.3 NPluginManagerAddDisabledPluginsChangedCallback Function

Adds user specified callback for "disabled plugins collection changed" event. This callback is called when disabled plugins collection was changed.

C++

```
NResult N_API NPluginManagerAddDisabledPluginsChangedCallback(HNPluginManager
hPluginManager, NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NPluginManagerRemoveDisabledPluginsChangedCallback [622]

Module

NPluginManager Unit [607]

7.1.2.41.1.4 NPluginManagerAddDisabledPluginsChanging Function

Adds user specified callback for "disabled plugins collection changing" event. This callback is called when disabled plugins collection is changing.

C++

```
NResult N_API NPluginManagerAddDisabledPluginsChanging(HNPluginManager hPluginManager,
HNCallback hCallback);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
HNCallback hCallback	[in] Handle to user defined function (callback).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

See Also

NPluginManagerRemoveDisabledPluginsChanging [622]

Module

NPluginManager Unit [607]

7.1.2.41.1.5 NPluginManagerAddDisabledPluginsChangingCallback Function

Adds user specified callback for "disabled plugins collection changing" event. This callback is called when disabled plugins collection is changing.

C++

```
NResult N_API NPluginManagerAddDisabledPluginsChangingCallback(HNPluginManager
hPluginManager, NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NPluginManagerRemoveDisabledPluginsChangingCallback [622]

Module

NPluginManager Unit [607]

7.1.2.41.1.6 NPluginManagerAddPlugin Function

Loads plugin. It's dependencies are searched relative to specified directory.

C++

```
NResult N_API NPluginManagerAddPlugin(HNPluginManager hPluginManager, HNPluginModule
hModule, const NChar * szDirectory, HNPlugin * phPlugin);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
HNPluginModule hModule	[in] Handle to NPluginModule [629].
const NChar * szDirectory	[in] Directory where plugin and its dependencies (if exist) are saved.
HNPlugin * phPlugin	[out] Handle to loaded plugin.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

The function is used when loading plugins compiled statically. The handle to the module is obtained from plugin's function having the ModuleOf name suffix. For more standard plugins (implemented in dynamic libraries) use NPluginManagerAddPluginFromFile [612] function or NPluginManagerRefresh [620] function to automatically load all plugins from a search directory (NPluginManagerGetPluginSearchPath [619]).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NPluginManager Unit [607]

7.1.2.41.1.7 NPluginManagerAddPluginAdded Function

Adds user specified callback for "plugin added" event. This callback is called when plugin was loaded into the system.

C++

```
NResult N_API NPluginManagerAddPluginAdded(HNPluginManager hPluginManager, HNCallback
hCallback);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
HNCallback hCallback	[in] Handle to user defined function (callback).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.8 NPluginManagerAddPluginAddedCallback Function

Adds user specified callback for "plugin added" event. This callback is called when plugin was loaded into the system.

C++

```
NResult N_API NPluginManagerAddPluginAddedCallback(HNPluginManager hPluginManager,
NPluginManagerPluginCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NPluginManagerPluginCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.9 NPluginManagerAddPluginFromFile Function

Loads plugin from user specified file.

C++

```
NResult N_API NPluginManagerAddPluginFromFile(HNPluginManager hPluginManager, const NChar *
szFileName, HNPlugin * phPlugin);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
const NChar * szFileName	[in] File name where plugin is saved.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNPlugin * phPlugin	[out] Handle to loaded plugin.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

The library dependencies (additional libraries required for plugin and specified during it's compilation) will be searched relatively to the directory containing the specified (*szFileName)* file.

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NPluginManager Unit [607]

7.1.2.41.1.10 NPluginManagerAddPluginsChanged Function

Adds user specified callback for "plugins changed" event.

C++

```
NResult N_API NPluginManagerAddPluginsChanged(HNPluginManager hPluginManager, HNCallback
hCallback);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
HNCallback hCallback	[in] Handle to user defined function (callback).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.11 NPluginManagerAddPluginsChangedCallback Function

Adds user specified callback for "plugins changed" event.

C++

```
NResult N_API NPluginManagerAddPluginsChangedCallback(HNPluginManager hPluginManager,
NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.12 NPluginManagerAddPluginsChanging Function

Adds user specified callback for "adding plugins changing" event. This callback is called when user started adding new

Copyright © 2009-2014 Neurotechnology. All rights reserved.

plugins

C++

```
NResult N_API NPluginManagerAddPluginsChanging(HNPluginManager hPluginManager, HNCallback hCallback);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
HNCallback hCallback	[in] Handle to user defined function (callback).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NPluginManagerRemovePluginsChanging [624]

Module

NPluginManager Unit [607]

7.1.2.41.1.13 NPluginManagerAddPluginsChangingCallback Function

Adds user specified callback for "adding plugins changing" event. This callback is called when user started adding new plugins

C++

```
NResult N_API NPluginManagerAddPluginsChangingCallback(HNPluginManager hPluginManager, NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NPluginManagerRemovePluginsChangingCallback [625]

Module

NPluginManager Unit [607]

7.1.2.41.1.14 NPluginManagerAllowsUnplug Function

Checks whether NPluginPlug [600] and NPluginUnplug [602] functions work

C++

```
NResult N_API NPluginManagerAllowsUnplug(HNPluginManager hPluginManager, NBool * pValue);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBool * pValue	[out] NBool [1004] value which indicates whether NPluginPlug [600] and NPluginUnplug [602] functions work. When NTrue [1064] is returned these functions works. Otherwise, an error is returned.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

NPluginManager Unit [607]

7.1.2.41.1.15 NPluginManagerClearDisabledPlugins Function

Clears the collection of disabled plugins.

C++

```
NResult N_API NPluginManagerClearDisabledPlugins(HNPluginManager hPluginManager);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NPluginManagerPlugAll [620]

NPluginManagerAddDisabledPlugin [609]

NPluginManagerGetDisabledPluginCount [616]

NPluginManagerGetDisabledPlugin [616]

Module

NPluginManager Unit [607]

7.1.2.41.1.16 NPluginManagerEnsurePluginSearchPerformed Function

Trigger plugin search in plugin directory [619] if search hasn't been performed yet.

C++

```
NResult N_API NPluginManagerEnsurePluginSearchPerformed(HNPluginManager hPluginManager);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle of the effected plugin manager.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If plugins has not been loaded before, all plugin libraries (file names matching certain pattern) present in plugin directory [619] will be loaded. To perform plugin search even if this has already been performed after application start use NPluginMangerRefresh.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NPluginManager Unit [607]

7.1.2.41.1.17 NPluginManagerGetDisabledPlugin Function

Gets disabled plugin specified by index.

C++

```
NResult N_API NPluginManagerGetDisabledPlugin(HNPluginManager hPluginManager, NInt index,
NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NInt index	[in] Disabled plugin index.
NChar * szValue	[out] Zero terminated string containing plugin name.
NInt valueSize	[in] Size of *szValue* memory buffer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If *szValue* is NULL [1075] function returns length of string to be written (not including the string terminator). Hence memory block pointed by *szValue* needs to be +1 byte bigger than the length returned.

See Also

NPluginManagerPlugAll [620]

NPluginManagerClearDisabledPlugin

NPluginManagerGetDisabledPluginCount [616]

NPluginManagerAddDisabledPlugin [609]

Module

NPluginManager Unit [607]

7.1.2.41.1.18 NPluginManagerGetDisabledPluginCount Function

Gets disabled plugins count.

C++

```
NResult N_API NPluginManagerGetDisabledPluginCount(HNPluginManager hPluginManager, NInt *
pValue);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NInt * pValue	[out] Disabled plugins count.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NPluginManagerGetDisabledPlugin [616]

NPluginManagerPlugAll [620]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NPluginManager Unit [607]

7.1.2.41.1.19 NPluginManagerGetInstances Function

Retrieves the instances of the plugin manager.

C++

```
NResult N_API NPluginManagerGetInstances(HNPluginManager * * parhValues, NInt *
pValueCount);
```

Parameters

Parameters	Description
HNPluginManager * * parhValues	[in] A pointer to an array of instances.
NInt * pValueCount	[in] A pointer to a number of the instances in an array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NPluginManager Unit [607]

7.1.2.41.1.20 NPluginManagerGetInterfaceType Function

Returns NPluginManager [607] interface type name.

C++

```
NResult N_API NPluginManagerGetInterfaceType(HNPluginManager hPluginManager, NChar *
szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NChar * szValue	[out] Pointer to memory block which contains interface type name.
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If *szValue* is NULL [1075] function returns size of item contained in field.

szValue memory block needs to be +1 byte bigger than value returned as NULL [1075] parameter has been passed. Additional space is used for zero termination of returned value.

Notes

NPluginManager [607] can plug in only these plugins which have the same interface type as NPluginManager [607]. When interface types mismatch these plugins will be listed with status "interface type mismatch".

Module

NPluginManager Unit [607]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.41.1.21 NPluginManagerGetInterfaceVersions Function

Returns the range of supported interface versions by specified NPluginManager [607].

C++

```
NResult N_API NPluginManagerGetInterfaceVersions(HNPluginManager hPluginManager,
NVersionRange_ * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607] to check which interface versions it supports.
NVersionRange_ * arValues	[out] Pointer to memory buffer that receives supported interface versions. Can be NULL [1075]. When NULL [1075] is passed as parameter, the length of arValues array is returned by NResult [1007]. Every NUInt [1009] in the array is a pair of NUShort [1010] which specifies version range (from...to).
NInt valuesLength	[in] Size of *arValues* memory buffer.

Returns

The length of `arValues` array.

Notes

NVersionRange [1011]* and NVersion [1010]* macros can be used to parse version range.

Module

NPluginManager Unit [607]

7.1.2.41.1.22 NPluginManagerGetPlugin Function

Returns NPlugin [594] specified by index.

C++

```
NResult N_API NPluginManagerGetPlugin(HNPluginManager hPluginManager, NInt index, HNPlugin
* phValue);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607] to retrieve one of its plugins.
NInt index	[in] Plugin index.
HNPlugin * phValue	[out] Pointer to plugin specified by index.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NPluginManager Unit [607]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.41.1.23 NPluginManagerGetPluginByName Function

Gets plugin by specified name.

C++

```
NResult N_API NPluginManagerGetPluginByName(HNPluginManager hPluginManager, const NChar *
szId, HNPlugin * phValue);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
const NChar * szId	[in] Zero-terminated string containing plugin name (Id).
HNPlugin * phValue	[out] Pointer to NPlugin [594].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.24 NPluginManagerGetPluginCount Function

Returns plugins count in specified NPluginManager [607].

C++

```
NResult N_API NPluginManagerGetPluginCount(HNPluginManager hPluginManager, NInt * pValue);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NInt * pValue	[out] Plugins count.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

NPluginManager Unit [607]

7.1.2.41.1.25 NPluginManagerGetPluginSearchPath Function

Gets plugin dependencies search path.

C++

```
NResult N_API NPluginManagerGetPluginSearchPath(HNPluginManager hPluginManager, NChar *
szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NChar * szValue	[out] Address of a buffer to store plugin dependencies directory.
NInt valueSize	[in] Size of *szValue* memory buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

All the plugins present in directory will be loaded in NPluginManagerRefresh [620] (or during first call to NPluginManagerPlugAll [620] ir refresh function is not used). The dependencies of these plugins will be searched relatively to this directory.

Module

NPluginManager Unit [607]

7.1.2.41.1.26 NPluginManagerPlugAll Function

Plugs all enabled plugins into the system.

C++

```
NResult N_API NPluginManagerPlugAll(HNPluginManager hPluginManager);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607] to plug all of its plugins.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Notes

- Only enabled plugins will be plugged, i.e. those with names not present in disabled plugins collection [616].
- If the function is called for the first time and NPluginManagerEnsurePluginSearchPerformed [615]/NPluginManagerRefresh [620] haven't been called yet the plugin search will be performed before pluggin.

Module

NPluginManager Unit [607]

7.1.2.41.1.27 NPluginManagerRefresh Function

Refreshes plugins list (search for added/removed plugin files).

C++

```
NResult N_API NPluginManagerRefresh(HNPluginManager hPluginManager);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManger to refresh its plugins list.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Notes

- The function can be used when new plugins need to be searched after their library files are added to plugin directory [619]. Alternatively the NPluginManagerAddPluginFromFile [612] function can be used to activate plugins one by one.
- On refresh enabled plugins will be unplugged, the plugin libraries located in plugin directory [619] will be reloaded and again plugged into the system. This means that if function NPluginManagerAllowsUnplug [614] returns NFalse [1041],

Copyright © 2009-2014 Neurotechnology. All rights reserved.

plugin manager will not be able to unplug and hence refresh plugins list.

- New plugins discovered by this operation will not be automatically plugged, call to NPLuginMangerPlugAll might be needed in such case.
- Different file name patterns might be employed by different users of NPluginManger, so that not all DLL present in the plugin directory are loaded. For example NPluginManger used by NDeviceManager automatically loads only files with "Ndm*.dll" name pattern.

Module

NPluginManager Unit [607]

7.1.2.41.1.28 NPluginManagerRemoveDisabledPlugin Function

Removes plugin specified by name from disabled plugins collection.

C++

```
NResult N_API NPluginManagerRemoveDisabledPlugin(HNPluginManager hPluginManager, const
NChar * szValue, NInt * pIndex);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
const NChar * szValue	[in] Plugin name.
NInt * pIndex	[out] Index of removed plugin.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.29 NPluginManagerRemoveDisabledPluginAt Function

Removes plugin specified by index from disabled plugins collection.

C++

```
NResult N_API NPluginManagerRemoveDisabledPluginAt(HNPluginManager hPluginManager, NInt
index);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NInt index	[in] Index of plugin which should be removed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.30 NPluginManagerRemoveDisabledPluginsChanged Function

Removes user specified callback for "disabled plugins collection changed" event.

C++

```
NResult N_API NPluginManagerRemoveDisabledPluginsChanged(HNPluginManager hPluginManager,
HNCallback hCallback);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
HNCallback hCallback	[in] Handle to user defined function (callback).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.31 NPluginManagerRemoveDisabledPluginsChangedCallback Function

Removes user specified callback for "disabled plugins collection changed" event.

C++

```
NResult N_API NPluginManagerRemoveDisabledPluginsChangedCallback(HNPluginManager
hPluginManager, NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.32 NPluginManagerRemoveDisabledPluginsChanging Function

Removes user specified callback for "disabled plugins collection changing" event.

C++

```
NResult N_API NPluginManagerRemoveDisabledPluginsChanging(HNPluginManager hPluginManager,
HNCallback hCallback);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
HNCallback hCallback	[in] Handle to user defined function (callback).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.33 NPluginManagerRemoveDisabledPluginsChangingCallback Function

Removes user specified callback for "disabled plugins collection changing" event.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NPluginManagerRemoveDisabledPluginsChangingCallback(HNPluginManager
hPluginManager, NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.34 NPluginManagerRemovePluginAdded Function

Removes user specified callback for "plugin added" event.

C++

```
NResult N_API NPluginManagerRemovePluginAdded(HNPluginManager hPluginManager, HNCallback
hCallback);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
HNCallback hCallback	[in] Handle to user defined function (callback).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.35 NPluginManagerRemovePluginAddedCallback Function

Removes user specified callback for "plugin added" event.

C++

```
NResult N_API NPluginManagerRemovePluginAddedCallback(HNPluginManager hPluginManager,
NPluginManagerPluginCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NPluginManagerPluginCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.41.1.36 NPluginManagerRemovePluginsChanged Function

Removes user specified callback for "plugins changed" event.

C++

```
NResult N_API NPluginManagerRemovePluginsChanged(HNPluginManager hPluginManager, HNCallback hCallback);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
HNCallback hCallback	[in] Handle to user defined function (callback) to remove.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.37 NPluginManagerRemovePluginsChangedCallback Function

Removes user specified callback for "plugins changed" event.

C++

```
NResult N_API NPluginManagerRemovePluginsChangedCallback(HNPluginManager hPluginManager, NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NObjectCallback pCallback	[in] Pointer to user defined function (callback) to remove.
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.38 NPluginManagerRemovePluginsChanging Function

Removes user specified callback for "adding plugins changing" event.

C++

```
NResult N_API NPluginManagerRemovePluginsChanging(HNPluginManager hPluginManager, HNCallback hCallback);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
HNCallback hCallback	[in] Handle to user defined function (callback).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NPluginManager Unit [607]

7.1.2.41.1.39 NPluginManagerRemovePluginsChangingCallback Function

Removes user specified callback for "adding plugins changing" event.

C++

```
NResult N_API NPluginManagerRemovePluginsChangingCallback(HNPluginManager hPluginManager,
NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginManager Unit [607]

7.1.2.41.1.40 NPluginManagerSetPluginSearchPath Function

Sets directory which will be automatically searched for plugins.

C++

```
NResult N_API NPluginManagerSetPluginSearchPath(HNPluginManager hPluginManager, const NChar
* szValue);
```

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManager [607].
const NChar * szValue	[in] Path to directory.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

All the plugins present in specified directory will be loaded in NPluginManagerRefresh [620] (or during first call to NPluginManagerPlugAll [620] ir refresh function is not used). The dependencies of these plugins will be searched relatively to this directory.

Module

NPluginManager Unit [607]

7.1.2.41.1.41 NPluginManagerUnplugAll Function

Unplugs all plugins.

C++

```
NResult N_API NPluginManagerUnplugAll(HNPluginManager hPluginManager);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNPluginManager hPluginManager	[in] Handle to NPluginManger to unplug its plugins.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

NPluginManager Unit [607]

7.1.2.41.2 Macros

7.1.2.41.2.1 NPluginManagerAddDisabledPlugin Macro

Picks a Unicode (W) or ASCII (A) version of the NPluginManagerAddDisabledPlugin function.

C++

```
#define NPluginManagerAddDisabledPlugin N_FUNC_AW(NPluginManagerAddDisabledPlugin)
```

Module

NPluginManager Unit [607]

7.1.2.41.2.2 NPluginManagerAddPlugin Macro

Picks a Unicode (W) or ASCII (A) version of the NPluginManagerAddPlugin function.

C++

```
#define NPluginManagerAddPlugin N_FUNC_AW(NPluginManagerAddPlugin)
```

Module

NPluginManager Unit [607]

7.1.2.41.2.3 NPluginManagerAddPluginFromFile Macro

Picks a Unicode (W) or ASCII (A) version of the NPluginManagerAddPluginFromFile function.

C++

```
#define NPluginManagerAddPluginFromFile N_FUNC_AW(NPluginManagerAddPluginFromFile)
```

Module

NPluginManager Unit [607]

7.1.2.41.2.4 NPluginManagerGetDisabledPlugin Macro

Picks a Unicode (W) or ASCII (A) version of the NPluginManagerGetDisabledPlugin function.

C++

```
#define NPluginManagerGetDisabledPlugin N_FUNC_AW(NPluginManagerGetDisabledPlugin)
```

Module

NPluginManager Unit [607]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.41.2.5 NPluginManagerGetInterfaceType Macro

Picks a Unicode (W) or ASCII (A) version of the NPluginManagerGetInterfaceType function.

C++

```
#define NPluginManagerGetInterfaceType N_FUNC_AW(NPluginManagerGetInterfaceType)
```

Module

NPluginManager Unit [607]

7.1.2.41.2.6 NPluginManagerGetPluginSearchPath Macro

Picks a Unicode (W) or ASCII (A) version of the NPluginManagerGetPluginSearchPath function.

C++

```
#define NPluginManagerGetPluginSearchPath N_FUNC_AW(NPluginManagerGetPluginSearchPath)
```

Module

NPluginManager Unit [607]

7.1.2.41.2.7 NPluginManagerRemoveDisabledPlugin Macro

Picks a Unicode (W) or ASCII (A) version of the NPluginManagerRemoveDisabledPlugin function.

C++

```
#define NPluginManagerRemoveDisabledPlugin N_FUNC_AW(NPluginManagerRemoveDisabledPlugin)
```

Module

NPluginManager Unit [607]

7.1.2.41.2.8 NPluginManagerSetPluginSearchPath Macro

Picks a Unicode (W) or ASCII (A) version of the NPluginManagerSetPluginSearchPath function.

C++

```
#define NPluginManagerSetPluginSearchPath N_FUNC_AW(NPluginManagerSetPluginSearchPath)
```

Module

NPluginManager Unit [607]

7.1.2.41.3 Files

7.1.2.41.3.1 NPluginManager.h

Header file for NPluginManager [607]. Provides functions for working with NPlugin [594].

Functions

	Name	Description
	NPluginManagerAddDisabledPlugin [609]	Disables plugin specified by name and adds to disabled plugins collection.
	NPluginManagerAddDisabledPluginsChanged [609]	Adds user specified callback for "disabled plugins collection changed" event. This callback is called when disabled plugins collection was changed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NPluginManagerAddDisabledPluginsChangedCallback [610]	Adds user specified callback for "disabled plugins collection changed" event. This callback is called when disabled plugins collection was changed.
	NPluginManagerAddDisabledPluginsChanging [610]	Adds user specified callback for "disabled plugins collection changing" event. This callback is called when disabled plugins collection is changing.
	NPluginManagerAddDisabledPluginsChangingCallback [611]	Adds user specified callback for "disabled plugins collection changing" event. This callback is called when disabled plugins collection is changing.
	NPluginManagerAddPlugin [611]	Loads plugin. It's dependencies are searched relative to specified directory.
	NPluginManagerAddPluginAdded [612]	Adds user specified callback for "plugin added" event. This callback is called when plugin was loaded into the system.
	NPluginManagerAddPluginAddedCallback [612]	Adds user specified callback for "plugin added" event. This callback is called when plugin was loaded into the system.
	NPluginManagerAddPluginFromFile [612]	Loads plugin from user specified file.
	NPluginManagerAddPluginsChanged [613]	Adds user specified callback for "plugins changed" event.
	NPluginManagerAddPluginsChangedCallback [613]	Adds user specified callback for "plugins changed" event.
	NPluginManagerAddPluginsChanging [613]	Adds user specified callback for "adding plugins changing" event. This callback is called when user started adding new plugins
	NPluginManagerAddPluginsChangingCallback [614]	Adds user specified callback for "adding plugins changing" event. This callback is called when user started adding new plugins
	NPluginManagerAllowsUnplug [614]	Checks whether NPluginPlug [600] and NPluginUnplug [602] functions work
	NPluginManagerClearDisabledPlugins [615]	Clears the collection of disabled plugins.
	NPluginManagerEnsurePluginSearchPerformed [615]	Trigger plugin search in plugin directory [619] if search hasn't been performed yet.
	NPluginManagerGetDisabledPlugin [616]	Gets disabled plugin specified by index.
	NPluginManagerGetDisabledPluginCount [616]	Gets disabled plugins count.
	NPluginManagerGetInstances [617]	Retrieves the instances of the plugin manager.
	NPluginManagerGetInterfaceType [617]	Returns NPluginManager [607] interface type name.
	NPluginManagerGetInterfaceVersions [618]	Returns the range of supported interface versions by specified NPluginManager [607].
	NPluginManagerGetPlugin [618]	Returns NPlugin [594] specified by index.
	NPluginManagerGetPluginByName [619]	Gets plugin by specified name.
	NPluginManagerGetPluginCount [619]	Returns plugins count in specified NPluginManager [607].
	NPluginManagerGetPluginSearchPath [619]	Gets plugin dependencies search path.
	NPluginManagerPlugAll [620]	Plugs all enabled plugins into the system.
	NPluginManagerRefresh [620]	Refreshes plugins list (search for added/removed plugin files).
	NPluginManagerRemoveDisabledPlugin [621]	Removes plugin specified by name from disabled plugins collection.
	NPluginManagerRemoveDisabledPluginAt [621]	Removes plugin specified by index from disabled plugins collection.
	NPluginManagerRemoveDisabledPluginsChanged [621]	Removes user specified callback for "disabled plugins collection changed" event.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NPluginManagerRemoveDisabledPluginsChangedCallback [622]	Removes user specified callback for "disabled plugins collection changed" event.
	NPluginManagerRemoveDisabledPluginsChanging [622]	Removes user specified callback for "disabled plugins collection changing" event.
	NPluginManagerRemoveDisabledPluginsChangingCallback [622]	Removes user specified callback for "disabled plugins collection changing" event.
	NPluginManagerRemovePluginAdded [623]	Removes user specified callback for "plugin added" event.
	NPluginManagerRemovePluginAddedCallback [623]	Removes user specified callback for "plugin added" event.
	NPluginManagerRemovePluginsChanged [624]	Removes user specified callback for "plugins changed" event.
	NPluginManagerRemovePluginsChangedCallback [624]	Removes user specified callback for "plugins changed" event.
	NPluginManagerRemovePluginsChanging [624]	Removes user specified callback for "adding plugins changing" event.
	NPluginManagerRemovePluginsChangingCallback [625]	Removes user specified callback for "adding plugins changing" event.
	NPluginManagerSetPluginSearchPath [625]	Sets directory which will be automatically searched for plugins.
	NPluginManagerUnplugAll [625]	Unplugs all plugins.

Macros

Name	Description
NPluginManagerAddDisabledPlugin [626]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerAddDisabledPlugin function.
NPluginManagerAddPlugin [626]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerAddPlugin function.
NPluginManagerAddPluginFromFile [626]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerAddPluginFromFile function.
NPluginManagerGetDisabledPlugin [626]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerGetDisabledPlugin function.
NPluginManagerGetInterfaceType [627]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerGetInterfaceType function.
NPluginManagerGetPluginSearchPath [627]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerGetPluginSearchPath function.
NPluginManagerRemoveDisabledPlugin [627]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerRemoveDisabledPlugin function.
NPluginManagerSetPluginSearchPath [627]	Picks a Unicode (W) or ASCII (A) version of the NPluginManagerSetPluginSearchPath function.

Module

NPluginManager Unit [607]

7.1.2.42 NPluginModule Unit

Provides functions to retrieve information about plugins.

Functions

	Name	Description
	NPluginModuleGetIncompatiblePlugins [630]	Gets incompatible plugins list.
	NPluginModuleCreate [631]	Creates a new NPluginModule.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NPluginModuleGetInterfaceType [631]	Gets plugin interface type name.
	NPluginModuleGetInterfaceVersions [631]	Returns the range of supported interface versions by specified NPluginModule.
	NPluginModuleSetIncompatiblePluginsN [632]	Sets the list of incompatible plugins for the specified plugin module.
	NPluginModuleGetPluginName [632]	Gets plugin name.
	NPluginModuleSetInterfaceTypeN [633]	Sets a new interface type for the specified plugin module.
	NPluginModuleSetPluginNameN [633]	Sets the name of the plugin for the specified plugin module.
	NPluginModuleSetPriority [633]	Sets the priority of the specified plugin module.
	NPluginModuleGetPriority [634]	Gets plugin priority over other plugins.
	HNPluginModule [634]	Handle to NPluginModule.

Macros

Name	Description
NPluginModuleGetIncompatiblePlugins [634]	Picks a Unicode (W) or ANSCII (A) version of the NPluginModuleGetIncompatiblePlugins function.
NPluginModuleGetInterfaceType [634]	Picks a Unicode (W) or ANSCII (A) version of the NPluginModuleGetInterfaceType function.
NPluginModuleGetPluginName [635]	Picks a Unicode (W) or ANSCII (A) version of the NPluginModuleGetPluginName function.
NPluginModuleSetIncompatiblePlugins [635]	Picks a Unicode (W) or ANSCII (A) version of the NPluginModuleSetIncompatiblePlugins function.
NPluginModuleSetInterfaceType [635]	Picks a Unicode (W) or ANSCII (A) version of the NPluginModuleSetInterfaceTypes function.
NPluginModuleSetPluginName [635]	Picks a Unicode (W) or ANSCII (A) version of the NPluginModuleSetPluginType function.

7.1.2.42.1 Functions

7.1.2.42.1.1 NPluginModuleGetIncompatiblePlugins Function

Gets incompatible plugins list.

C++

```
NResult N_API NPluginModuleGetIncompatiblePlugins(HNPluginModule hModule, NChar * szValue,
NInt valueSize);
```

Parameters

Parameters	Description
HNPluginModule hModule	[in] Handle to NPluginModule [629].
NChar * szValue	[out] Pointer to incompatible plugins names separated by semicolon.
NInt valueSize	[in] Size of memory buffer (*szValue*) containing string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginModule Unit [629]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.42.1.2 NPluginModuleCreate Function

Creates a new NPluginModule [629].

C++

```
NResult N_API NPluginModuleCreate(HNPluginModule * phModule);
```

Parameters

Parameters	Description
HNPluginModule * phModule	[out] Pointer to a handle to the new NPluginModule [629].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginModule Unit [629]

7.1.2.42.1.3 NPluginModuleGetInterfaceType Function

Gets plugin interface type name.

C++

```
NResult N_API NPluginModuleGetInterfaceType(HNPluginModule hModule, NChar * szValue, NInt
valueSize);
```

Parameters

Parameters	Description
HNPluginModule hModule	[in] Handle to NPluginModule [629].
NChar * szValue	[out] Pointer to memory block which contains interface type name.
NInt valueSize	[in] Size of memory buffer containing name (*szValue*).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If *szValue* is NULL [1075] function returns size of item contained in field.

szValue memory block needs to be +1 byte bigger than value returned as NULL [1075] parameter has been passed. Additional space is used for zero termination of returned value.

Module

NPluginModule Unit [629]

7.1.2.42.1.4 NPluginModuleGetInterfaceVersions Function

Returns the range of supported interface versions by specified NPluginModule [629].

C++

```
NResult N_API NPluginModuleGetInterfaceVersions(HNPluginModule hModule, NVersionRange_ *
arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNPluginModule hModule	[in] Handle to NPluginModule [629] to check which interface versions it supports.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NVersionRange_ * arValues	[out] Pointer to memory buffer that receives supported interface versions. Can be NULL [1075]. When NULL [1075] is passed as parameter, the length of *arValues* array is returned by NResult [1007]. Every NUInt [1009] in the array is a pair of NUShort [1010] which specifies version range (from...to).
NInt valuesLength	[in] Size of *arValues* memory buffer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

NVersionRange [1011]* and NVersion [1010]* macros can be used to parse version range.

Module

NPluginModule Unit [629]

7.1.2.42.1.5 NPluginModuleSetIncompatiblePluginsN Function

Sets the list of incompatible plugins for the specified plugin module.

C++

```
NResult N_API NPluginModuleSetIncompatiblePluginsN(HNPluginModule hModule, HNString hValue);
```

Parameters

Parameters	Description
HNPluginModule hModule	[in] Handle to the NPluginModule [629].
HNString hValue	[in] Handle to the list of incompatible plugins.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginModule Unit [629]

7.1.2.42.1.6 NPluginModuleGetPluginName Function

Gets plugin name.

C++

```
NResult N_API NPluginModuleGetPluginName(HNPluginModule hModule, NChar * szValue, NInt
valueSize);
```

Parameters

Parameters	Description
HNPluginModule hModule	[in] Handle to NPluginModule [629].
NChar * szValue	[out] Pointer to memory buffer where plugin name is saved.
NInt valueSize	[in] Size of *szValue* memory buffer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginModule Unit [629]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.42.1.7 NPluginModuleSetInterfaceTypeN Function

Sets a new interface type for the specified plugin module.

C++

```
NResult N_API NPluginModuleSetInterfaceTypeN(HNPluginModule hModule, HNString hValue);
```

Parameters

Parameters	Description
HNPluginModule hModule	[in] Handle to the NPluginModule [629].
HNString hValue	[in] Handle to NString [678] containing new interface type value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginModule Unit [629]

7.1.2.42.1.8 NPluginModuleSetPluginNameN Function

Sets the name of the plugin for the specified plugin module.

C++

```
NResult N_API NPluginModuleSetPluginNameN(HNPluginModule hModule, HNString hValue);
```

Parameters

Parameters	Description
HNPluginModule hModule	[in] Handle to the NPluginModule [629].
HNString hValue	[in] Handle to the string containing the name of the plugin.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginModule Unit [629]

7.1.2.42.1.9 NPluginModuleSetPriority Function

Sets the priority of the specified plugin module.

C++

```
NResult N_API NPluginModuleSetPriority(HNPluginModule hModule, NInt value);
```

Parameters

Parameters	Description
HNPluginModule hModule	[in] Handle to the NPluginModule [629].
NInt value	[in] New priority value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginModule Unit [629]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.42.1.10 NPluginModuleGetPriority Function

Gets plugin priority over other plugins.

C++

```
NResult N_API NPluginModuleGetPriority(HNPluginModule hModule, NInt * pValue);
```

Parameters

Parameters	Description
HNPluginModule hModule	[in] Handle to NPluginModule [629].
NInt * pValue	[out] Plugin priority.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

Priority influences plugins adding to the system. If two or more plugins are incompatible or duplicate plugins exist plugin with higher priority will be added to the system. Priority is set during plugin compilation.

Default plugin priority is 0.

Module

NPluginModule Unit [629]

7.1.2.42.1.11 HNPluginModule

Handle to NPluginModule [629].

Module

NPluginModule Unit [629]

7.1.2.42.2 Macros

7.1.2.42.2.1 NPluginModuleGetIncompatiblePlugins Macro

Picks a Unicode (W) or ANSCII (A) version of the NPluginModuleGetIncompatiblePlugins function.

C++

```
#define NPluginModuleGetIncompatiblePlugins N_FUNC_AW(NPluginModuleGetIncompatiblePlugins)
```

Module

NPluginModule Unit [629]

7.1.2.42.2.2 NPluginModuleGetInterfaceType Macro

Picks a Unicode (W) or ANSCII (A) version of the NPluginModuleGetInterfaceType function.

C++

```
#define NPluginModuleGetInterfaceType N_FUNC_AW(NPluginModuleGetInterfaceType)
```

Module

NPluginModule Unit [629]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.42.2.3 NPluginModuleGetPluginName Macro

Picks a Unicode (W) or ANSCII (A) version of the NPluginModuleGetPluginName function.

C++

```
#define NPluginModuleGetPluginName N_FUNC_AW(NPluginModuleGetPluginName)
```

Module

NPluginModule Unit [629]

7.1.2.42.2.4 NPluginModuleSetIncompatiblePlugins Macro

Picks a Unicode (W) or ANSCII (A) version of the NPluginModuleSetIncompatiblePlugins function.

C++

```
#define NPluginModuleSetIncompatiblePlugins N_FUNC_AW(NPluginModuleSetIncompatiblePlugins)
```

Module

NPluginModule Unit [629]

7.1.2.42.2.5 NPluginModuleSetInterfaceType Macro

Picks a Unicode (W) or ANSCII (A) version of the NPluginModuleSetInterfaceTypes function.

C++

```
#define NPluginModuleSetInterfaceType N_FUNC_AW(NPluginModuleSetInterfaceType)
```

Module

NPluginModule Unit [629]

7.1.2.42.2.6 NPluginModuleSetPluginName Macro

Picks a Unicode (W) or ANSCII (A) version of the NPluginModuleSetPluginType function.

C++

```
#define NPluginModuleSetPluginName N_FUNC_AW(NPluginModuleSetPluginName)
```

Module

NPluginModule Unit [629]

7.1.2.43 NPluginModuleInternal Unit

Provides functions to modify information about plugins.

Functions

	Name	Description
	NPluginModuleSetInterfaceVersions [636]	Sets plugin interface versions for the specified plugin manager.
	NPluginModuleSetPlug [636]	Sets the plug process for the specified plugin module.
	NPluginModuleSetUnplug [636]	Sets the unplug process for the specified plugin module.

7.1.2.43.1 Functions

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.43.1.1 NPluginModuleSetInterfaceVersions Function

Sets plugin interface versions for the specified plugin manager.

C++

```
NResult N_API NPluginModuleSetInterfaceVersions(HNPluginModule hModule, const
NPluginInterfaceVersionInfo * arValues, NInt count);
```

Parameters

Parameters	Description
HNPluginModule hModule	[in] Handle to the NPluginModule [629].
const NPluginInterfaceVersionInfo * arValues	[in] Pointer to an array of plugin interface versions.
NInt count	[in] The number of interface version.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginModuleInternal Unit [635]

7.1.2.43.1.2 NPluginModuleSetPlug Function

Sets the plug process for the specified plugin module.

C++

```
NResult N_API NPluginModuleSetPlug(HNPluginModule hModule, NPluginModulePlugProc pValue);
```

Parameters

Parameters	Description
HNPluginModule hModule	[in] Handle to the NPluginModule [629].
NPluginModulePlugProc pValue	[in] The plug process.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginModuleInternal Unit [635]

7.1.2.43.1.3 NPluginModuleSetUnplug Function

Sets the unplug process for the specified plugin module.

C++

```
NResult N_API NPluginModuleSetUnplug(HNPluginModule hModule, NPluginModuleUnplugProc
pValue);
```

Parameters

Parameters	Description
HNPluginModule hModule	[in] Handle to the NPluginModule [629].
NPluginModuleUnplugProc pValue	[in] The unplug process.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPluginModuleInternal Unit [635]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.44 NProcessorInfo Unit

Provides functionality for getting processor information.

Files

Name	Description
NProcessorInfo.h [649]	Header file for the NProcessorInfo. Provides functionality for getting processor information.

Functions

	Name	Description
	NProcessorInfoGetCount [638]	Retrieves the number of logical processors in the current group.
	NProcessorInfoGetModelInfo [639]	Retrieves processor model information.
	NProcessorInfoGetModelNameEx [639]	Retrieves processor model name.
	NProcessorInfoGetModelNameN [639]	Retrieves processor model name.
	NProcessorInfoGetVendor [640]	Retrieves a pointer to enumeration that contains information about processor's vendor.
	NProcessorInfoGetVendorNameEx [640]	Retrieves processor's vendor name.
	NProcessorInfoGetVendorNameN [640]	Retrieves processor's vendor name.
	NProcessorInfoIs3DNowSupported [641]	
	NProcessorInfoIs3DNowSupportedEx [641]	Returns a Boolean value which indicates 3D support for the particular processor model.
	NProcessorInfoIsLZCntSupported [641]	
	NProcessorInfoIsLZCntSupportedEx [642]	Returns a Boolean value which indicates LZCnt support for the particular processor model.
	NProcessorInfoIsMmxSupported [642]	
	NProcessorInfoIsMmxSupportedEx [642]	Returns a Boolean value which indicates if a single instruction set (Mmx) is supported for the particular processor model.
	NProcessorInfoIsNeonSupported [642]	Returns a Boolean value which indicates if Neon is supported for the particular processor model.
	NProcessorInfoIsPopCntSupported [643]	
	NProcessorInfoIsPopCntSupportedEx [643]	Returns a Boolean value which indicates if PopCnt is supported for the particular processor model.
	NProcessorInfoIsSse2Supported [643]	
	NProcessorInfoIsSse2SupportedEx [644]	Returns a Boolean value which indicates if a Streaming SIMD Extension 2 (SSE2) instruction set is supported for a particular processor model.
	NProcessorInfoIsSse3Supported [644]	
	NProcessorInfoIsSse3SupportedEx [644]	Returns a Boolean value which indicates if a Streaming SIMD Extension 3 (SSE3) instruction set is supported for a particular processor model.
	NProcessorInfoIsSse41Supported [645]	

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NProcessorInfoIsSse41SupportedEx [645]	Returns a Boolean value which indicates if a Streaming SIMD Extension 4.1 (SSE41) instruction set is supported for a particular processor model.
	NProcessorInfoIsSse42Supported [645]	
	NProcessorInfoIsSse42SupportedEx [645]	Returns a Boolean value which indicates if a Streaming SIMD Extension 4.2 (SSE42) instruction set is supported for a particular processor model.
	NProcessorInfoIsSse4aSupported [646]	
	NProcessorInfoIsSse4aSupportedEx [646]	Returns a Boolean value which indicates if a Streaming SIMD Extension 4a (SSE4a) instruction set is supported for a particular processor model.
	NProcessorInfoIsSse5Supported [646]	
	NProcessorInfoIsSse5SupportedEx [647]	Returns a Boolean value which indicates if a Streaming SIMD Extension 5 (SSE5) instruction set is supported for a particular processor model.
	NProcessorInfoIsSseSupported [647]	
	NProcessorInfoIsSseSupportedEx [647]	Returns a Boolean value which indicates if a Streaming SIMD (SSE) instruction set is supported for a particular processor model.
	NProcessorInfoIsSsse3Supported [648]	
	NProcessorInfoIsSsse3SupportedEx [648]	Returns a Boolean value which indicates if a Supplemental Streaming SIMD Extension 3 (SSSE3) instruction set is supported for a particular processor model.

Macros

Name	Description
NProcessorInfoGetModelNameEx [649]	Picks a Unicode (W) or ASCII (A) version of the NProcessorInfoGetModelNameEx function.
NProcessorInfoGetVendorNameEx [649]	Picks a Unicode (W) or ASCII (A) version of the NProcessorInfoGetVendorNameEx function.

Structs, Records, Enums

	Name	Description
	NProcessorVendor_ [648]	Enumerates different processor vendors.
	NProcessorVendor [648]	Enumerates different processor vendors.

7.1.2.44.1 Functions

7.1.2.44.1.1 NProcessorInfoGetCount Function

Retrieves the number of logical processors in the current group.

C++

```
NResult N_API NProcessorInfoGetCount(NInt * pValue);
```

Parameters

Parameters	Description
NInt * pValue	[out] A pointer to the number of processors.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.2 NProcessorInfoGetModelInfo Function

Retrieves processor model information.

C++

```
NResult N_API NProcessorInfoGetModelInfo(NInt * pFamily, NInt * pModel, NInt * pStepping);
```

Parameters

Parameters	Description
NInt * pFamily	[out] A pointer to a value defining processor family.
NInt * pModel	[out] A pointer to a value defining processor model.
NInt * pStepping	[out] A pointer to a value defining processor stepping.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.3 NProcessorInfoGetModelNameEx Function

Retrieves processor model name.

C++

```
NResult N_API NProcessorInfoGetModelNameEx(NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
NChar * szValue	[in] A pointer to memory block that contains processor model name.
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.4 NProcessorInfoGetModelNameN Function

Retrieves processor model name.

C++

```
NResult N_API NProcessorInfoGetModelNameN(HNString * phValue);
```

Parameters

Parameters	Description
HNString * phValue	[out] A pointer to a string containing processor model name.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.5 NProcessorInfoGetVendor Function

Retrieves a pointer to enumeration that contains information about processor's vendor.

C++

```
NResult N_API NProcessorInfoGetVendor(NProcessorVendor * pValue);
```

Parameters

Parameters	Description
NProcessorVendor * pValue	[out] A pointer to memory block that contains NProcessorVendor [648] enumeration.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	pValue [870] is NULL [1075].

Module

NProcessorInfo Unit [637]

7.1.2.44.1.6 NProcessorInfoGetVendorNameEx Function

Retrieves processor's vendor name.

C++

```
NResult N_API NProcessorInfoGetVendorNameEx(NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
NChar * szValue	[out] A pointer to memory block that contains processor's vendor name.
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.7 NProcessorInfoGetVendorNameN Function

Retrieves processor's vendor name.

C++

```
NResult N_API NProcessorInfoGetVendorNameN(HNString * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString * phValue	[out] A pointer to a string containing processor vendor name.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.8 NProcessorInfoIs3DNowSupported Function

C++

```
NBool N_API NProcessorInfoIs3DNowSupported();
```

Returns

Returns a Boolean value which indicates 3D support for the particular processor model.

Remarks

This function is deprecated, use NProcessorInfoIs3DNowSupportedEx [641] instead.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.9 NProcessorInfoIs3DNowSupportedEx Function

Returns a Boolean value which indicates 3D support for the particular processor model.

C++

```
NResult N_API NProcessorInfoIs3DNowSupportedEx(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value, indicating 3D support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.10 NProcessorInfoIsLZCntSupported Function

C++

```
NBool N_API NProcessorInfoIsLZCntSupported();
```

Returns

Returns a Boolean value which indicates LZCnt support for the particular processor model.

Remarks

This function is deprecated, use NProcessorInfoIsLZCntSupportedEx [642] instead.

Module

NProcessorInfo Unit [637]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.44.1.11 NProcessorInfoIsLZCntSupportedEx Function

Returns a Boolean value which indicates LZCnt support for the particular processor model.

C++

```
NResult N_API NProcessorInfoIsLZCntSupportedEx(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value indicating LZCnt support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.12 NProcessorInfoIsMmxSupported Function

C++

```
NBool N_API NProcessorInfoIsMmxSupported();
```

Returns

Returns a Boolean value which indicates if a single instruction set (Mmx) is supported for the particular processor model.

Remarks

This function is deprecated, use NProcessorInfoIsMmxSupportedEx [642] instead.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.13 NProcessorInfoIsMmxSupportedEx Function

Returns a Boolean value which indicates if a single instruction set (Mmx) is supported for the particular processor model.

C++

```
NResult N_API NProcessorInfoIsMmxSupportedEx(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value indicating Mmx support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.14 NProcessorInfoIsNeonSupported Function

Returns a Boolean value which indicates if Neon is supported for the particular processor model.

C++

```
NResult N_API NProcessorInfoIsNeonSupported(NBool * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value indicating Neon support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.15 NProcessorInfoIsPopCntSupported Function

C++

```
NBool N_API NProcessorInfoIsPopCntSupported();
```

Returns

Returns a Boolean value which indicates if PopCnt is supported for the particular processor model.

Remarks

This function is deprecated, use NProcessorInfoIsPopCntSupportedEx [643] instead.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.16 NProcessorInfoIsPopCntSupportedEx Function

Returns a Boolean value which indicates if PopCnt is supported for the particular processor model.

C++

```
NResult N_API NProcessorInfoIsPopCntSupportedEx(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value indicating PopCnt support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.17 NProcessorInfoIsSse2Supported Function

C++

```
NBool N_API NProcessorInfoIsSse2Supported();
```

Returns

Returns a Boolean value which indicates if a Streaming SIMD Extension 2 (SSE2) instruction set is supported for a particular processor model.

Remarks

This function is deprecated, use NProcessorInfoIsSse2SupportedEx [644] instead.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.18 NProcessorInfoIsSse2SupportedEx Function

Returns a Boolean value which indicates if a Streaming SIMD Extension 2 (SSE2) instruction set is supported for a particular processor model.

C++

```
NResult N_API NProcessorInfoIsSse2SupportedEx(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value indicating SSE2 support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.19 NProcessorInfoIsSse3Supported Function

C++

```
NBool N_API NProcessorInfoIsSse3Supported();
```

Returns

Returns a Boolean value which indicates if a Streaming SIMD Extension 3 (SSE3) instruction set is supported for a particular processor model.

Remarks

This function is deprecated, use NProcessorInfoIsSse3SupportedEx [644] instead.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.20 NProcessorInfoIsSse3SupportedEx Function

Returns a Boolean value which indicates if a Streaming SIMD Extension 3 (SSE3) instruction set is supported for a particular processor model.

C++

```
NResult N_API NProcessorInfoIsSse3SupportedEx(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value indicating SSE3 support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.44.1.21 NProcessorInfoIsSse41Supported Function

C++

```
NBool N_API NProcessorInfoIsSse41Supported();
```

Returns

Returns a Boolean value which indicates if a Streaming SIMD Extension 4.1 (SSE41) instruction set is supported for a particular processor model.

Remarks

This function is deprecated, use NProcessorInfoIsSse41SupportedEx [645] instead.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.22 NProcessorInfoIsSse41SupportedEx Function

Returns a Boolean value which indicates if a Streaming SIMD Extension 4.1 (SSE41) instruction set is supported for a particular processor model.

C++

```
NResult N_API NProcessorInfoIsSse41SupportedEx(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value indicating SSE41 support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.23 NProcessorInfoIsSse42Supported Function

C++

```
NBool N_API NProcessorInfoIsSse42Supported();
```

Returns

Returns a Boolean value which indicates if a Streaming SIMD Extension 4.2 (SSE42) instruction set is supported for a particular processor model.

Remarks

This function is deprecated, use NProcessorInfoIsSse42SupportedEx [645] instead.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.24 NProcessorInfoIsSse42SupportedEx Function

Returns a Boolean value which indicates if a Streaming SIMD Extension 4.2 (SSE42) instruction set is supported for a particular processor model.

C++

```
NResult N_API NProcessorInfoIsSse42SupportedEx(NBool * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value indicating SSE42 support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.25 NProcessorInfoIsSse4aSupported Function

C++

```
NBool N_API NProcessorInfoIsSse4aSupported();
```

Returns

Returns a Boolean value which indicates if a Streaming SIMD Extension 4a (SSE4a) instruction set is supported for a particular processor model.

Remarks

This function is deprecated, use NProcessorInfoIsSse4aSupportedEx [646] instead.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.26 NProcessorInfoIsSse4aSupportedEx Function

Returns a Boolean value which indicates if a Streaming SIMD Extension 4a (SSE4a) instruction set is supported for a particular processor model.

C++

```
NResult N_API NProcessorInfoIsSse4aSupportedEx(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value indicating SSE4a support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.27 NProcessorInfoIsSse5Supported Function

C++

```
NBool N_API NProcessorInfoIsSse5Supported();
```

Returns

Returns a Boolean value which indicates if a Streaming SIMD Extension 5 (SSE5) instruction set is supported for a particular processor model.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

This function is deprecated, use NProcessorInfoIsSse5SupportedEx [647] instead.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.28 NProcessorInfoIsSse5SupportedEx Function

Returns a Boolean value which indicates if a Streaming SIMD Extension 5 (SSE5) instruction set is supported for a particular processor model.

C++

```
NResult N_API NProcessorInfoIsSse5SupportedEx(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value indicating SSE5 support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.29 NProcessorInfoIsSseSupported Function

C++

```
NBool N_API NProcessorInfoIsSseSupported();
```

Returns

Returns a Boolean value which indicates if a Streaming SIMD (SSE) instruction set is supported for a particular processor model.

Remarks

This function is deprecated, use NProcessorInfoIsSseSupportedEx [647] instead.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.30 NProcessorInfoIsSseSupportedEx Function

Returns a Boolean value which indicates if a Streaming SIMD (SSE) instruction set is supported for a particular processor model.

C++

```
NResult N_API NProcessorInfoIsSseSupportedEx(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value indicating SSE support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.31 NProcessorInfoIsSsse3Supported Function

C++

```
NBool N_API NProcessorInfoIsSsse3Supported();
```

Returns

Returns a Boolean value which indicates if a Supplemental Streaming SIMD Extension 3 (SSSE3) instruction set is supported for a particular processor model.

Remarks

This function is deprecated, use NProcessorInfoIsSsse3SupportedEx [648] instead.

Module

NProcessorInfo Unit [637]

7.1.2.44.1.32 NProcessorInfoIsSsse3SupportedEx Function

Returns a Boolean value which indicates if a Supplemental Streaming SIMD Extension 3 (SSSE3) instruction set is supported for a particular processor model.

C++

```
NResult N_API NProcessorInfoIsSsse3SupportedEx(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] A pointer to a NBool [1004] value indicating SSSE3 support.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NProcessorInfo Unit [637]

7.1.2.44.2 Structs, Records, Enums

7.1.2.44.2.1 NProcessorVendor Enumeration

Enumerates different processor vendors.

C++

```
typedef enum NProcessorVendor_ {
  npvUnknown = 0,
  npvAmd = 1,
  npvCentaur = 2,
  npvCyrix = 3,
  npvIntel = 4,
  npvNationalSemiconductor = 5,
  npvNexGen = 6,
  npvRiseTechnology = 7,
  npvSiS = 8,
  npvTransmeta = 9,
  npvUmc = 10
} NProcessorVendor;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Members

Members	Description
npvUnknown = 0	Vendor is unknown.
npvAmd = 1	Advanced Micro Devices, Inc. (AMD)
npvCentaur = 2	Centaur Technology, Inc.
npvCyrix = 3	Cyrix, Inc.
npvIntel = 4	Intel Corporation.
npvNationalSemiconductor = 5	National Semiconductor Corporation.
npvNexGen = 6	NexGen Technologies, Inc.
npvRiseTechnology = 7	Rise Technology, Inc.
npvSiS = 8	Silicon Integrated Systems Corp.
npvTransmeta = 9	Transmeta Corp.
npvUmc = 10	United Microelectronics Corporation.

Module

NProcessorInfo Unit [637]

7.1.2.44.3 Macros

7.1.2.44.3.1 NProcessorInfoGetModelNameEx Macro

Picks a Unicode (W) or ASCII (A) version of the NProcessorInfoGetModelNameEx function.

C++

```
#define NProcessorInfoGetModelNameEx N_FUNC_AW(NProcessorInfoGetModelNameEx)
```

Module

NProcessorInfo Unit [637]

7.1.2.44.3.2 NProcessorInfoGetVendorNameEx Macro

Picks a Unicode (W) or ASCII (A) version of the NProcessorInfoGetVendorNameEx function.

C++

```
#define NProcessorInfoGetVendorNameEx N_FUNC_AW(NProcessorInfoGetVendorNameEx)
```

Module

NProcessorInfo Unit [637]

7.1.2.44.4 Files

7.1.2.44.4.1 NProcessorInfo.h

Header file for the NProcessorInfo [637]. Provides functionality for getting processor information.

Enumerations

	Name	Description
	NProcessorVendor_ [648]	Enumerates different processor vendors.
	NProcessorVendor [648]	Enumerates different processor vendors.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NProcessorInfoGetCount [638]	Retrieves the number of logical processors in the current group.
	NProcessorInfoGetModelInfo [639]	Retrieves processor model information.
	NProcessorInfoGetModelNameEx [639]	Retrieves processor model name.
	NProcessorInfoGetModelNameN [639]	Retrieves processor model name.
	NProcessorInfoGetVendor [640]	Retrieves a pointer to enumeration that contains information about processor's vendor.
	NProcessorInfoGetVendorNameEx [640]	Retrieves processor's vendor name.
	NProcessorInfoGetVendorNameN [640]	Retrieves processor's vendor name.
	NProcessorInfoIs3DNowSupported [641]	
	NProcessorInfoIs3DNowSupportedEx [641]	Returns a Boolean value which indicates 3D support for the particular processor model.
	NProcessorInfoIsLZCntSupported [641]	
	NProcessorInfoIsLZCntSupportedEx [642]	Returns a Boolean value which indicates LZCnt support for the particular processor model.
	NProcessorInfoIsMmxSupported [642]	
	NProcessorInfoIsMmxSupportedEx [642]	Returns a Boolean value which indicates if a single instruction set (Mmx) is supported for the particular processor model.
	NProcessorInfoIsNeonSupported [642]	Returns a Boolean value which indicates if Neon is supported for the particular processor model.
	NProcessorInfoIsPopCntSupported [643]	
	NProcessorInfoIsPopCntSupportedEx [643]	Returns a Boolean value which indicates if PopCnt is supported for the particular processor model.
	NProcessorInfoIsSse2Supported [643]	
	NProcessorInfoIsSse2SupportedEx [644]	Returns a Boolean value which indicates if a Streaming SIMD Extension 2 (SSE2) instruction set is supported for a particular processor model.
	NProcessorInfoIsSse3Supported [644]	
	NProcessorInfoIsSse3SupportedEx [644]	Returns a Boolean value which indicates if a Streaming SIMD Extension 3 (SSE3) instruction set is supported for a particular processor model.
	NProcessorInfoIsSse41Supported [645]	
	NProcessorInfoIsSse41SupportedEx [645]	Returns a Boolean value which indicates if a Streaming SIMD Extension 4.1 (SSE41) instruction set is supported for a particular processor model.
	NProcessorInfoIsSse42Supported [645]	
	NProcessorInfoIsSse42SupportedEx [645]	Returns a Boolean value which indicates if a Streaming SIMD Extension 4.2 (SSE42) instruction set is supported for a particular processor model.
	NProcessorInfoIsSse4aSupported [646]	

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NProcessorInfoIsSse4aSupportedEx [646]	Returns a Boolean value which indicates if a Streaming SIMD Extension 4a (SSE4a) instruction set is supported for a particular processor model.
	NProcessorInfoIsSse5Supported [646]	
	NProcessorInfoIsSse5SupportedEx [647]	Returns a Boolean value which indicates if a Streaming SIMD Extension 5 (SSE5) instruction set is supported for a particular processor model.
	NProcessorInfoIsSseSupported [647]	
	NProcessorInfoIsSseSupportedEx [647]	Returns a Boolean value which indicates if a Streaming SIMD (SSE) instruction set is supported for a particular processor model.
	NProcessorInfoIsSsse3Supported [648]	
	NProcessorInfoIsSsse3SupportedEx [648]	Returns a Boolean value which indicates if a Supplemental Streaming SIMD Extension 3 (SSSE3) instruction set is supported for a particular processor model.

Macros

Name	Description
NProcessorInfoGetModelNameEx [649]	Picks a Unicode (W) or ASCII (A) version of the NProcessorInfoGetModelNameEx function.
NProcessorInfoGetVendorNameEx [649]	Picks a Unicode (W) or ASCII (A) version of the NProcessorInfoGetVendorNameEx function.

Module

NProcessorInfo Unit [637]

7.1.2.45 NReDeprecate Unit

7.1.2.46 NRWLock Unit

Functions

	Name	Description
	NRWLockDispose [651]	Disposes a reader/writer lock.
	NRWLockEnterRead [652]	Acquires a reader/writer lock in shared mode.
	NRWLockEnterWrite [652]	Acquires a reader/writer lock in exclusive mode.
	NRWLockExitRead [652]	Releases a reader/writer lock that was acquired in shared mode.
	NRWLockExitWrite [652]	Releases a reader/writer lock that was acquired in exclusive mode.
	NRWLockInit [653]	Initializes a reader/writer lock.
	NRWLockTryEnterRead [653]	Acquires a reader/writer lock in shared mode. A NBool [1004] value indicates whether the function succeeded.
	NRWLockTryEnterWrite [653]	Acquires a reader/writer lock in exclusive mode. A NBool [1004] value indicates whether the function succeeded.

7.1.2.46.1 Functions

7.1.2.46.1.1 NRWLockDispose Function

Disposes a reader/writer lock.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
void N_API NRWLockDispose(NRWLock * pLock);
```

Parameters

Parameters	Description
NRWLock * pLock	[in, out] A pointer to the RW lock.

Module

NRWLock Unit [651]

7.1.2.46.1.2 NRWLockEnterRead Function

Acquires a reader/writer lock in shared mode.

C++

```
void N_API NRWLockEnterRead(NRWLock * pLock);
```

Parameters

Parameters	Description
NRWLock * pLock	[in, out] A pointer to the RW lock.

Module

NRWLock Unit [651]

7.1.2.46.1.3 NRWLockEnterWrite Function

Acquires a reader/writer lock in exclusive mode.

C++

```
void N_API NRWLockEnterWrite(NRWLock * pLock);
```

Parameters

Parameters	Description
NRWLock * pLock	[in, out] A pointer to the RW lock.

Module

NRWLock Unit [651]

7.1.2.46.1.4 NRWLockExitRead Function

Releases a reader/writer lock that was acquired in shared mode.

C++

```
void N_API NRWLockExitRead(NRWLock * pLock);
```

Parameters

Parameters	Description
NRWLock * pLock	[in, out] A pointer to the RW lock.

Module

NRWLock Unit [651]

7.1.2.46.1.5 NRWLockExitWrite Function

Releases a reader/writer lock that was acquired in exclusive mode.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
void N_API NRWLockExitWrite(NRWLock * pLock);
```

Parameters

Parameters	Description
NRWLock * pLock	[in, out] A pointer to the RW lock.

Module

NRWLock Unit [651]

7.1.2.46.1.6 NRWLockInit Function

Initializes a reader/writer lock.

C++

```
void N_API NRWLockInit(NRWLock * pLock);
```

Parameters

Parameters	Description
NRWLock * pLock	[in, out] A pointer to the RW lock.

Module

NRWLock Unit [651]

7.1.2.46.1.7 NRWLockTryEnterRead Function

Acquires a reader/writer lock in shared mode. A NBool [1004] value indicates whether the function succeeded.

C++

```
NBool N_API NRWLockTryEnterRead(NRWLock * pLock);
```

Parameters

Parameters	Description
NRWLock * pLock	[in, out] A pointer to the RW lock.

Returns

NTrue [1064] if the function succeeded; else NFalse [1041].

Module

NRWLock Unit [651]

7.1.2.46.1.8 NRWLockTryEnterWrite Function

Acquires a reader/writer lock in exclusive mode. A NBool [1004] value indicates whether the function succeeded.

C++

```
NBool N_API NRWLockTryEnterWrite(NRWLock * pLock);
```

Parameters

Parameters	Description
NRWLock * pLock	[in, out] A pointer to the RW lock.

Returns

NTrue [1064] if the function succeeded; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NRWLock Unit [651]

7.1.2.47 NSemaphore Unit

Cross platform semaphore (synchronization primitive) implementation.

See Also

- NWaitObject [1123] for NSemaphore functionality shared with other synchronization objects.

Functions

	Name	Description
	NSemaphoreCreate [654]	Creates and initializes a semaphore object.
	NSemaphoreCreateFromOSHandle [654]	Creates a new NSemaphore for the specified semaphore handle.
	NSemaphoreGetOSHandle [655]	Gets OS handle for the specified semaphore.
	NSemaphoreRelease [655]	Increments the value of the semaphore by one.

7.1.2.47.1 Functions

7.1.2.47.1.1 NSemaphoreCreate Function

Creates and initializes a semaphore object.

C++

```
NResult N_API NSemaphoreCreate(NInt initialCount, HNSemaphore * phSemaphore);
```

Parameters

Parameters	Description
NInt initialCount	[in] The initial value of the semaphore (0: non signalled; positive: signalled).
HNSemaphore * phSemaphore	[out] Pointer to the variable that will receive the handle of the created object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This is usual synchronization semaphore. It's state is described by nonnegative number, the count of wait operations what will complete without blocking. The value is decremented on each successful return from NWaitObjectWaitFor [1123]/NWaitObjectTimedWaitFor and when it reaches zero, the threads performing wait operation will block until semaphore value is raised by other threads with NSemaphoreRelease [655] function.

Module

NSemaphore Unit [654]

7.1.2.47.1.2 NSemaphoreCreateFromOSHandle Function

Creates a new NSemaphore [654] for the specified semaphore handle.

C++

```
NResult N_API NSemaphoreCreateFromOSHandle(NHandle handle, NBool ownsHandle, HNSemaphore *
phSemaphore);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NHandle handle	[in] A semaphore handle for the semaphore that the current NSemaphore [654] will encapsulate.
NBool ownsHandle	[in] NTrue [1064] if the semaphore handle will be owned by this NSemaphore [654]; otherwise, NFalse [1041].
HNSemaphore * phSemaphore	[out] A pointer to the created NSemaphore [654].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSemaphore Unit [654]

7.1.2.47.1.3 NSemaphoreGetOSHandle Function

Gets OS handle for the specified semaphore.

C++

```
NResult N_API NSemaphoreGetOSHandle(HNSemaphore hSemaphore, NHandle * phValue);
```

Parameters

Parameters	Description
HNSemaphore hSemaphore	[in] A handle to NSemaphore [654].
NHandle * phValue	[out] OS handle to a semaphore.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSemaphore Unit [654]

7.1.2.47.1.4 NSemaphoreRelease Function

Increments the value of the semaphore by one.

C++

```
NResult N_API NSemaphoreRelease(HNSemaphore hSemaphore);
```

Parameters

Parameters	Description
HNSemaphore hSemaphore	[in] Handle of the semaphore.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

- The NSemaphore [654] is derived from NWaitObject [1123]. The NWaitObjectWaitFor [1123] or NWaitObjectTimedWaitFor should be used with the semaphore handle to wait until value can be decremented.
- If upon the call the value of the semaphore was 0 (nonsignalled) and some threads are blocked waiting for the semaphore to become signalled one thread will be woken and wait function will return successfully decrementing the value back to 0.

Module

NSemaphore Unit [654]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.48 NStopwatch Unit

Provides a set of methods and properties that you can use to accurately measure elapsed time.

Functions

	Name	Description
	NStopwatchCreate [656]	Creates a new NStopwatch object.
	NStopwatchGetElapsed [656]	Gets the total elapsed time measured by the specified NStopwatch.
	NStopwatchGetElapsedMilliseconds [657]	Gets the total elapsed time measured by the specified NStopwatch, in milliseconds.
	NStopwatchGetElapsedTicks [657]	Gets the total elapsed time measured by the specified NStopwatch, in timer ticks.
	NStopwatchGetFrequency [657]	Gets the frequency of the timer as the number of ticks per second.
	NStopwatchGetTimestamp [658]	Gets the current number of ticks in the timer mechanism.
	NStopwatchIsHighResolution [658]	Indicates whether the timer is based on a high-resolution performance counter.
	NStopwatchIsRunning [658]	Gets a value indicating whether the NStopwatch timer is running.
	NStopwatchReset [659]	Stops time interval measurement and resets the elapsed time to zero.
	NStopwatchRestart [659]	Stops time interval measurement, resets the elapsed time to zero, and starts measuring elapsed time.
	NStopwatchStart [659]	Starts, or resumes, measuring elapsed time for an interval.
	NStopwatchStartNew [660]	Creates a new NStopwatch object, sets the elapsed time property to zero, and starts measuring elapsed time.
	NStopwatchStop [660]	Stops measuring elapsed time for an interval.

7.1.2.48.1 Functions

7.1.2.48.1.1 NStopwatchCreate Function

Creates a new NStopwatch [656] object.

C++

```
NResult N_API NStopwatchCreate(HNStopwatch * phStopwatch);
```

Parameters

Parameters	Description
HNStopwatch * phStopwatch	[out] Pointer to a handle to the new NStopwatch [656] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.48.1.2 NStopwatchGetElapsed Function

Gets the total elapsed time measured by the specified NStopwatch [656].

C++

```
NResult N_API NStopwatchGetElapsed(HNStopwatch hStopwatch, NTimeSpan_ * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNStopwatch hStopwatch	[in] Handle to the NStopwatch [656].
NTimeSpan_ * pValue	[out] Pointer to the TimeSpan representing the elapsed time.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.48.1.3 NStopwatchGetElapsedMilliseconds Function

Gets the total elapsed time measured by the specified NStopwatch [656], in milliseconds.

C++

```
NResult N_API NStopwatchGetElapsedMilliseconds(HNStopwatch hStopwatch, NLong * pValue);
```

Parameters

Parameters	Description
HNStopwatch hStopwatch	[in] Handle to the NStopwatch [656].
NLong * pValue	[out] Pointer to the elapsed time in milliseconds.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.48.1.4 NStopwatchGetElapsedTicks Function

Gets the total elapsed time measured by the specified NStopwatch [656], in timer ticks.

C++

```
NResult N_API NStopwatchGetElapsedTicks(HNStopwatch hStopwatch, NLong * pValue);
```

Parameters

Parameters	Description
HNStopwatch hStopwatch	[in] Handle to the NStopwatch [656].
NLong * pValue	[out] Number of timer ticks representing the elapsed time.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.48.1.5 NStopwatchGetFrequency Function

Gets the frequency of the timer as the number of ticks per second.

C++

```
NResult N_API NStopwatchGetFrequency(NLong * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NLong * pValue	[out] Pointer to the number of ticks per second.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.48.1.6 NStopwatchGetTimestamp Function

Gets the current number of ticks in the timer mechanism.

C++

```
NResult N_API NStopwatchGetTimestamp(NLong * pValue);
```

Parameters

Parameters	Description
NLong * pValue	[out] Pointer to the tick counter value of the underlying timer mechanism.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.48.1.7 NStopwatchIsHighResolution Function

Indicates whether the timer is based on a high-resolution performance counter.

C++

```
NResult N_API NStopwatchIsHighResolution(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] Pointer to NBool [1004] value indicating whether the timer is based on a high-resolution performance counter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.48.1.8 NStopwatchIsRunning Function

Gets a value indicating whether the NStopwatch [656] timer is running.

C++

```
NResult N_API NStopwatchIsRunning(HNStopwatch hStopwatch, NBool * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNStopwatch hStopwatch	[in] Handle to the NStopwatch [656].
NBool * pValue	[out] NTrue [1064] if NStopwatch [656] is running; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.48.1.9 NStopwatchReset Function

Stops time interval measurement and resets the elapsed time to zero.

C++

```
NResult N_API NStopwatchReset(HNStopwatch hStopwatch);
```

Parameters

Parameters	Description
HNStopwatch hStopwatch	[in] Handle to the NStopwatch [656] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.48.1.10 NStopwatchRestart Function

Stops time interval measurement, resets the elapsed time to zero, and starts measuring elapsed time.

C++

```
NResult N_API NStopwatchRestart(HNStopwatch hStopwatch);
```

Parameters

Parameters	Description
HNStopwatch hStopwatch	[in] Handle to the NStopwatch [656] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.48.1.11 NStopwatchStart Function

Starts, or resumes, measuring elapsed time for an interval.

C++

```
NResult N_API NStopwatchStart(HNStopwatch hStopwatch);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNStopwatch hStopwatch	[in] Handle to the NStopwatch [656] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.48.1.12 NStopwatchStartNew Function

Creates a new NStopwatch [656] object, sets the elapsed time property to zero, and starts measuring elapsed time.

C++

```
NResult N_API NStopwatchStartNew(HNStopwatch * phStopwatch);
```

Parameters

Parameters	Description
HNStopwatch * phStopwatch	[out] Pointer to a handle to the new NStopwatch [656] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.48.1.13 NStopwatchStop Function

Stops measuring elapsed time for an interval.

C++

```
NResult N_API NStopwatchStop(HNStopwatch hStopwatch);
```

Parameters

Parameters	Description
HNStopwatch hStopwatch	[in] Handle to the NStopwatch [656] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStopwatch Unit [656]

7.1.2.49 NStream Unit

Supports internal Neurotechnology libraries infrastructure and should not be used directly in your code.

Functions

	Name	Description
	NStreamCanRead [661]	Gets a value indicating whether the current stream supports reading.
	NStreamCanSeek [661]	Gets a value indicating whether the current stream supports seeking.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NStreamCanWrite [662]	Gets a value indicating whether the current stream supports writing.
	NStreamCopyTo [662]	Reads the bytes from the current NStream and writes them to another NStream.
	NStreamCopyToWithBufferSize [662]	Reads the bytes from the current NStream and writes them to another NStream, using a specified buffer size.
	NStreamFlush [663]	Clears all buffers for this stream and causes any buffered data to be written to the underlying device.
	NStreamGetLength [663]	Gets the length in bytes of the NStream.
	NStreamGetNull [664]	Returns NStream with no backing store.
	NStreamGetPosition [664]	Gets the position within the current NStream.
	NStreamRead [664]	Reads a sequence of bytes from the NStream and advances the position within the stream by the number of bytes read.
	NStreamReadByte [665]	Reads a byte from NStream and advances the position within NStream by one byte, or returns -1 if at the end of NStream.
	NStreamReadN [665]	Reads a sequence of bytes from NBuffer [313] to NStream and advances the current position within NStream by the number of bytes read.
	NStreamSeek [665]	Sets the position within NStream.
	NStreamSetLength [666]	Sets the length in bytes of the NStream.
	NStreamSetPosition [666]	Sets the position within the current NStream.
	NStreamSynchronized [666]	Creates a thread-safe (synchronized) wrapper around the NStream.
	NStreamWrite [667]	Writes a sequence of bytes from an array to NStream and advances the current position within NStream by the number of bytes written.
	NStreamWriteByte [667]	Writes a byte to the current position in the stream and advances the position within the stream by one byte.
	NStreamWriteN [667]	Writes a sequence of bytes from NBuffer [313] to NStream and advances the current position within NStream by the number of bytes written.

7.1.2.49.1 Functions

7.1.2.49.1.1 NStreamCanRead Function

Gets a value indicating whether the current stream supports reading.

C++

```
NResult N_API NStreamCanRead(HNStream hStream, NBool * pValue);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
NBool * pValue	[out] A pointer to NTrue [1064] if the NStream [660] supports reading; otherwise, to NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.2 NStreamCanSeek Function

Gets a value indicating whether the current stream supports seeking.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStreamCanSeek(HNStream hStream, NBool * pValue);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
NBool * pValue	[out] A pointer to NTrue [1064] if the NStream [660] supports seeking; otherwise, to NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.3 NStreamCanWrite Function

Gets a value indicating whether the current stream supports writing.

C++

```
NResult N_API NStreamCanWrite(HNStream hStream, NBool * pValue);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
NBool * pValue	[out] A pointer to NTrue [1064] if the NStream [660] supports writing; otherwise, to NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.4 NStreamCopyTo Function

Reads the bytes from the current NStream [660] and writes them to another NStream [660].

C++

```
NResult N_API NStreamCopyTo(HNStream hStream, HNStream hDstStream);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to the current NStream [660].
HNStream hDstStream	[in] A handle to the destination NStream [660].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.5 NStreamCopyToWithBufferSize Function

Reads the bytes from the current NStream [660] and writes them to another NStream [660], using a specified buffer size.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStreamCopyToWithBufferSize(HNStream hStream, HNStream hDstStream, NSizeType bufferSize);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to the current NStream [660].
HNStream hDstStream	[in] A handle to the destination NStream [660].
NSizeType bufferSize	[in] The size of the buffer. This value must be greater than zero. The default size is 4096.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.6 NStreamFlush Function

Clears all buffers for this stream and causes any buffered data to be written to the underlying device.

C++

```
NResult N_API NStreamFlush(HNStream hStream);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.7 NStreamGetLength Function

Gets the length in bytes of the NStream [660].

C++

```
NResult N_API NStreamGetLength(HNStream hStream, NLong * pValue);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
NLong * pValue	[out] A pointer to a long value representing the length of the NStream [660] in bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.49.1.8 NStreamGetNull Function

Returns NStream [660] with no backing store.

C++

```
NResult N_API NStreamGetNull(HNStream * phValue);
```

Parameters

Parameters	Description
HNStream * phValue	[out] A pointer to NStream [660] with no backing store.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NStream Unit [660]

7.1.2.49.1.9 NStreamGetPosition Function

Gets the position within the current NStream [660].

C++

```
NResult N_API NStreamGetPosition(HNStream hStream, NLong * pValue);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
NLong * pValue	[out] A pointer to the current position within NStream [660].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.10 NStreamRead Function

Reads a sequence of bytes from the NStream [660] and advances the position within the stream by the number of bytes read.

C++

```
NResult N_API NStreamRead(HNStream hStream, void * pBuffer, NSizeType bufferSize, NSizeType
* pSizeRead);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
void * pBuffer	[in] A pointer to NBuffer [313] to write the data into.
NSizeType bufferSize	[in] Size of NBuffer [313] in bytes.
NSizeType * pSizeRead	[out] A pointer to the number of bytes read.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.11 NStreamReadByte Function

Reads a byte from NStream [660] and advances the position within NStream [660] by one byte, or returns -1 if at the end of NStream [660].

C++

```
NResult N_API NStreamReadByte(HNStream hStream, NInt * pValue);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
NInt * pValue	[out] A pointer to the unsigned byte cast to a NInt32 [1006], or -1 if at the end of NStream [660].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.12 NStreamReadN Function

Reads a sequence of bytes from NBuffer [313] to NStream [660] and advances the current position within NStream [660] by the number of bytes read.

C++

```
NResult N_API NStreamReadN(HNStream hStream, HNBuffer hBuffer, NSizeType * pSizeRead);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
HNBuffer hBuffer	[in] A handle to NBuffer [313] to write the data into.
NSizeType * pSizeRead	[out] A pointer to the number of bytes read.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.13 NStreamSeek Function

Sets the position within NStream [660].

C++

```
NResult N_API NStreamSeek(HNStream hStream, NLong offset, NSeekOrigin origin);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
NLong offset	[in] A byte offset relative to the origin parameter.
NSeekOrigin origin	[in] A value of type NSeekOrigin [497] indicating the reference point used to obtain the new position.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.14 NStreamSetLength Function

Sets the length in bytes of the NStream [660].

C++

```
NResult N_API NStreamSetLength(HNStream hStream, NLong value);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
NLong value	[in] A long value representing the length of the NStream [660] in bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.15 NStreamSetPosition Function

Sets the position within the current NStream [660].

C++

```
NResult N_API NStreamSetPosition(HNStream hStream, NLong value);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
NLong value	[in] The position within NStream [660].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.16 NStreamSynchronized Function

Creates a thread-safe (synchronized) wrapper around the NStream [660].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStreamSynchronized(HNStream hStream, HNStream * phValue);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660] to synchronize
HNStream * phValue	[out] A pointer to synchronized NStream [660].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.17 NStreamWrite Function

Writes a sequence of bytes from an array to NStream [660] and advances the current position within NStream [660] by the number of bytes written.

C++

```
NResult N_API NStreamWrite(HNStream hStream, const void * pBuffer, NSizeType bufferSize);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
const void * pBuffer	[in] A pointer to an array of bytes.
NSizeType bufferSize	[in] The size of the array of bytes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.18 NStreamWriteByte Function

Writes a byte to the current position in the stream and advances the position within the stream by one byte.

C++

```
NResult N_API NStreamWriteByte(HNStream hStream, NByte value);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
NByte value	[in] The byte to write to the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.49.1.19 NStreamWriteN Function

Writes a sequence of bytes from NBuffer [313] to NStream [660] and advances the current position within NStream

Copyright © 2009-2014 Neurotechnology. All rights reserved.

[660] by the number of bytes written.

C++

```
NResult N_API NStreamWriteN(HNStream hStream, HNBuffer hBuffer);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to NStream [660].
HNBuffer hBuffer	[in] A handle to NBuffer [313] to write data from.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStream Unit [660]

7.1.2.50 NStreamReader Unit

Implements a TextReader that reads characters from a byte stream in a particular encoding.

Functions

	Name	Description
	NStreamReaderCreate [669]	Creates a new NStreamReader class for the specified NStream [660].
	NStreamReaderCreateFromFile [669]	Creates a new NStreamReader for the specified file.
	NStreamReaderCreateFromFileWithEncoding [669]	Creates a new NStreamReader for the specified file with specified encoding.
	NStreamReaderCreateFromFileWithEncodingAndBufferSize [670]	Creates a new NStreamReader for the specified file with specified encoding and buffer size.
	NStreamReaderCreateWithEncoding [670]	Creates a new NStreamReader for the specified NStream [660] with specified encoding.
	NStreamReaderCreateWithEncodingAndBufferSize [670]	Creates a new NStreamReader for the specified NStream [660] with specified encoding and buffer size.
	NStreamReaderGetBaseStream [671]	Returns the underlying stream of the specified stream reader.
	NStreamReaderGetCurrentEncoding [671]	Gets the current encoding of the specified stream reader.
	NStreamReaderGetNull [672]	Creates a NStreamReader object around an empty stream.
	NStreamReaderIsEndOfStream [672]	Checks whether the current stream position is at the end of the stream.

Macros

Name	Description
NStreamReaderCreateFromFile [672]	Picks a Unicode (W) or ASCII (A) version of the NStreamReaderCreateFromFile function.
NStreamReaderCreateFromFileWithEncoding [672]	Picks a Unicode (W) or ASCII (A) version of the NStreamReaderCreateFromFileWithEncoding function.
NStreamReaderCreateFromFileWithEncodingAndBufferSize [673]	Picks a Unicode (W) or ASCII (A) version of the NStreamReaderCreateFromFileWithEncodingAndBufferSize function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.50.1 Functions

7.1.2.50.1.1 NStreamReaderCreate Function

Creates a new NStreamReader [668] class for the specified NStream [660].

C++

```
NResult N_API NStreamReaderCreate(HNStream hStream, HNStreamReader * phReader);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to the NStream [660].
HNStreamReader * phReader	[out] Pointer to a handle to the new NStreamReader [668].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamReader Unit [668]

7.1.2.50.1.2 NStreamReaderCreateFromFile Function

Creates a new NStreamReader [668] for the specified file.

C++

```
NResult N_API NStreamReaderCreateFromFile(const NChar * szFileName, HNStreamReader *
phReader);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Pointer to the name of the file.
HNStreamReader * phReader	[out] Pointer to a handle to the new NStreamReader [668].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamReader Unit [668]

7.1.2.50.1.3 NStreamReaderCreateFromFileWithEncoding Function

Creates a new NStreamReader [668] for the specified file with specified encoding.

C++

```
NResult N_API NStreamReaderCreateFromFileWithEncoding(const NChar * szFileName, NEncoding
encoding, NBool detectEncodingFromByteOrderMarks, HNStreamReader * phReader);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Pointer to the name of the file.
NEncoding encoding	[in] The encoding.
NBool detectEncodingFromByteOrderMarks	[in] NTrue [1064] if the encoding should be detected from byte order marks; else NFalse [1041].
HNStreamReader * phReader	[out] Pointer to a handle to the new NStreamReader [668].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamReader Unit [668]

7.1.2.50.1.4 NStreamReaderCreateFromFileWithEncodingAndBufferSize Function

Creates a new NStreamReader [668] for the specified file with specified encoding and buffer size.

C++

```
NResult N_API NStreamReaderCreateFromFileWithEncodingAndBufferSize(const NChar *
szFileName, NEncoding encoding, NBool detectEncodingFromByteOrderMarks, NSizeType
bufferSize, HNStreamReader * phReader);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Pointer to the name of the file.
NEncoding encoding	[in] The encoding.
NBool detectEncodingFromByteOrderMarks	[in] NTrue [1064] if the encoding should be detected from byte order marks; else NFalse [1041].
NSizeType bufferSize	[in] The buffer size.
HNStreamReader * phReader	[out] Pointer to a handle to the new NStreamReader [668].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamReader Unit [668]

7.1.2.50.1.5 NStreamReaderCreateWithEncoding Function

Creates a new NStreamReader [668] for the specified NStream [660] with specified encoding.

C++

```
NResult N_API NStreamReaderCreateWithEncoding(HNStream hStream, NEncoding encoding, NBool
detectEncodingFromByteOrderMarks, HNStreamReader * phReader);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to the NStream [660].
NEncoding encoding	[in] The encoding.
NBool detectEncodingFromByteOrderMarks	[in] NTrue [1064] if the encoding should be detected from byte order marks; else NFalse [1041].
HNStreamReader * phReader	[out] Pointer to a handle to the new NStreamReader [668].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamReader Unit [668]

7.1.2.50.1.6 NStreamReaderCreateWithEncodingAndBufferSize Function

Creates a new NStreamReader [668] for the specified NStream [660] with specified encoding and buffer size.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStreamReaderCreateWithEncodingAndBufferSize(HNStream hStream, NEncoding encoding, NBool detectEncodingFromByteOrderMarks, NSizeType bufferSize, HNStreamReader * phReader);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to the NStream [660].
NEncoding encoding	[in] The encoding.
NBool detectEncodingFromByteOrderMarks	[in] NTrue [1064] if the encoding should be detected from byte order marks; else NFalse [1041].
NSizeType bufferSize	[in] The buffer size.
HNStreamReader * phReader	[out] Pointer to a handle to the new NStreamReader [668].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamReader Unit [668]

7.1.2.50.1.7 NStreamReaderGetBaseStream Function

Returns the underlying stream of the specified stream reader.

C++

```
NResult N_API NStreamReaderGetBaseStream(HNStreamReader hReader, HNStream * phValue);
```

Parameters

Parameters	Description
HNStreamReader hReader	[in] Handle to the NStreamReader [668].
HNStream * phValue	[out] Pointer to a handle to the underlying NStream [660].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamReader Unit [668]

7.1.2.50.1.8 NStreamReaderGetCurrentEncoding Function

Gets the current encoding of the specified stream reader.

C++

```
NResult N_API NStreamReaderGetCurrentEncoding(HNStreamReader hReader, NEncoding * pValue);
```

Parameters

Parameters	Description
HNStreamReader hReader	[in] Handle to the NStreamReader [668].
NEncoding * pValue	[out] Pointer to the encoding of the stream reader.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamReader Unit [668]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.50.1.9 NStreamReaderGetNull Function

Creates a NStreamReader [668] object around an empty stream.

C++

```
NResult N_API NStreamReaderGetNull(HNStreamReader * phValue);
```

Parameters

Parameters	Description
HNStreamReader * phValue	[out] Pointer to a handle to the NStreamReader [668].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamReader Unit [668]

7.1.2.50.1.10 NStreamReaderIsEndOfStream Function

Checks whether the current stream position is at the end of the stream.

C++

```
NResult N_API NStreamReaderIsEndOfStream(HNStreamReader hReader, NBool * pValue);
```

Parameters

Parameters	Description
HNStreamReader hReader	[in] Handle to the NStreamReader [668].
NBool * pValue	[out] Pointer to a value indicating whether the current stream position is at the end of the stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamReader Unit [668]

7.1.2.50.2 Macros

7.1.2.50.2.1 NStreamReaderCreateFromFile Macro

Picks a Unicode (W) or ASCII (A) version of the NStreamReaderCreateFromFile function.

C++

```
#define NStreamReaderCreateFromFile N_FUNC_AW(NStreamReaderCreateFromFile)
```

Module

NStreamReader Unit [668]

7.1.2.50.2.2 NStreamReaderCreateFromFileWithEncoding Macro

Picks a Unicode (W) or ASCII (A) version of the NStreamReaderCreateFromFileWithEncoding function.

C++

```
#define NStreamReaderCreateFromFileWithEncoding
N_FUNC_AW(NStreamReaderCreateFromFileWithEncoding)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStreamReader Unit [668]

7.1.2.50.2.3 NStreamReaderCreateFromFileWithEncodingAndBufferSize Macro

Picks a Unicode (W) or ASCII (A) version of the NStreamReaderCreateFromFileWithEncodingAndBufferSize function.

C++

```
#define NStreamReaderCreateFromFileWithEncodingAndBufferSize
N_FUNC_AW(NStreamReaderCreateFromFileWithEncodingAndBufferSize)
```

Module

NStreamReader Unit [668]

7.1.2.51 NStreamWriter Unit

Implements a TextWriter for writing characters to a stream in a particular encoding.

Functions

	Name	Description
	NStreamWriterCreate [674]	Creates a new NStreamWriter for the specified NStream [660].
	NStreamWriterCreateFromFile [674]	Creates a new NStreamWriter for the specified file.
	NStreamWriterCreateFromFileWithEncoding [674]	Creates a new NStreamWriter for the specified file with specified encoding.
	NStreamWriterCreateFromFileWithEncodingAndBufferSize [675]	Creates a new NStreamWriter for the specified file with specified encoding and buffer size.
	NStreamWriterCreateWithEncoding [675]	Creates a new NStreamWriter for the specified NStream [660] with specified encoding.
	NStreamWriterCreateWithEncodingAndBufferSize [676]	Creates a new NStreamWriter for the specified NStream [660] with specified encoding and buffer size.
	NStreamWriterGetAutoFlush [676]	Gets a value indicating whether the StreamWriter will flush its buffer to the underlying stream after every call to StreamWriter.Write.
	NStreamWriterGetBaseStream [676]	Returns the underlying stream of the specified stream writer.
	NStreamWriterGetNull [677]	Creates a NStreamWriter object around an empty stream.
	NStreamWriterSetAutoFlush [677]	Sets a value indicating whether the StreamWriter will flush its buffer to the underlying stream after every call to StreamWriter.Write.

Macros

Name	Description
NStreamWriterCreateFromFile [677]	Picks a Unicode (W) or ASCII (A) version of the NStreamWriterCreateFromFile function.
NStreamWriterCreateFromFileWithEncoding [678]	Picks a Unicode (W) or ASCII (A) version of the NStreamWriterCreateFromFileWithEncoding function.
NStreamWriterCreateFromFileWithEncodingAndBufferSize [678]	Picks a Unicode (W) or ASCII (A) version of the NStreamWriterCreateFromFileWithEncodingAndBufferSize function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.51.1 Functions

7.1.2.51.1.1 NStreamWriterCreate Function

Creates a new NStreamWriter [673] for the specified NStream [660].

C++

```
NResult N_API NStreamWriterCreate(HNStream hStream, HNStreamWriter * phWriter);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to the NStream [660].
HNStreamWriter * phWriter	[out] Pointer to a handle to the new NStreamWriter [673].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamWriter Unit [673]

7.1.2.51.1.2 NStreamWriterCreateFromFile Function

Creates a new NStreamWriter [673] for the specified file.

C++

```
NResult N_API NStreamWriterCreateFromFile(const NChar * szFileName, NBool append,
HNStreamWriter * phWriter);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Pointer to the name of the file.
NBool append	[in] NTrue [1064] if the writer should append byte to the file; else NFalse [1041].
HNStreamWriter * phWriter	[out] Pointer to a handle to the new NStreamWriter [673].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamWriter Unit [673]

7.1.2.51.1.3 NStreamWriterCreateFromFileWithEncoding Function

Creates a new NStreamWriter [673] for the specified file with specified encoding.

C++

```
NResult N_API NStreamWriterCreateFromFileWithEncoding(const NChar * szFileName, NBool
append, NEncoding encoding, HNStreamWriter * phWriter);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Pointer to the name of the file.
NBool append	[in] NTrue [1064] if the writer should append byte to the file; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NEncoding encoding	[in] The encoding.
HNStreamWriter * phWriter	[out] Pointer to a handle to the new NStreamWriter [673].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamWriter Unit [673]

7.1.2.51.1.4 NStreamWriterCreateFromFileWithEncodingAndBufferSize Function

Creates a new NStreamWriter [673] for the specified file with specified encoding and buffer size.

C++

```
NResult N_API NStreamWriterCreateFromFileWithEncodingAndBufferSize(const NChar *
szFileName, NBool append, NEncoding encoding, NSizeType bufferSize, HNStreamWriter *
phWriter);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Pointer to the name of the file.
NBool append	[in] NTrue [1064] if the writer should append byte to the file; else NFalse [1041].
NEncoding encoding	[in] The encoding.
NSizeType bufferSize	[in] The buffer size.
HNStreamWriter * phWriter	[out] Pointer to a handle to the new NStreamWriter [673].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamWriter Unit [673]

7.1.2.51.1.5 NStreamWriterCreateWithEncoding Function

Creates a new NStreamWriter [673] for the specified NStream [660] with specified encoding.

C++

```
NResult N_API NStreamWriterCreateWithEncoding(HNStream hStream, NEncoding encoding,
HNStreamWriter * phWriter);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to the NStream [660].
NEncoding encoding	[in] The encoding.
HNStreamWriter * phWriter	[out] Pointer to a handle to the new NStreamWriter [673].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamWriter Unit [673]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.51.1.6 NStreamWriterCreateWithEncodingAndBufferSize Function

Creates a new NStreamWriter [673] for the specified NStream [660] with specified encoding and buffer size.

C++

```
NResult N_API NStreamWriterCreateWithEncodingAndBufferSize(HNStream hStream, NEncoding encoding, NSizeType bufferSize, HNStreamWriter * phWriter);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to the NStream [660].
NEncoding encoding	[in] The encoding.
NSizeType bufferSize	[in] The buffer size.
HNStreamWriter * phWriter	[out] Pointer to a handle to the new NStreamWriter [673].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamWriter Unit [673]

7.1.2.51.1.7 NStreamWriterGetAutoFlush Function

Gets a value indicating whether the StreamWriter will flush its buffer to the underlying stream after every call to StreamWriter.Write.

C++

```
NResult N_API NStreamWriterGetAutoFlush(HNStreamWriter hWriter, NBool * pValue);
```

Parameters

Parameters	Description
HNStreamWriter hWriter	[in] Handle to the NStreamWriter [673].
NBool * pValue	[out] NTrue [1064] if the NStreamWriter [673] flushes its buffer; otherwise, NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamWriter Unit [673]

7.1.2.51.1.8 NStreamWriterGetBaseStream Function

Returns the underlying stream of the specified stream writer.

C++

```
NResult N_API NStreamWriterGetBaseStream(HNStreamWriter hWriter, HNStream * phValue);
```

Parameters

Parameters	Description
HNStreamWriter hWriter	[in] Handle to the NStreamWriter [673].
HNStream * phValue	_nt_

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStreamWriter Unit [673]

7.1.2.51.1.9 NStreamWriterGetNull Function

Creates a NStreamWriter [673] object around an empty stream.

C++

```
NResult N_API NStreamWriterGetNull(HNStreamWriter * phValue);
```

Parameters

Parameters	Description
HNStreamWriter * phValue	[out] Pointer to a handle to the new NStreamWriter [673].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamWriter Unit [673]

7.1.2.51.1.10 NStreamWriterSetAutoFlush Function

Sets a value indicating whether the StreamWriter will flush its buffer to the underlying stream after every call to StreamWriter.Write.

C++

```
NResult N_API NStreamWriterSetAutoFlush(HNStreamWriter hWriter, NBool value);
```

Parameters

Parameters	Description
HNStreamWriter hWriter	[in] Handle to the NStreamWriter [673].
NBool value	[in] NTrue [1064] to force NStreamWriter [673] to flush its buffer; otherwise, NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStreamWriter Unit [673]

7.1.2.51.2 Macros

7.1.2.51.2.1 NStreamWriterCreateFromFile Macro

Picks a Unicode (W) or ASCII (A) version of the NStreamWriterCreateFromFile function.

C++

```
#define NStreamWriterCreateFromFile N_FUNC_AW(NStreamWriterCreateFromFile)
```

Module

NStreamWriter Unit [673]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.51.2.2 NStreamWriterCreateFromFileWithEncoding Macro

Picks a Unicode (W) or ASCII (A) version of the NStreamWriterCreateFromFileWithEncoding function.

C++

```
#define NStreamWriterCreateFromFileWithEncoding
N_FUNC_AW(NStreamWriterCreateFromFileWithEncoding)
```

Module

NStreamWriter Unit [673]

7.1.2.51.2.3 NStreamWriterCreateFromFileWithEncodingAndBufferSize Macro

Picks a Unicode (W) or ASCII (A) version of the NStreamWriterCreateFromFileWithEncodingAndBufferSize function.

C++

```
#define NStreamWriterCreateFromFileWithEncodingAndBufferSize
N_FUNC_AW(NStreamWriterCreateFromFileWithEncodingAndBufferSize)
```

Module

NStreamWriter Unit [673]

7.1.2.52 NString Unit

Provides functionality for working with strings.

Files

Name	Description
NString.h [762]	Header file for the NString unit.

Functions

	Name	Description
	NStringAppendStrOrChars [687]	Extends specified string content by adding an additional string or chars at its end.
	NStringClone [687]	Clones specified NString.
	NStringCompareRangeNA [688]	Compares substrings of the two specified strings and retrieves their relative position in the sort order.
	NStringCompareRangeNW [688]	Compares substrings of the two specified strings and retrieves their relative position in the sort order.
	NStringCompareRangeStrOrCharsA [689]	Compares substrings of a string and a string or a charset and retrieves their relative position in the sort order.
	NStringCompareRangeStrOrCharsW [689]	Compares substrings of a string and a string or a charset and retrieves their relative position in the sort order.
	NStringConcatArray [690]	Concatenates strings specified by array.
	NStringConcatMany [690]	Concatenates specified strings.
	NStringConcatManyVA [691]	Concatenates specified strings.
	NStringConcatN [691]	Concatenates strings specified by handles.
	NStringConcatStrOrChars [692]	Concatenates strings or charsets specified by handles.
	NStringContainsStrN [692]	Checks whether specified string occurs within other string.
	NStringContainsStrOrChars [692]	Returns a value indicating whether specified chars or string occurs within string specified by handle.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NStringCopyToStrOrChars [693]	Copies a specified number of characters from a specified position in handle to a specified array.
	NStringCreateFromChar [693]	Creates string from specified character.
	NStringCreateFromStrOrChars [694]	Creates string from specified array of characters.
	NStringCreateWrapper [694]	Creates a new string handle based on the specified string.
	NStringEndsWithN [694]	Checks whether the end of the string matches the specified string.
	NStringEndsWithStrOrCharsA [695]	Checks whether the end of the string matches the specified string or charset.
	NStringEndsWithStrOrCharsW [695]	Checks whether the end of the string matches the specified string or charset.
	NStringEqualsN [696]	Checks whether two specified strings have the same value.
	NStringEqualsStrOrCharsA [696]	Checks whether two specified strings or charsets have the same value.
	NStringEqualsStrOrCharsW [696]	Checks whether two specified strings or charsets have the same value.
	NStringFormat [697]	Returns a string representing the specified format and parameters.
	NStringFormatVA [697]	Returns a string representing the specified format and arguments.
	NStringFree [698]	Deletes the string specified by handle. After the string is deleted the specified handle is no longer valid.
	NStringFreeArray [698]	Deletes array of strings specified by handles. After the strings is deleted the specified object handles are no longer valid.
	NStringFreeElements [698]	Deletes strings specified by handles in array. After the strings is deleted the specified string handles are no longer valid.
	NStringGetBuffer [698]	Gets characters buffer (string) from specified handle to NString.
	NStringGetEmpty [699]	Gets empty string.
	NStringGetHashCode [699]	Gets the hash code
	NStringGetLength [699]	Gets length of specified string.
	NStringHasEmbeddedNull [700]	Checked whether the specified string has an embedded null value.
	NStringIndexOf [700]	Returns the zero-based index of the first occurrence of the specified character in the string.
	NStringIndexOfAny [701]	Returns the zero-based index of the first occurrence in the string of any character in a specified array of characters.
	NStringIndexOfAnyFrom [701]	Returns the zero-based index of the first occurrence in the string of any character in a specified array of characters.
	NStringIndexOfAnyInRange [701]	Returns the zero-based index of the first occurrence in the string of any character in a specified array of characters.
	NStringIndexOfFrom [702]	Returns the zero-based index of the first occurrence of the specified character in the string starting from specified index.
	NStringIndexOfInRange [702]	Returns the zero-based index of the first character's occurrence in string's range.
	NStringIndexOfStrFromN [703]	Returns the zero-based index of the first occurrence of the specified string in the string starting from specified index.
	NStringIndexOfStrInRangeN [703]	Returns the zero-based index of the first occurrence of the specified string in the string's range.
	NStringIndexOfStrN [704]	Returns the zero-based index of the first occurrence of the specified NString in the string.
	NStringIndexOfStrOrChars [704]	Returns the zero-based index of the first occurrence of the specified string or character in the string.
	NStringIndexOfStrOrCharsFrom [705]	Returns the zero-based index of the first occurrence of the specified string or character in the string.
	NStringIndexOfStrOrCharsInRange [705]	Returns the zero-based index of the first occurrence of the specified string or character in the string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NStringInsertN [706]	Returns a new string in which a specified string is inserted at a specified index position.
	NStringInsertStrOrChars [706]	Returns a new string in which a specified string or characters is inserted at a specified index position.
	NStringIsA [706]	Checks whether specified string contains ANSI characters.
	NStringIsEmpty [707]	Checks whether specified string is empty string.
	NStringIsW [707]	Checks whether specified string contains Unicode characters.
	NStringIsWhiteSpace [707]	Checks whether specified string consists only of white-space characters.
	NStringJoinArray [708]	Concatenates all the elements of a string array, using the specified separator between each element.
	NStringJoinArrayVN [708]	Concatenates all the elements of a string array, using the specified separator between each element.
	NStringJoinMany [709]	Concatenates all the elements of a string arrays, using the specified separator between each element.
	NStringJoinManyVA [709]	Concatenates all the elements of a string arrays, using the specified separator between each element.
	NStringJoinManyVAVN [709]	Concatenates all the elements of a string arrays, using the specified separator between each element.
	NStringJoinManyVN [710]	Concatenates all the elements of a string arrays, using the specified separator between each element.
	NStringLastIndexOf [710]	Returns the zero-based index position of the last occurrence of a specified character within string.
	NStringLastIndexOfAny [710]	Returns the zero-based index position of the last occurrence in the string of one or more characters specified in an array.
	NStringLastIndexOfAnyFrom [711]	Returns the zero-based index position of the last occurrence in the string of one or more characters specified in an array.
	NStringLastIndexOfAnyInRange [711]	Returns the zero-based index position of the last occurrence in the string's range of one or more characters specified in an array.
	NStringLastIndexOfFrom [712]	Returns the zero-based index of the last occurrence of the specified character in the string starting from specified index.
	NStringLastIndexOfInRange [712]	Returns the zero-based index of the last character's occurrence in string's range.
	NStringLastIndexOfStrFromN [713]	Returns the zero-based index of the last occurrence of the specified string in the NString starting from specified index.
	NStringLastIndexOfStrInRangeN [713]	Returns the zero-based index of the last occurrence of the specified string in the NString in specified range.
	NStringLastIndexOfStrN [714]	Returns the zero-based index of the last occurrence of the specified NString in the string.
	NStringLastIndexOfStrOrChars [714]	Returns the zero-based index of the last occurrence of the specified string or character in the string.
	NStringLastIndexOfStrOrCharsFrom [715]	Returns the zero-based index of the last occurrence of the specified string or character in the string.
	NStringLastIndexOfStrOrCharsInRange [715]	Returns the zero-based index of the last occurrence of the specified string or character in the string.
	NStringPadLeft [716]	Returns a new string that right-aligns the characters by padding them with spaces on the left, for a specified total length.
	NStringPadLeftWith [716]	Returns a new string that right-aligns the characters by padding them on the left with a specified character, for a specified total length.
	NStringPadRight [717]	Returns a new string that left-aligns the characters by padding them with spaces on the right, for a specified total length.
	NStringPadRightWith [717]	Returns a new string that left-aligns the characters by padding them on the right with a specified character, for a specified total length.
	NStringPrependStrOrChars [718]	Prepends a string or a charset to another string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NStringPromoteBuffer [718]	Creates a new NString which takes ownership of the memory buffer.
	NStringRemove [718]	Returns a new string in which all the characters in *hString*, beginning at a specified position and continuing through the last position, have been deleted.
	NStringRemoveFrom [719]	Returns a new string in which all the characters in *hString*, beginning at a specified position have been deleted.
	NStringReplace [719]	Returns a new string in which all occurrences of a specified character in *hString* are replaced with another specified character.
	NStringReplaceStrOrChars [720]	Replaces a part of a string or a char array with new values.
	NStringSet [720]	Sets the value of the specified string.
	NStringSplitWithCount [720]	Returns a string array that contains the substrings that are delimited by elements of a specified character array. A parameter specifies the maximum number of substrings to return.
	NStringSplitWithStrsAndCount [721]	Returns a string array that contains the substrings that are delimited by elements of a specified character array. A parameter specifies the maximum number of substrings to return.
	NStringSplitWithStrsAndCountN [721]	Returns a string array that contains the substrings that are delimited by elements of a specified string array. A parameter specifies the maximum number of substrings to return.
	NStringStartsWithN [722]	Checks whether the beginning of *hString* matches the specified string.
	NStringStartsWithStrOrCharsA [722]	Checks whether the beginning of *hString* matches the specified string or chars.
	NStringStartsWithStrOrCharsW [723]	Checks whether the beginning of *hString* matches the specified string or chars.
	NStringSubstring [723]	Gets a substring from *hString*. The substring starts at a specified character position.
	NStringSubstringFrom [724]	Gets a substring from *hString*. The substring starts at a specified character position.
	NStringToLower [724]	Converts specified string to lowercase.
	NStringToString [724]	Converts the specified string to another string with specified format.
	NStringToStringOld [725]	Converts the specified string to charset.
	NStringToStrOrCharArray [725]	Converts a string to another string or a character array.
	NStringToStrOrCharArrayRange [726]	Converts a string to another string or a part of character array.
	NStringToUpper [726]	Converts specified string to uppercase.
	NStringTrim [726]	Removes all leading and trailing white-space characters from specified string.
	NStringTrimAny [727]	Removes all leading and trailing occurrences of a sets of characters from specified string.
	NStringTrimEnd [727]	Removes all trailing characters from *hString*.
	NStringTrimEndAny [727]	Removes all trailing occurrences of a set of characters specified in an array from *hString*.
	NStringTrimStart [728]	Removes all trailing characters from *hString*.
	NStringTrimStartAny [728]	Removes all leading occurrences of a set of characters specified in an array from *hString*.

Macros

Name	Description
N_NEW_LINE [729]	Picks a Unicode (W) or ASCII (A) version of the N_NEW_LINE macro.
N_STRING_SIZE [729]	Defines the string size.
NStringAppend [729]	Extends specified string content by adding an additional string at its end.
NStringAppendChars [729]	Extends specified string content by adding additional chars at its end.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringAppendStrOrChars [730]	Picks a Unicode (W) or ASCII (A) version of the NStringAppendStrOrChars function.
NStringCompare [730]	Compares two strings and retrieves their relative position in the sort order.
NStringCompareChars [730]	Compares a string with a char array and retrieves their relative position in the sort order.
NStringCompareN [731]	Compares two strings and retrieves their relative position in the sort order.
NStringCompareRange [731]	Compares substrings of the two specified strings and retrieves their relative position in the sort order.
NStringCompareRangeChars [732]	Compares substrings of a string and a charset and retrieves their relative position in the sort order.
NStringCompareRangeN [732]	Picks a Unicode (W) or ASCII (A) version of the NStringCompareRangeN function.
NStringCompareRangeStrOrChars [732]	Picks a Unicode (W) or ASCII (A) version of the NStringCompareRangeStrOrChars function.
NStringCompareStrOrChars [732]	Compares a string with a string or char array and retrieves their relative position in the sort order.
NStringConcat [733]	Concatenates strings specified by handles.
NStringConcatArray [733]	Picks a Unicode (W) or ASCII (A) version of the NStringConcatArray function.
NStringConcatChars [733]	Concatenates charsets specified by handles.
NStringConcatMany [734]	Picks a Unicode (W) or ASCII (A) version of the NStringConcatMany function.
NStringConcatManyVA [734]	Picks a Unicode (W) or ASCII (A) version of the NStringConcatManyVA function.
NStringConcatStrOrChars [734]	Picks a Unicode (W) or ASCII (A) version of the NStringConcatStrOrChars function.
NStringContainsChars [734]	Returns a value indicating whether specified chars occur within string specified by handle.
NStringContainsStr [735]	Returns a value indicating whether specified string occurs within string specified by handle.
NStringContainsStrOrChars [735]	Picks a Unicode (W) or ASCII (A) version of the NStringContainsStrOrChars function.
NStringCopyTo [735]	Copies a specified number of characters from a specified position in handle to a specified array.
NStringCopyToString [736]	Copies a specified number of characters from a specified position in handle to a specified string.
NStringCopyToStrOrChars [736]	Picks a Unicode (W) or ASCII (A) version of the NStringCopyToStrOrChars function.
NStringCreate [736]	Creates a string with the specified value.
NStringCreateFromChar [737]	Picks a Unicode (W) or ASCII (A) version of the NStringCreateFromChar function.
NStringCreateFromChars [737]	Creates a string from the specified array of characters.
NStringCreateFromStrOrChars [737]	Picks a Unicode (W) or ASCII (A) version of the NStringCreateFromStrOrChars function.
NStringCreateWrapper [738]	Picks a Unicode (W) or ASCII (A) version of the NStringCreateWrapper function.
NStringEndsWith [738]	Checks whether the end of the string matches the specified string.
NStringEndsWithChars [738]	Checks whether the end of the string matches the specified string or charset.
NStringEndsWithStrOrChars [739]	Picks a Unicode (W) or ASCII (A) version of the NStringEndsWithStrOrChars function.
NStringEquals [739]	Checks whether two specified stings have the same value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringEqualsChars [739]	Checks whether the specified string and the specified charset have the same value.
NStringEqualsStrOrChars [740]	Picks a Unicode (W) or ASCII (A) version of the NStringEqualsStrOrChars function.
NStringFormat [740]	Picks a Unicode (W) or ASCII (A) version of the NStringFormat function.
NStringFormatVA [740]	Picks a Unicode (W) or ASCII (A) version of the NStringFormatVA function.
NStringGetBuffer [740]	Picks a Unicode (W) or ASCII (A) version of the NStringGetBuffer function.
NStringGetChar [740]	Picks a Unicode (W) or ASCII (A) version of the NStringGetChar function.
NStringGetLength [740]	Picks a Unicode (W) or ASCII (A) version of the NStringGetLength function.
NStringIndexOf [741]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOf function.
NStringIndexOfAny [741]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfAny function.
NStringIndexOfAnyFrom [741]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfAnyFrom function.
NStringIndexOfAnyInRange [741]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfAnyInRange function.
NStringIndexOfChars [741]	Returns the zero-based index of the first occurrence of the specified character in the string.
NStringIndexOfCharsFrom [742]	Returns the zero-based index of the first occurrence of the specified character in the string.
NStringIndexOfCharsInRange [742]	Returns the zero-based index of the first occurrence of the specified character in the string.
NStringIndexOfFrom [743]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfFrom function.
NStringIndexOfInRange [743]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfInRange function.
NStringIndexOfStr [743]	Returns the zero-based index of the first occurrence of the specified string in the string.
NStringIndexOfStrFrom [744]	Returns the zero-based index of the first occurrence of the specified string in the string.
NStringIndexOfStrFromN [744]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrFromN function.
NStringIndexOfStrInRange [744]	Returns the zero-based index of the first occurrence of the specified string in the string.
NStringIndexOfStrInRangeN [745]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrInRangeN function.
NStringIndexOfStrN [745]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrN function.
NStringIndexOfStrOrChars [745]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrOrChars function.
NStringIndexOfStrOrCharsFrom [745]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrOrCharsFrom function.
NStringIndexOfStrOrCharsInRange [745]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrOrCharsInRange function.
NStringInsert [745]	Returns a new string in which a specified string is inserted at a specified index position.
NStringInsertChars [746]	Returns a new string in which a specified characters are inserted at a specified index position.
NStringInsertN [746]	Picks a Unicode (W) or ASCII (A) version of the NStringInsertN function.
NStringInsertStrOrChars [746]	Picks a Unicode (W) or ASCII (A) version of the NStringInsertStrOrChars function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringJoinArray [747]	Picks a Unicode (W) or ASCII (A) version of the NStringJoinArray function.
NStringJoinArrayVN [747]	Picks a Unicode (W) or ASCII (A) version of the NStringJoinArrayVN function.
NStringJoinMany [747]	Picks a Unicode (W) or ASCII (A) version of the NStringJoinMany function.
NStringJoinManyVAVN [747]	Picks a Unicode (W) or ASCII (A) version of the NStringJoinManyVAVN function.
NStringJoinManyVN [747]	Picks a Unicode (W) or ASCII (A) version of the NStringJoinManyVN function.
NStringLastIndexOf [748]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOf function.
NStringLastIndexOfAny [748]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfAny function.
NStringLastIndexOfAnyFrom [748]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfAnyFrom function.
NStringLastIndexOfAnyInRange [748]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfAnyInRange function.
NStringLastIndexOfChars [748]	Returns the zero-based index of the last occurrence of the specified character in the string.
NStringLastIndexOfCharsFrom [749]	Returns the zero-based index of the last occurrence of the specified character in the string.
NStringLastIndexOfCharsInRange [749]	Returns the zero-based index of the last occurrence of the specified character in the string.
NStringLastIndexOfFrom [750]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfFrom function.
NStringLastIndexOfInRange [750]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfInRange function.
NStringLastIndexOfStr [750]	Returns the zero-based index of the last occurrence of the specified string in the string.
NStringLastIndexOfStrFrom [750]	Returns the zero-based index of the last occurrence of the specified string in the string.
NStringLastIndexOfStrFromN [751]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrFromN function.
NStringLastIndexOfStrInRange [751]	Returns the zero-based index of the last occurrence of the specified string in the string.
NStringLastIndexOfStrInRangeN [752]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrInRangeN function.
NStringLastIndexOfStrN [752]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrN function.
NStringLastIndexOfStrOrChars [752]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrOrChars function.
NStringLastIndexOfStrOrCharsFrom [752]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrOrCharsFrom function.
NStringLastIndexOfStrOrCharsInRange [752]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrOrCharsInRange function.
NStringPadLeft [752]	Picks a Unicode (W) or ASCII (A) version of the NStringPadLeft function.
NStringPadLeftWith [753]	Picks a Unicode (W) or ASCII (A) version of the NStringPadLeftWith function.
NStringPadRight [753]	Picks a Unicode (W) or ASCII (A) version of the NStringPadRight function.
NStringPadRightWith [753]	Picks a Unicode (W) or ASCII (A) version of the NStringPadRightWith function.
NStringPrepend [753]	Prepends a string to another string.
NStringPrependChars [754]	Prepends a charset to another string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringPrependStrOrChars [754]	Picks a Unicode (W) or ASCII (A) version of the NStringPrependStrOrChars function.
NStringPromoteBuffer [754]	Picks a Unicode (W) or ASCII (A) version of the NStringPromoteBuffer function.
NStringRemove [754]	Picks a Unicode (W) or ASCII (A) version of the NStringRemove function.
NStringRemoveFrom [754]	Picks a Unicode (W) or ASCII (A) version of the NStringRemoveFrom function.
NStringReplace [755]	Picks a Unicode (W) or ASCII (A) version of the NStringReplace function.
NStringReplaceChars [755]	Replaces a part of a char array with new values.
NStringReplaceStr [755]	Replaces a part of a string with new values.
NStringReplaceStrOrChars [756]	Picks a Unicode (W) or ASCII (A) version of the NStringReplaceStrOrChars function.
NStringSplit [756]	Returns a string array that contains the substrings that are delimited by elements of a specified character array.
NStringSplitWithCount [756]	Picks a Unicode (W) or ASCII (A) version of the NStringSplitWithCount function.
NStringSplitWithStrs [757]	Returns a string array that contains the substrings that are delimited by elements of a specified character array.
NStringSplitWithStrsAndCount [757]	Picks a Unicode (W) or ASCII (A) version of the NStringSplitWithStrsAndCount function.
NStringSplitWithStrsN [757]	Returns a string array that contains the substrings that are delimited by elements of a specified string array.
NStringStartsWith [758]	Checks whether the beginning of *hString* matches the specified string.
NStringStartsWithChars [758]	Checks whether the beginning of *hString* matches the specified charset.
NStringStartsWithStrOrChars [759]	Picks a Unicode (W) or ASCII (A) version of the NStringStartsWithStrOrChars function.
NStringSubstring [759]	Picks a Unicode (W) or ASCII (A) version of the NStringSubstring function.
NStringSubstringFrom [759]	Picks a Unicode (W) or ASCII (A) version of the NStringSubstringFrom function.
NStringToCharArray [759]	Converts a string to character array.
NStringToCharArrayRange [759]	Converts a string to a part of character array.
NStringToNewString [760]	Converts a string to another string.
NStringToNewStringRange [760]	Converts a string to another string or a part of character array.
NStringToString [761]	Picks a Unicode (W) or ASCII (A) version of the NStringToString function.
NStringToStringOld [761]	Picks a Unicode (W) or ASCII (A) version of the NStringToStringOld function.
NStringToStrOrCharArray [761]	Picks a Unicode (W) or ASCII (A) version of the NStringToStrOrCharArray function.
NStringToStrOrCharArrayRange [761]	Picks a Unicode (W) or ASCII (A) version of the NStringToStrOrCharArrayRange function.
NStringTrimAny [761]	Picks a Unicode (W) or ASCII (A) version of the NStringTrimAny function.
NStringTrimEndAny [762]	Picks a Unicode (W) or ASCII (A) version of the NStringTrimEndAny function.
NStringTrimStartAny [762]	Picks a Unicode (W) or ASCII (A) version of the NStringTrimStartAny function.

7.1.2.52.1 Format Specifications

Format Specifications

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Format line takes the form of {type[,length]:formatString}

- *type* component defines the type of the value formatted. It is a mandatory component. A list of all types used in formatting can be found in NString.c
- *length* component defines the minimum number of characters in the string representation of the parameter. If positive, the parameter is right-aligned; if negative, it is left-aligned. It as an optional component.
- *formatString* component represents a format string that is appropriate for the type of object being formatted. It is a mandatory component.
-

Text format strings

NAChar [1004], NWChar, const NAChar [1004] *, const NWChar *, HNString ignore format strings and as such, format string need be supplied for their formatting.

Boolean format strings

NBoolean [1004] uses format string of type *X/Y*, where *X* is a string for NTrue [1064] and *Y* is a string for NFalse [1041].

Polymorphic format strings

NObject [536], NValue [1094] - format strings used depend on the type of the object/value formatted. Refer to NXxx for the list of supported format strings.

Numeric format strings

- are used to format numeric types (NInt [1006]#, NUInt [1009]#, NSingle [1008], NDouble [1005], NSizeType [1008] and NSSizeType [1009]; NPointer is formatted as numeric, however, "G" for it equals X8/X16 (for 32-bit/64-bit); NResult [1007] is formatted as NInt32 [1006] but ignores the format strings).

(# = 8/16/32/64)

They consist of a single alphabetic character called the format specifier, an optional integer (maximum two digits), called the precision specifier. The precision specifier ranges from 0 to 99 and affects the number of digits in the result.

Standard format specifiers are listed below.

Format specifier	Name	Description
"C" or "c"	Currency	Results in a currency value. Supported by all numeric types. Precision specifier indicates the number of decimal digits.
"D" or "d"	Decimal	Results in integer digits with optional negative sign. Supported by integral types only. Precision specifier indicates the minimum number of digits.
"E" or "e"	Scientific	Results in an exponential notation. Supported by all numeric types. Precision specifier indicates the number of decimal digits.
"F" or "f"	Fixed-point	Results in integral and decimal digits with optional negative sign. Supported by all numeric types. Precision specifier indicates the number of decimal digits.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

"G" or "g"	General	Results in the most compact of either fixed-point or scientific notation. Supported by all numeric types. Precision specifier indicates the number of significant digits.
"N" or "n"	Number	Results in integral and decimal digits, group separators, and a decimal separator with optional negative sign. Supported by all numeric types. Precision specifier indicates the desired number of decimal places.
"P" or "p"	Percent	Results in the number multiplied by 100 and displayed with a percent symbol. Supported by all numeric types. Precision specifier indicates the desired number of decimal places.
"R" or "r"	Round-trip	Results in a string that can round-trip to an identical number. Supported by NSingle [1008] and NDouble [1005]. Precision specifier is ignored and not required.
"X" or "x"	Hexadecimal	Results in a hexadecimal string. Supported by integral types only. Precision specifier indicates the number of digits in the result string.
Other	Unknown	Throws a FormatException at runtime.

Module

NString Unit [678]

7.1.2.52.2 Functions

7.1.2.52.2.1 NStringAppendStrOrChars Function

Extends specified string content by adding an additional string or chars at its end.

C++

```
NResult N_API NStringAppendStrOrChars(HNString hString, const NChar * arValue, NInt
valueLength, HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678] which should be appended.
const NChar * arValue	[in] Array with a sequence of characters. These characters will be appended at the end of *hString*.
NInt valueLength	[in] Size of *arValue* memory buffer.
HNString * phResultString	[out] Result string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.2 NStringClone Function

Clones specified NString [678].

C++

```
NResult N_API NStringClone(HNString hString, HNString * phResultString);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678] which should be cloned.
HNString * phResultString	[out] Instance of NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.3 NStringCompareRangeNA Function

Compares substrings of the two specified strings and retrieves their relative position in the sort order.

C++

```
NResult N_API NStringCompareRangeNA(HNString hString, NInt index, HNString hOtherString,
NInt otherIndex, NInt length, NBool ignoreCase, NInt * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to the first string to use in comparison.
NInt index	[in] Position of the substring in *hString.*
HNString hOtherString	[in] Handle to the second string to use in comparison.
NInt otherIndex	[in] Position of the substring in *hOtherString.*
NInt length	[in] The maximum number of characters in the substrings to compare.
NBool ignoreCase	[in] NTrue [1064] to ignore letter case during comparison; else NFalse [1041].
NInt * pResult	[out] Pointer to NInt [1006] value that indicates the lexical relationship between the two comparands.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This is an ANSCII version of the function.

Module

NString Unit [678]

7.1.2.52.2.4 NStringCompareRangeNW Function

Compares substrings of the two specified strings and retrieves their relative position in the sort order.

C++

```
NResult N_API NStringCompareRangeNW(HNString hString, NInt index, HNString hOtherString,
NInt otherIndex, NInt length, NBool ignoreCase, NInt * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to the first string to use in comparison.
NInt index	[in] Position of the substring in *hString.*
HNString hOtherString	[in] Handle to the second string to use in comparison.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt otherIndex	[in] Position of the substring in *hOtherString.*
NInt length	[in] The maximum number of characters in the substrings to compare.
NBool ignoreCase	[in] NTrue [1064] to ignore letter case during comparison; else NFalse [1041].
NInt * pResult	[out] Pointer to NInt [1006] value that indicates the lexical relationship between the two comparands.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This is a Unicode version of the function.

Module

NString Unit [678]

7.1.2.52.2.5 NStringCompareRangeStrOrCharsA Function

Compares substrings of a string and a string or a charset and retrieves their relative position in the sort order.

C++

```
NResult N_API NStringCompareRangeStrOrCharsA(HNString hString, NInt index, const NAChar *
arValue, NInt valueLength, NInt length, NBool ignoreCase, NInt * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to the string to use in comparison.
NInt index	[in] Position of the substring in *hString.*
const NAChar * arValue	[in] Handle to the charset to use in comparison.
NInt valueLength	[in] Length of the charset.
NInt length	[in] The maximum number of characters in the substrings to compare.
NBool ignoreCase	[in] NTrue [1064] to ignore letter case during comparison; else NFalse [1041].
NInt * pResult	[out] Pointer to NInt [1006] value that indicates the lexical relationship between the two comparands.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This is an ANSCII version of the function.

Module

NString Unit [678]

7.1.2.52.2.6 NStringCompareRangeStrOrCharsW Function

Compares substrings of a string and a string or a charset and retrieves their relative position in the sort order.

C++

```
NResult N_API NStringCompareRangeStrOrCharsW(HNString hString, NInt index, const NWChar *
arValue, NInt valueLength, NInt length, NBool ignoreCase, NInt * pResult);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hString	[in] Handle to the string to use in comparison.
NInt index	[in] Position of the substring in *hString.*
const NWChar * arValue	[in] Handle to the charset to use in comparison.
NInt valueLength	[in] Length of the charset.
NInt length	[in] The maximum number of characters in the substrings to compare.
NBool ignoreCase	[in] NTrue [1064] to ignore letter case during comparison; else NFalse [1041].
NInt * pResult	[out] Pointer to NInt [1006] value that indicates the lexical relationship between the two comparands.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This is a Unicode version of the function.

Module

NString Unit [678]

7.1.2.52.2.7 NStringConcatArray Function

Concatenates strings specified by array.

C++

```
NResult N_API NStringConcatArray(const NChar * const * arszValues, NInt count, HNString *
phResultString);
```

Parameters

Parameters	Description
const NChar * const * arszValues	[in] String array to concatenate.
NInt count	[in] *arszValues* array items count.
HNString * phResultString	[out] The concatenated elements of *arszValues*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

The function concatenates each *arszValues* element; it does not add any delimiters.

Module

NString Unit [678]

7.1.2.52.2.8 NStringConcatMany Function

Concatenates specified strings.

C++

```
NResult N_API NStringConcatMany(HNString * phResultString, NInt count, ...
multipleParameters);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString * phResultString	[out] The concatenated elements.
NInt count	[in] NChar [1005] elements count.
... multipleParameters	[in] Function parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This function must specify as many `const NChar [1005] *` elements as specified by *count* parameter.

Module

NString Unit [678]

7.1.2.52.2.9 NStringConcatManyVA Function

Concatenates specified strings.

C++

```
NResult N_API NStringConcatManyVA(HNString * phResultString, NInt count, va_list args);
```

Parameters

Parameters	Description
HNString * phResultString	[out] The concatenated elements.
NInt count	[in] NChar [1005] elements count.
va_list args	[in] Function arguments.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.10 NStringConcatN Function

Concatenates strings specified by handles.

C++

```
NResult N_API NStringConcatN(HNString hValue1, HNString hValue2, HNString * phResultString);
```

Parameters

Parameters	Description
HNString hValue1	[in] Handle to first string to concatenate.
HNString hValue2	[in] Handle to second string to concatenate.
HNString * phResultString	[out] The concatenated strings.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.2.11 NStringConcatStrOrChars Function

Concatenates strings or charsets specified by handles.

C++

```
NResult N_API NStringConcatStrOrChars(const NChar * arValue1, NInt value1Length, const
NChar * arValue2, NInt value2Length, HNString * phResultString);
```

Parameters

Parameters	Description
const NChar * arValue1	[in] Pointer to the first string or charset.
NInt value1Length	[in] Length of the first string or charset.
const NChar * arValue2	[in] Pointer to the second string or charset.
NInt value2Length	[in] Length of the second string or charset.
HNString * phResultString	[out] Pointer to a handle to the resulting string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.12 NStringContainsStrN Function

Checks whether specified string occurs within other string.

C++

```
NResult N_API NStringContainsStrN(HNString hString, HNString hValue, NBool * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] String specified by handle where *hValue* string will be sought.
HNString hValue	[in] String specified by handle to seek.
NBool * pResult	[out] Pointer to *hValue* seek result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.13 NStringContainsStrOrChars Function

Returns a value indicating whether specified chars or string occurs within string specified by handle.

C++

```
NResult N_API NStringContainsStrOrChars(HNString hString, const NChar * arValue, NInt
valueLength, NBool * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678] to check.
const NChar * arValue	[in] NChar [1005] array which contains chars to check in hString.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt valueLength	[in] Length of memory buffer where *arValue* is stored.
NBool * pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.14 NStringCopyToStrOrChars Function

Copies a specified number of characters from a specified position in handle to a specified array.

C++

```
NResult N_API NStringCopyToStrOrChars(HNString hString, NInt sourceIndex, NChar * arValue,
NInt valueLength, NBool nullTerminate, NInt count);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678] which contains characters to copy.
NInt sourceIndex	[in] The index of the first character in this handle to copy.
NChar * arValue	[out] Pointer to memory buffer which contains copied characters.
NInt valueLength	[in] Size of *arValue* memory buffer
NBool nullTerminate	[in] If NTrue [1064], null terminated string will be copied.
NInt count	[in] The number of characters in *hString* to copy to *arValue*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.15 NStringCreateFromChar Function

Creates string from specified character.

C++

```
NResult N_API NStringCreateFromChar(NChar value, NInt count, HNString * phString);
```

Parameters

Parameters	Description
NChar value	[in] Character which will be used to create the string.
NInt count	[in] Characters count in string. If count > 1, characters will be duplicated in the string.
HNString * phString	[out] Pointer to handle which contains created string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.2.16 NStringCreateFromStrOrChars Function

Creates string from specified array of characters.

C++

```
NResult N_API NStringCreateFromStrOrChars(const NChar * arValue, NInt length, HNString *
phString);
```

Parameters

Parameters	Description
const NChar * arValue	[in] Array of chars.
NInt length	[in] Size of *arValue* memory buffer.
HNString * phString	[out] Pointer to handle which contains created string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.17 NStringCreateWrapper Function

Creates a new string handle based on the specified string.

C++

```
NResult N_API NStringCreateWrapper(const NChar * szValue, NInt length, NStringHeader *
pStringHeader, HNString * phString);
```

Parameters

Parameters	Description
const NChar * szValue	[in] Source array of characters (string) for new handle.
NInt length	[in] The length of source array.
NStringHeader * pStringHeader	[out] Pointer to NStringHeader which represents header for NString [678].
HNString * phString	[out] Pointer to newly created string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.18 NStringEndsWithN Function

Checks whether the end of the string matches the specified string.

C++

```
NResult N_API NStringEndsWithN(HNString hString, HNString hOtherString, NBool ignoreCase,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to string to check.
HNString hOtherString	[in] Handle to the string to compare to the *hString*.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBool * pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.19 NStringEndsWithStrOrCharsA Function

Checks whether the end of the string matches the specified string or charset.

C++

```
NResult N_API NStringEndsWithStrOrCharsA(HNString hString, const NAChar * arValue, NInt
valueLength, NBool ignoreCase, NBool * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to string to check.
const NAChar * arValue	[in] Pointer to the char array to compare to the *hString*.
NInt valueLength	[in] Length of *arValue.*
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NBool * pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This is an ANSCII version of the function.

Module

NString Unit [678]

7.1.2.52.2.20 NStringEndsWithStrOrCharsW Function

Checks whether the end of the string matches the specified string or charset.

C++

```
NResult N_API NStringEndsWithStrOrCharsW(HNString hString, const NWChar * arValue, NInt
valueLength, NBool ignoreCase, NBool * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to string to check.
const NWChar * arValue	[in] Pointer to the char array to compare to the *hString*.
NInt valueLength	[in] Length of *arValue.*
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NBool * pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This is a Unicode version of the function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NString Unit [678]

7.1.2.52.2.21 NStringEqualsN Function

Checks whether two specified strings have the same value.

C++

```
NResult N_API NStringEqualsN(HNString hString, HNString hOtherString, NBool ignoreCase,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to the first string to compare.
HNString hOtherString	[in] Handle to the second string to compare.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NBool * pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.22 NStringEqualsStrOrCharsA Function

Checks whether two specified strings or charsets have the same value.

C++

```
NResult N_API NStringEqualsStrOrCharsA(HNString hString, const NAChar * arValue, NInt
valueLength, NBool ignoreCase, NBool * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to the string to compare.
const NAChar * arValue	[in] Pointer to the char array to compare.
NInt valueLength	[in] Length of *arValue.*
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NBool * pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This is an ANSCII version of the function.

Module

NString Unit [678]

7.1.2.52.2.23 NStringEqualsStrOrCharsW Function

Checks whether two specified strings or charsets have the same value.

C++

```
NResult N_API NStringEqualsStrOrCharsW(HNString hString, const NWChar * arValue, NInt
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
valueLength, NBool ignoreCase, NBool * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to the string to compare.
const NWChar * arValue	[in] Pointer to the char array to compare.
NInt valueLength	[in] Length of *arValue.*
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NBool * pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This is a Unicode version of the function.

Module

NString Unit [678]

7.1.2.52.2.24 NStringFormat Function

Returns a string representing the specified format and parameters.

C++

```
NResult N_API NStringFormat(HNString * phValue, const NChar * szFormat, ...
multipleParameters);
```

Parameters

Parameters	Description
HNString * phValue	[out] Pointer to a handle to NString [678].
const NChar * szFormat	[in] Pointer to composite format string.
... multipleParameters	[in] Format parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.25 NStringFormatVA Function

Returns a string representing the specified format and arguments.

C++

```
NResult N_API NStringFormatVA(HNString * phValue, const NChar * szFormat, va_list args);
```

Parameters

Parameters	Description
HNString * phValue	[out] Pointer to a handle to NString [678].
const NChar * szFormat	[in] Pointer to composite format string.
va_list args	[in] Format argument list.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NString Unit [678]

7.1.2.52.2.26 NStringFree Function

Deletes the string specified by handle. After the string is deleted the specified handle is no longer valid.

C++

```
void N_API NStringFree(HNString hString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to string to be deleted.

Module

NString Unit [678]

7.1.2.52.2.27 NStringFreeArray Function

Deletes array of strings specified by handles. After the strings is deleted the specified object handles are no longer valid.

C++

```
void N_API NStringFreeArray(HNString * arhStrings, NInt count);
```

Parameters

Parameters	Description
HNString * arhStrings	[in] Handle to strings array to be deleted.
NInt count	[in] Strings array count.

Module

NString Unit [678]

7.1.2.52.2.28 NStringFreeElements Function

Deletes strings specified by handles in array. After the strings is deleted the specified string handles are no longer valid.

C++

```
void N_API NStringFreeElements(HNString * arhStrings, NInt count);
```

Parameters

Parameters	Description
HNString * arhStrings	[in] Handle to strings array to be deleted.
NInt count	[in] *arhStrings* array count.

Remarks

This function does not free array where strings are saved. If you need to dispose all array use NStringtFreeArray.

Module

NString Unit [678]

7.1.2.52.2.29 NStringGetBuffer Function

Gets characters buffer (string) from specified handle to NString [678].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStringGetBuffer(HNString hString, NInt * pLength, const NChar * * pszBuffer);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678] to retrieve string buffer.
NInt * pLength	[out] Size of *pszBuffer*.
const NChar * * pszBuffer	[out] Pointer to zero terminated string from *hString*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.30 NStringGetEmpty Function

Gets empty string.

C++

```
NResult N_API NStringGetEmpty(HNString * phValue);
```

Parameters

Parameters	Description
HNString * phValue	[out] Handle to empty string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.31 NStringGetHashCode Function

Gets the hash code

C++

```
NResult N_API NStringGetHashCode(HNString hString, NInt * pHashCode);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678] to retrieve its hash code.
NInt * pHashCode	[out] String hash code.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.32 NStringGetLength Function

Gets length of specified string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStringGetLength(HNString hString, NInt * pValue);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NInt * pValue	[out] String length.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.33 NStringHasEmbeddedNull Function

Checked whether the specified string has an embedded null value.

C++

```
NResult N_API NStringHasEmbeddedNull(HNString hString, NBool * pValue);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to the string.
NBool * pValue	[out] Pointer to NTrue [1064] if the string has an embedded null; else pointer to NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.34 NStringIndexOf Function

Returns the zero-based index of the first occurrence of the specified character in the string.

C++

```
NResult N_API NStringIndexOf(HNString hString, NChar value, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NChar value	[in] Character to seek.
NInt * pIndex	[out] The zero-based index position of value if that character is found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.2.35 NStringIndexOfAny Function

Returns the zero-based index of the first occurrence in the string of any character in a specified array of characters.

C++

```
NResult N_API NStringIndexOfAny(HNString hString, const NChar * arAnyOf, NInt anyOfLength,
NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arAnyOf	[in] Character array containing one or more characters to seek.
NInt anyOfLength	[in] Length of *arAnyOf* array.
NInt * pIndex	[out] The zero-based index position of the first occurrence in this string where any character in *arAnyOf* was found; -1 if no character in *arAnyOf* was found. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.36 NStringIndexOfAnyFrom Function

Returns the zero-based index of the first occurrence in the string of any character in a specified array of characters.

C++

```
NResult N_API NStringIndexOfAnyFrom(HNString hString, const NChar * arAnyOf, NInt
anyOfLength, NInt startIndex, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arAnyOf	[in] Character array containing one or more characters to seek.
NInt anyOfLength	[in] Length of *arAnyOf* array.
NInt startIndex	[in] Seek start index.
NInt * pIndex	[out] The zero-based index position of the first occurrence in this string where any character in *arAnyOf* was found; -1 if no character in *arAnyOf* was found. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.37 NStringIndexOfAnyInRange Function

Returns the zero-based index of the first occurrence in the string of any character in a specified array of characters.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStringIndexOfAnyInRange(HNString hString, const NChar * arAnyOf, NInt anyOfLength, NInt startIndex, NInt count, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arAnyOf	[in] Character array containing one or more characters to seek.
NInt anyOfLength	[in] Length of *arAnyOf* array.
NInt startIndex	[in] Seek start index.
NInt count	[in] Characters count (range to perform search).
NInt * pIndex	[out] The zero-based index position of the first occurrence in this string where any character in *arAnyOf* was found; -1 if no character in *arAnyOf* was found. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.38 NStringIndexOfFrom Function

Returns the zero-based index of the first occurrence of the specified character in the string starting from specified index.

C++

```
NResult N_API NStringIndexOfFrom(HNString hString, NChar value, NInt startIndex, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NChar value	[in] Character to seek.
NInt startIndex	[in] Seek start index in *hString*.
NInt * pIndex	[out] The zero-based index position of value if that character is found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.39 NStringIndexOfInRange Function

Returns the zero-based index of the first character's occurrence in string's range.

C++

```
NResult N_API NStringIndexOfInRange(HNString hString, NChar value, NInt startIndex, NInt count, NInt * pIndex);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NChar value	[in] Character to seek.
NInt startIndex	[in] Seek start index.
NInt count	[in] Characters count (range to perform search).
NInt * pIndex	[out] The zero-based index position of the first occurrence in this string range where specified character was found; -1 if no character was found. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.40 NStringIndexOfStrFromN Function

Returns the zero-based index of the first occurrence of the specified string in the string starting from specified index.

C++

```
NResult N_API NStringIndexOfStrFromN(HNString hString, HNString hValue, NInt startIndex,
NBool ignoreCase, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
HNString hValue	[in] Handle to string to seek.
NInt startIndex	[in] Seek start index in *hString*.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NInt * pIndex	[out] The zero-based index position of *hValue* if that string is found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.41 NStringIndexOfStrInRangeN Function

Returns the zero-based index of the first occurrence of the specified string in the string's range.

C++

```
NResult N_API NStringIndexOfStrInRangeN(HNString hString, HNString hValue, NInt startIndex,
NInt count, NBool ignoreCase, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
HNString hValue	[in] Handle to string to seek.
NInt startIndex	[in] Seek start index in *hString*.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt count	[in] *hString* range to seek (characters count starting with *startIndex*).
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NInt * pIndex	[out] The zero-based index position of *hValue* if that string is found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.42 NStringIndexOfStrN Function

Returns the zero-based index of the first occurrence of the specified NString [678] in the string.

C++

```
NResult N_API NStringIndexOfStrN(HNString hString, HNString hValue, NBool ignoreCase, NInt
* pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
HNString hValue	[in] NString [678] to seek.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NInt * pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.43 NStringIndexOfStrOrChars Function

Returns the zero-based index of the first occurrence of the specified string or character in the string.

C++

```
NResult N_API NStringIndexOfStrOrChars(HNString hString, const NChar * arValue, NInt
valueLength, NBool ignoreCase, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arValue	[in] NChar [1005] array which contains chars or string to check in *hString*.
NInt valueLength	[in] Size of *arValue* memory buffer.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NInt * pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.44 NStringIndexOfStrOrCharsFrom Function

Returns the zero-based index of the first occurrence of the specified string or character in the string.

C++

```
NResult N_API NStringIndexOfStrOrCharsFrom(HNString hString, const NChar * arValue, NInt
valueLength, NInt startIndex, NBool ignoreCase, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arValue	[in] NChar [1005] array which contains chars or string to check in *hString*.
NInt valueLength	[in] Size of *arValue* memory buffer.
NInt startIndex	[in] Seek start index.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NInt * pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.45 NStringIndexOfStrOrCharsInRange Function

Returns the zero-based index of the first occurrence of the specified string or character in the string.

C++

```
NResult N_API NStringIndexOfStrOrCharsInRange(HNString hString, const NChar * arValue, NInt
valueLength, NInt startIndex, NInt count, NBool ignoreCase, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arValue	[in] NChar [1005] array which contains chars or string to check in *hString*.
NInt valueLength	[in] Size of *arValue* memory buffer.
NInt startIndex	[in] Seek start index.
NInt count	[in] Range to perform search.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NInt * pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.46 NStringInsertN Function

Returns a new string in which a specified string is inserted at a specified index position.

C++

```
NResult N_API NStringInsertN(HNString hString, NInt index, HNString hValue, HNString *
phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NInt index	[in] The zero-based index position of the insertion.
HNString hValue	[in] NString [678] to insert.
HNString * phResultString	[out] Pointer to new string that is equivalent to *hString*, but with *hValue* inserted at position *index*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.47 NStringInsertStrOrChars Function

Returns a new string in which a specified string or characters is inserted at a specified index position.

C++

```
NResult N_API NStringInsertStrOrChars(HNString hString, NInt index, const NChar * arValue,
NInt valueLength, HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NInt index	[in] The zero-based index position of the insertion.
const NChar * arValue	[in] NChar [1005] array containing chars or string to insert.
NInt valueLength	[in] Size of *arValue* memory buffer.
HNString * phResultString	[out] Pointer to new string that is equivalent to *hString*, but with *hValue* inserted at position *index*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.48 NStringIsA Function

Checks whether specified string contains ANSI characters.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NBool N_API NStringIsA(HNString hString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.49 NStringIsEmpty Function

Checks whether specified string is empty string.

C++

```
NBool N_API NStringIsEmpty(HNString hString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.50 NStringIsW Function

Checks whether specified string contains Unicode characters.

C++

```
NBool N_API NStringIsW(HNString hString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.51 NStringIsWhiteSpace Function

Checks whether specified string consists only of white-space characters.

C++

```
NResult N_API NStringIsWhiteSpace(HNString hString, NBool * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NBool * pValue	[out] Check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.52 NStringJoinArray Function

Concatenates all the elements of a string array, using the specified separator between each element.

C++

```
NResult N_API NStringJoinArray(const NChar * szSeparator, const NChar * const * arszValues,
NInt count, HNString * phResultString);
```

Parameters

Parameters	Description
const NChar * szSeparator	[in] Zero terminated string to use as a separator.
const NChar * const * arszValues	[in] String array to concatenate.
NInt count	[in] Length of the array.
HNString * phResultString	[out] Pointer to a handle to the resulting string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.53 NStringJoinArrayVN Function

Concatenates all the elements of a string array, using the specified separator between each element.

C++

```
NResult N_API NStringJoinArrayVN(const NChar * szSeparator, const HNString * arhValues,
NInt count, HNString * phResultString);
```

Parameters

Parameters	Description
const NChar * szSeparator	[in] Zero terminated string to use as a separator.
const HNString * arhValues	[in] String handle array to concatenate.
NInt count	[in] Length of the array.
HNString * phResultString	[out] Pointer to a handle to the resulting string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.2.54 NStringJoinMany Function

Concatenates all the elements of a string arrays, using the specified separator between each element.

C++

```
NResult N_API NStringJoinMany(const NChar * szSeparator, HNString * phResultString, NInt
count, ... multipleParameters);
```

Parameters

Parameters	Description
const NChar * szSeparator	[in] Zero terminated string to use as a separator.
HNString * phResultString	[out] The concatenated elements of specified strings.
NInt count	[in] Array items count.
... multipleParameters	[in] Function parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.55 NStringJoinManyVA Function

Concatenates all the elements of a string arrays, using the specified separator between each element.

C++

```
NResult N_API NStringJoinManyVA(const NChar * szSeparator, HNString * phResultString, NInt
count, va_list args);
```

Parameters

Parameters	Description
const NChar * szSeparator	[in] Zero terminated string to use as a separator.
HNString * phResultString	[out] The concatenated elements of specified strings.
NInt count	[in] Array items count.
va_list args	[in] Function argument list.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.56 NStringJoinManyVAVN Function

Concatenates all the elements of a string arrays, using the specified separator between each element.

C++

```
NResult N_API NStringJoinManyVAVN(const NChar * szSeparator, HNString * phResultString,
NInt count, va_list args);
```

Parameters

Parameters	Description
const NChar * szSeparator	[in] Zero terminated string to use as a separator.
HNString * phResultString	[out] The concatenated elements of specified strings.
NInt count	[in] Array items count.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

va_list args	[in] Function argument list.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.57 NStringJoinManyVN Function

Concatenates all the elements of a string arrays, using the specified separator between each element.

C++

```
NResult N_API NStringJoinManyVN(const NChar * szSeparator, HNString * phResultString, NInt
count, ... multipleParameters);
```

Parameters

Parameters	Description
const NChar * szSeparator	[in] Zero terminated string to use as a separator.
HNString * phResultString	[out] The concatenated elements of specified strings.
NInt count	[in] Array items count.
... multipleParameters	[in] Function parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.58 NStringLastIndexOf Function

Returns the zero-based index position of the last occurrence of a specified character within string.

C++

```
NResult N_API NStringLastIndexOf(HNString hString, NChar value, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NChar value	[in] Character to seek.
NInt * pIndex	[out] The zero-based index position of *value* if that character was found, or -1 if it is not.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.59 NStringLastIndexOfAny Function

Returns the zero-based index position of the last occurrence in the string of one or more characters specified in an array.

C++

```
NResult N_API NStringLastIndexOfAny(HNString hString, const NChar * arAnyOf, NInt
anyOfLength, NInt * pIndex);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arAnyOf	[in] Characters array containing one or more characters to seek.
NInt anyOfLength	[in] Length of *arAnyOf* array.
NInt * pIndex	[out] The zero-based index position of the last occurrence in this string where any character in *arAnyOf* was found; -1 if no character in *arAnyOf* was found. **Note**: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.60 NStringLastIndexOfAnyFrom Function

Returns the zero-based index position of the last occurrence in the string of one or more characters specified in an array.

C++

```
NResult N_API NStringLastIndexOfAnyFrom(HNString hString, const NChar * arAnyOf, NInt
anyOfLength, NInt startIndex, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arAnyOf	[in] Characters array containing one or more characters to seek.
NInt anyOfLength	[in] Length of *arAnyOf* array.
NInt startIndex	[in] Seek start index in *hString*.
NInt * pIndex	[out] The zero-based index position of the last occurrence in this string where any character in *arAnyOf* was found; -1 if no character in *arAnyOf* was found. **Note**: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.61 NStringLastIndexOfAnyInRange Function

Returns the zero-based index position of the last occurrence in the string's range of one or more characters specified in an array.

C++

```
NResult N_API NStringLastIndexOfAnyInRange(HNString hString, const NChar * arAnyOf, NInt
anyOfLength, NInt startIndex, NInt count, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const NChar * arAnyOf	[in] Characters array containing one or more characters to seek.
NInt anyOfLength	[in] Length of *arAnyOf* array.
NInt startIndex	[in] Seek start index in *hString*.
NInt count	[in] *hString* range to seek (characters count starting with *startIndex*).
NInt * pIndex	[out] The zero-based index position of the last occurrence in this string where any character in *arAnyOf* was found; -1 if no character in *arAnyOf* was found. **Note**: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.62 NStringLastIndexOfFrom Function

Returns the zero-based index of the last occurrence of the specified character in the string starting from specified index.

C++

```
NResult N_API NStringLastIndexOfFrom(HNString hString, NChar value, NInt startIndex, NInt *
pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NChar value	[in] [in] Character to seek.
NInt startIndex	[in] Seek start index in *hString*.
NInt * pIndex	[out] The zero-based index position of the last occurrence in this string where any character in *arAnyOf* was found; -1 if no character in *arAnyOf* was found. **Note**: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.63 NStringLastIndexOfInRange Function

Returns the zero-based index of the last character's occurrence in string's range.

C++

```
NResult N_API NStringLastIndexOfInRange(HNString hString, NChar value, NInt startIndex,
NInt count, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NChar value	[in] Character to seek.
NInt startIndex	[in] Seek start index in *hString*.
NInt count	[in] Characters count (range to perform search).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt * pIndex	[out] The zero-based index position of the last occurrence in this string where any character in *arAnyOf* was found; -1 if no character in *arAnyOf* was found. **Note**: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.64 NStringLastIndexOfStrFromN Function

Returns the zero-based index of the last occurrence of the specified string in the NString [678] starting from specified index.

C++

```
NResult N_API NStringLastIndexOfStrFromN(HNString hString, HNString hValue, NInt
startIndex, NBool ignoreCase, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
HNString hValue	[in] Handle to string to seek.
NInt startIndex	[in] Seek start index in *hString*.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NInt * pIndex	[out] The zero-based index position of *hValue* if that string is found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.65 NStringLastIndexOfStrInRangeN Function

Returns the zero-based index of the last occurrence of the specified string in the NString [678] in specified range.

C++

```
NResult N_API NStringLastIndexOfStrInRangeN(HNString hString, HNString hValue, NInt
startIndex, NInt count, NBool ignoreCase, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
HNString hValue	[in] Handle to string to seek.
NInt startIndex	[in] Seeking range starting index.
NInt count	[in] Seeking range length.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NInt * pIndex	[out] The zero-based index position of *hValue* if that string is found, or -1 if it is not. Note: Index numbering starts from 0.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.66 NStringLastIndexOfStrN Function

Returns the zero-based index of the last occurrence of the specified NString [678] in the string.

C++

```
NResult N_API NStringLastIndexOfStrN(HNString hString, HNString hValue, NBool ignoreCase,
NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
HNString hValue	[in] NString [678] to seek.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NInt * pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.67 NStringLastIndexOfStrOrChars Function

Returns the zero-based index of the last occurrence of the specified string or character in the string.

C++

```
NResult N_API NStringLastIndexOfStrOrChars(HNString hString, const NChar * arValue, NInt
valueLength, NBool ignoreCase, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arValue	[in] NChar [1005] array which contains chars or string to check in hString.
NInt valueLength	[in] Size of *arValue* memory buffer.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NInt * pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.2.68 NStringLastIndexOfStrOrCharsFrom Function

Returns the zero-based index of the last occurrence of the specified string or character in the string.

C++

```
NResult N_API NStringLastIndexOfStrOrCharsFrom(HNString hString, const NChar * arValue,
NInt valueLength, NInt startIndex, NBool ignoreCase, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arValue	[in] NChar [1005] array which contains chars or string to check in hString.
NInt valueLength	[in] Size of *arValue* memory buffer.
NInt startIndex	[in] Seek start index.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NInt * pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.69 NStringLastIndexOfStrOrCharsInRange Function

Returns the zero-based index of the last occurrence of the specified string or character in the string.

C++

```
NResult N_API NStringLastIndexOfStrOrCharsInRange(HNString hString, const NChar * arValue,
NInt valueLength, NInt startIndex, NInt count, NBool ignoreCase, NInt * pIndex);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arValue	[in] NChar [1005] array which contains chars or string to check in *hString*.
NInt valueLength	[in] Size of *arValue* memory buffer.
NInt startIndex	[in] Seek start index.
NInt count	[in] Range to perform search.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NInt * pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.2.70 NStringPadLeft Function

Returns a new string that right-aligns the characters by padding them with spaces on the left, for a specified total length.

C++

```
NResult N_API NStringPadLeft(HNString hString, NInt totalWidth, HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NInt totalWidth	[in] The number of characters in the resulting string, equal to the number of original characters plus any additional padding characters.
HNString * phResultString	[out] A new string that is equivalent to *hString*, but right-aligned and padded on the left with as many spaces as needed to create a length of *totalWidth*. However, if *totalWidth* is less than the length of *hString*, the existing *hString* is returned. If *totalWidth* is equal to the length of *hString*, a new string that is identical to *hString* is returned.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.71 NStringPadLeftWith Function

Returns a new string that right-aligns the characters by padding them on the left with a specified character, for a specified total length.

C++

```
NResult N_API NStringPadLeftWith(HNString hString, NInt totalWidth, NChar paddingChar,
HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NInt totalWidth	[in] The number of characters in the resulting string, equal to the number of original characters plus any additional padding characters.
NChar paddingChar	[in] Padding character.
HNString * phResultString	[out] A new string that is equivalent to *hString*, but right-aligned and padded on the left with as many spaces as needed to create a length of *totalWidth*. However, if *totalWidth* is less than the length of *hString*, the existing *hString* is returned. If *totalWidth* is equal to the length of *hString*, a new string that is identical to *hString* is returned.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.2.72 NStringPadRight Function

Returns a new string that left-aligns the characters by padding them with spaces on the right, for a specified total length.

C++

```
NResult N_API NStringPadRight(HNString hString, NInt totalWidth, HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NInt totalWidth	[in] The number of characters in the resulting string, equal to the number of original characters plus any additional padding characters.
HNString * phResultString	[out] A new string that is equivalent to *hString*, but left-aligned and padded on the right with as many spaces as needed to create a length of *totalWidth*. However, if *totalWidth* is less than the length of *hString*, the function returns *hString*. If *totalWidth* is equal to the length *hString*, the function returns a new string that is identical to *hString*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.73 NStringPadRightWith Function

Returns a new string that left-aligns the characters by padding them on the right with a specified character, for a specified total length.

C++

```
NResult N_API NStringPadRightWith(HNString hString, NInt totalWidth, NChar paddingChar,
HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NInt totalWidth	[in] The number of characters in the resulting string, equal to the number of original characters plus any additional padding characters.
NChar paddingChar	[in] Padding character.
HNString * phResultString	[out] A new string that is equivalent to *hString*, but left-aligned and padded on the right with as many spaces as needed to create a length of *totalWidth*. However, if *totalWidth* is less than the length of *hString*, the function returns *hString*. If *totalWidth* is equal to the length *hString*, the function returns a new string that is identical to *hString*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.2.74 NStringPrependStrOrChars Function

Prepends a string or a charset to another string.

C++

```
NResult N_API NStringPrependStrOrChars(HNString hString, const NChar * arValue, NInt valueLength, HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678] which should be prepended.
const NChar * arValue	[in] Array with a sequence of characters. These characters will be prepended at the beginning of *hString*.
NInt valueLength	[in] Size of *arValue* memory buffer.
HNString * phResultString	[out] Result string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.75 NStringPromoteBuffer Function

Creates a new NString [678] which takes ownership of the memory buffer.

C++

```
NResult N_API NStringPromoteBuffer(NChar * szValue, NInt length, NMemoryType memoryType, HNString * phString);
```

Parameters

Parameters	Description
NChar * szValue	[out] Pointer to the characters from which the string is to be created.
NInt length	[in] Length of the new string.
NMemoryType memoryType	[in] Memory type of the block containing source characters.
HNString * phString	[out] Pointer to handle to the new NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.76 NStringRemove Function

Returns a new string in which all the characters in *hString*, beginning at a specified position and continuing through the last position, have been deleted.

C++

```
NResult N_API NStringRemove(HNString hString, NInt startIndex, NInt count, HNString * phResultString);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NInt startIndex	[in] The zero-based position to begin deleting characters.
NInt count	[in] The number of characters to delete from *hString*.
HNString * phResultString	[out] A new string that is equivalent to *hString* except for the removed characters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.77 NStringRemoveFrom Function

Returns a new string in which all the characters in *hString*, beginning at a specified position have been deleted.

C++

```
NResult N_API NStringRemoveFrom(HNString hString, NInt startIndex, HNString *
phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NInt startIndex	[in] The zero-based position to begin deleting characters.
HNString * phResultString	[out] A new string that is equivalent to *hString* except for the removed characters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.78 NStringReplace Function

Returns a new string in which all occurrences of a specified character in *hString* are replaced with another specified character.

C++

```
NResult N_API NStringReplace(HNString hString, NChar oldChar, NChar newChar, HNString *
phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NChar oldChar	[in] Character to be replaced.
NChar newChar	[in] Character to replace all occurrences of *oldChar*.
HNString * phResultString	[out] String that is equivalent to *hString* except that all instances of *oldChar* are replaced with *newChar*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NString Unit [678]

7.1.2.52.2.79 NStringReplaceStrOrChars Function

Replaces a part of a string or a char array with new values.

C++

```
NResult N_API NStringReplaceStrOrChars(HNString hString, const NChar * arOldValue, NInt oldValueLength, const NChar * arNewValue, NInt newValueLength, HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to the original NString [678].
const NChar * arOldValue	[in] Char array representing values to be replaced.
NInt oldValueLength	[in] Length of the old value array.
const NChar * arNewValue	[in] Char array representing replacement values.
NInt newValueLength	[in] Length of the new value array.
HNString * phResultString	[out] Pointer to a handle to the modified NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.80 NStringSet Function

Sets the value of the specified string.

C++

```
NResult N_API NStringSet(HNString hString, HNString * phVariable);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to the NString [678].
HNString * phVariable	[out] Pointer to a handle to the new NString [678] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.81 NStringSplitWithCount Function

Returns a string array that contains the substrings that are delimited by elements of a specified character array. A parameter specifies the maximum number of substrings to return.

C++

```
NResult N_API NStringSplitWithCount(HNString hString, const NChar * arSeparators, NInt separatorCount, NInt count, NBool removeEmptyEntries, HNString * * parhValues, NInt * pValueCount);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arSeparators	[in] An array of characters that delimit the substrings in *hString.*
NInt separatorCount	[in] Separators count in *arSeparators*.
NInt count	[in] The maximum number of substrings to return.
NBool removeEmptyEntries	[in] NTrue [1064] - empty array elements from the array returned are omitted; NFalse [1041] - empty array elements in the array returned are included.
HNString * * parhValues	[out] An array whose elements contain the substrings in *hString* that are delimited by one or more chars in separator.
NInt * pValueCount	[out] *parhValues* array size.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.82 NStringSplitWithStrsAndCount Function

Returns a string array that contains the substrings that are delimited by elements of a specified character array. A parameter specifies the maximum number of substrings to return.

C++

```
NResult N_API NStringSplitWithStrsAndCount(HNString hString, const NChar * const *
arszSeparators, NInt separatorCount, NInt count, NBool removeEmptyEntries, HNString * *
parhValues, NInt * pValueCount);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * const * arszSeparators	[in] A pointer to an array of characters that delimit the substrings in *hString.*
NInt separatorCount	[in] Separators count in *arszSeparators*.
NInt count	[in] The maximum number of substrings to return.
NBool removeEmptyEntries	[in] NTrue [1064] - empty array elements from the array returned are omitted; NFalse [1041] - empty array elements in the array returned are included.
HNString * * parhValues	[out] An array whose elements contain the substrings in *hString* that are delimited by one or more chars in separator.
NInt * pValueCount	[out] *parhValues* array size.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.83 NStringSplitWithStrsAndCountN Function

Returns a string array that contains the substrings that are delimited by elements of a specified string array. A parameter specifies the maximum number of substrings to return.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStringSplitWithStrsAndCountN(HNString hString, const HNString *
arhSeparators, NInt separatorCount, NInt count, NBool removeEmptyEntries, HNString * *
parhValues, NInt * pValueCount);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const HNString * arhSeparators	[in] A pointer to an array of string handles that delimit the substrings in *hString.*
NInt separatorCount	[in] Separators count in *arhSeparators*.
NInt count	[in] The maximum number of substrings to return.
NBool removeEmptyEntries	[in] NTrue [1064] - empty array elements from the array returned are omitted; NFalse [1041] - empty array elements in the array returned are included.
HNString * * parhValues	[out] An array whose elements contain the substrings in *hString* that are delimited by one or more chars in separator.
NInt * pValueCount	[out] *parhValues* array size.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.84 NStringStartsWithN Function

Checks whether the beginning of *hString* matches the specified string.

C++

```
NResult N_API NStringStartsWithN(HNString hString, HNString hOtherString, NBool ignoreCase,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
HNString hOtherString	[in] Handle to NString [678] to compare.
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NBool * pResult	[out] Check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.85 NStringStartsWithStrOrCharsA Function

Checks whether the beginning of *hString* matches the specified string or chars.

C++

```
NResult N_API NStringStartsWithStrOrCharsA(HNString hString, const NAChar * arValue, NInt
valueLength, NBool ignoreCase, NBool * pResult);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NAChar * arValue	[in] Pointer to a char array to compare.
NInt valueLength	[in] Length of *arValue.*
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NBool * pResult	[out] Check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

This is an ANSCII version of the function.

Module

NString Unit [678]

7.1.2.52.2.86 NStringStartsWithStrOrCharsW Function

Checks whether the beginning of *hString* matches the specified string or chars.

C++

```
NResult N_API NStringStartsWithStrOrCharsW(HNString hString, const NWChar * arValue, NInt
valueLength, NBool ignoreCase, NBool * pResult);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NWChar * arValue	[in] Pointer to a char array to compare.
NInt valueLength	[in] Length of *arValue.*
NBool ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
NBool * pResult	[out] Check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.87 NStringSubstring Function

Gets a substring from *hString*. The substring starts at a specified character position.

C++

```
NResult N_API NStringSubstring(HNString hString, NInt startIndex, NInt length, HNString *
phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NInt startIndex	[in] The zero-based starting character position of a substring in *hString*.
NInt length	[in] The number of characters in the substring.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNString * phResultString	[out] A string that is equivalent to the substring of length *length* that begins at *startIndex* in *hString*, or Empty if *startIndex* is equal to the length of *hString* and *length* is zero.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.88 NStringSubstringFrom Function

Gets a substring from *hString*. The substring starts at a specified character position.

C++

```
NResult N_API NStringSubstringFrom(HNString hString, NInt startIndex, HNString *
phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NInt startIndex	[in] The zero-based starting character position of a substring in *hString*.
HNString * phResultString	[out] A pointer to handle to a string, equal to substring from *startIndex* to the end of the NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.89 NStringToLower Function

Converts specified string to lowercase.

C++

```
NResult N_API NStringToLower(HNString hString, HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
HNString * phResultString	[out] A string in lowercase.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.90 NStringToString Function

Converts the specified string to another string with specified format.

C++

```
NResult N_API NStringToString(HNString hString, const NChar * szFormat, HNString * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hString	[in] Handle to the original NString [678].
const NChar * szFormat	[in] Pointer to the new format.
HNString * phValue	[out] Pointer to a handle to the converted string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.91 NStringToStringOld Function

Converts the specified string to charset.

C++

```
NResult N_API NStringToStringOld(HNString hString, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to the original NString [678].
NChar * szValue	[out] Pointer to a handle to the converted NChar [1005].
NInt valueSize	[in] Size of the converted NChar [1005].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.92 NStringToStrOrCharArray Function

Converts a string to another string or a character array.

C++

```
NResult N_API NStringToStrOrCharArray(HNString hString, NBool nullTerminate, NChar * *
parValues, NInt * pLength);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NBool nullTerminate	[in] NTrue [1064] if null values are to be terminated; else NFalse [1041].
NChar * * parValues	[out] Pointer to a pointer to the resulting char array.
NInt * pLength	[out] Pointer to the length of the resulting char array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.2.93 NStringToStrOrCharArrayRange Function

Converts a string to another string or a part of character array.

C++

```
NResult N_API NStringToStrOrCharArrayRange(HNString hString, NBool nullTerminate, NInt startIndex, NInt length, NChar * * parValues, NInt * pLength);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
NBool nullTerminate	[in] NTrue [1064] if null values are to be terminated; else NFalse [1041].
NInt startIndex	[in] Starting index in the array, at which the converted characters are being inserted.
NInt length	[in] Length of the range into which the converted characters are being inserted.
NChar * * parValues	[out] Pointer to a pointer to the resulting char array.
NInt * pLength	[out] Pointer to the length of the resulting char array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.94 NStringToUpper Function

Converts specified string to uppercase.

C++

```
NResult N_API NStringToUpper(HNString hString, HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
HNString * phResultString	[out] A string in uppercase.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.95 NStringTrim Function

Removes all leading and trailing white-space characters from specified string.

C++

```
NResult N_API NStringTrim(HNString hString, HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNString * phResultString	[out] *hString* with white-space characters removed from the start and end.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.96 NStringTrimAny Function

Removes all leading and trailing occurrences of a sets of characters from specified string.

C++

```
NResult N_API NStringTrimAny(HNString hString, const NChar * arChars, NInt charCount,
HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arChars	[in] An array of characters to remove.
NInt charCount	[in] Characters in *arChars* array count.
HNString * phResultString	[out] String with all trailing characters removed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.97 NStringTrimEnd Function

Removes all trailing characters from *hString*.

C++

```
NResult N_API NStringTrimEnd(HNString hString, HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
HNString * phResultString	[out] String with all trailing characters removed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.98 NStringTrimEndAny Function

Removes all trailing occurrences of a set of characters specified in an array from *hString*.

C++

```
NResult N_API NStringTrimEndAny(HNString hString, const NChar * arChars, NInt charCount,
HNString * phResultString);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arChars	[in] An array of characters to remove.
NInt charCount	[in] Characters in *arChars* array count.
HNString * phResultString	[out] String with all trailing characters removed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.99 NStringTrimStart Function

Removes all trailing characters from *hString*.

C++

```
NResult N_API NStringTrimStart(HNString hString, HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
HNString * phResultString	[out] String with all trailing characters removed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.2.100 NStringTrimStartAny Function

Removes all leading occurrences of a set of characters specified in an array from *hString*.

C++

```
NResult N_API NStringTrimStartAny(HNString hString, const NChar * arChars, NInt charCount,
HNString * phResultString);
```

Parameters

Parameters	Description
HNString hString	[in] Handle to NString [678].
const NChar * arChars	[in] An array of characters to remove.
NInt charCount	[in] Characters in *arChars* array count.
HNString * phResultString	[out] String with all trailing characters removed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3 Macros

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.3.1 N_NEW_LINE Macro

Picks a Unicode (W) or ASCII (A) version of the N_NEW_LINE macro.

C++

```
#define N_NEW_LINE N_MACRO_AW(N_NEW_LINE)
```

Module

NString Unit [678]

7.1.2.52.3.2 N_STRING_SIZE Macro

Defines the string size.

C++

```
#define N_STRING_SIZE 24
```

Module

NString Unit [678]

7.1.2.52.3.3 NStringAppend Macro

Extends specified string content by adding an additional string at its end.

C++

```
#define NStringAppend(hString, szValue, phResultString) NStringAppendStrOrChars(hString,
szValue, -1, phResultString)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678] which should be appended.
szValue	[in] Pointer to a string that will be appended at the end of *hString*.
phResultString	[out] Result string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.4 NStringAppendChars Macro

Extends specified string content by adding additional chars at its end.

C++

```
#define NStringAppendChars(hString, arValue, valueLength, phResultString)
NStringAppendStrOrChars(hString, arValue, valueLength, phResultString)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678] which should be appended.
arValue	[in] Array with a sequence of characters. These characters will be appended at the end of *hString*.
valueLength	[in] Size of *arValue* memory buffer.
phResultString	[out] Result string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.5 NStringAppendStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringAppendStrOrChars function.

C++

```
#define NStringAppendStrOrChars N_FUNC_AW(NStringAppendStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.6 NStringCompare Macro

Compares two strings and retrieves their relative position in the sort order.

C++

```
#define NStringCompare(hString, szValue, ignoreCase, pResult)
NStringCompareStrOrChars(hString, szValue, -1, ignoreCase, pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to the first string to use in comparison.
szValue	[in] Pointer to the second string to use in comparison.
ignoreCase	[in] NTrue [1064] to ignore letter case during comparison; else NFalse [1041].
pResult	[out] Pointer to NInt [1006] value that indicates the lexical relationship between the two comparands.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.7 NStringCompareChars Macro

Compares a string with a char array and retrieves their relative position in the sort order.

C++

```
#define NStringCompareChars(hString, arValue, valueLength, ignoreCase, pResult)
NStringCompareStrOrChars(hString, arValue, valueLength, ignoreCase, pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to the string to use in comparison.
arValue	[in] Pointer to the char array to use in comparison.
valueLength	[in] Length of the char array.
ignoreCase	[in] NTrue [1064] to ignore letter case during comparison; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

pResult	[out] Pointer to NInt [1006] value that indicates the lexical relationship between the two comparands.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.8 NStringCompareN Macro

Compares two strings and retrieves their relative position in the sort order.

C++

```
#define NStringCompareN(hString, hOtherString, ignoreCase, pResult)
NStringCompareRangeN(hString, 0, hOtherString, 0, N_INT_MAX, ignoreCase, pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to the first string to use in comparison.
hOtherString	[in] Handle to the second string to use in comparison.
ignoreCase	[in] NTrue [1064] to ignore letter case during comparison; else NFalse [1041].
pResult	[out] Pointer to NInt [1006] value that indicates the lexical relationship between the two comparands.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.9 NStringCompareRange Macro

Compares substrings of the two specified strings and retrieves their relative position in the sort order.

C++

```
#define NStringCompareRange(hString, index, szValue, length, ignoreCase, pResult)
NStringCompareRangeStrOrChars(hString, index, szValue, -1, length, ignoreCase, pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to the first string to use in comparison.
index	[in] Position of the substring in *hString.*
szValue	[in] Pointer to the second string to use in comparison.
length	[in] The maximum number of characters in the substrings to compare.
ignoreCase	[in] NTrue [1064] to ignore letter case during comparison; else NFalse [1041].
pResult	[out] Pointer to NInt [1006] value that indicates the lexical relationship between the two comparands.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NString Unit [678]

7.1.2.52.3.10 NStringCompareRangeChars Macro

Compares substrings of a string and a charset and retrieves their relative position in the sort order.

C++

```
#define NStringCompareRangeChars(hString, index, arValue, valueLength, length, ignoreCase,
pResult) NStringCompareRangeStrOrChars(hString, index, arValue, valueLength, length,
ignoreCase, pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to the string to use in comparison.
index	[in] Position of the substring in *hString.*
arValue	[in] Handle to the charset to use in comparison.
valueLength	[in] Length of the charset.
length	[in] The maximum number of characters in the substrings to compare.
ignoreCase	[in] NTrue [1064] to ignore letter case during comparison; else NFalse [1041].
pResult	[out] Pointer to NInt [1006] value that indicates the lexical relationship between the two comparands.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.11 NStringCompareRangeN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringCompareRangeN function.

C++

```
#define NStringCompareRangeN N_FUNC_AW(NStringCompareRangeN)
```

Module

NString Unit [678]

7.1.2.52.3.12 NStringCompareRangeStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringCompareRangeStrOrChars function.

C++

```
#define NStringCompareRangeStrOrChars N_FUNC_AW(NStringCompareRangeStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.13 NStringCompareStrOrChars Macro

Compares a string with a string or char array and retrieves their relative position in the sort order.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NStringCompareStrOrChars(hString, arValue, valueLength, ignoreCase, pResult)
NStringCompareRangeStrOrChars(hString, 0, arValue, valueLength, N_INT_MAX, ignoreCase,
pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to the string to use in comparison.
arValue	[in] Pointer to the char array to use in comparison.
valueLength	[in] Length of the char array.
ignoreCase	[in] NTrue [1064] to ignore letter case during comparison; else NFalse [1041].
pResult	[out] Pointer to NInt [1006] value that indicates the lexical relationship between the two comparands.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.14 NStringConcat Macro

Concatenates strings specified by handles.

C++

```
#define NStringConcat(szValue1, szValue2, phResultString) NStringConcatStrOrChars(szValue1,
-1, szValue2, -1, phResultString)
```

Parameters

Parameters	Description
hValue1	[in] Handle to first string to concatenate.
hValue2	[in] Handle to second string to concatenate.
phResultString	[out] The concatenated strings.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.15 NStringConcatArray Macro

Picks a Unicode (W) or ASCII (A) version of the NStringConcatArray function.

C++

```
#define NStringConcatArray N_FUNC_AW(NStringConcatArray)
```

Module

NString Unit [678]

7.1.2.52.3.16 NStringConcatChars Macro

Concatenates charsets specified by handles.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NStringConcatChars(arValue1, value1Length, arValue2, value2Length, phResultString)
NStringConcatStrOrChars(arValue1, value1Length, arValue2, value2Length, phResultString)
```

Parameters

Parameters	Description
arValue1	[in] Pointer to the first charset.
value1Length	[in] Length of the first charset.
arValue2	[in] Pointer to the second charset.
value2Length	[in] Length of the second charset.
phResultString	[out] Pointer to a handle to the resulting string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.17 NStringConcatMany Macro

Picks a Unicode (W) or ASCII (A) version of the NStringConcatMany function.

C++

```
#define NStringConcatMany N_FUNC_AW(NStringConcatMany)
```

Module

NString Unit [678]

7.1.2.52.3.18 NStringConcatManyVA Macro

Picks a Unicode (W) or ASCII (A) version of the NStringConcatManyVA function.

C++

```
#define NStringConcatManyVA N_FUNC_AW(NStringConcatManyVA)
```

Module

NString Unit [678]

7.1.2.52.3.19 NStringConcatStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringConcatStrOrChars function.

C++

```
#define NStringConcatStrOrChars N_FUNC_AW(NStringConcatStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.20 NStringContainsChars Macro

Returns a value indicating whether specified chars occur within string specified by handle.

C++

```
#define NStringContainsChars(hString, arValue, valueLength, pResult)
NStringContainsStrOrChars(hString, arValue, valueLength, pResult)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
hString	[in] Handle to NString [678] to check.
arValue	[in] NChar [1005] array which contains chars to check in hString.
valueLength	[in] Length of memory buffer where *arValue* is stored.
pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.21 NStringContainsStr Macro

Returns a value indicating whether specified string occurs within string specified by handle.

C++

```
#define NStringContainsStr(hString, szValue, pResult) NStringContainsStrOrChars(hString,
szValue, -1, pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678] to check.
szValue	[in] Pointer to NString [678] to check in hString.
valueLength	[in] Length of NString [678] pointed to by *szValue*.
pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.22 NStringContainsStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringContainsStrOrChars function.

C++

```
#define NStringContainsStrOrChars N_FUNC_AW(NStringContainsStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.23 NStringCopyTo Macro

Copies a specified number of characters from a specified position in handle to a specified array.

C++

```
#define NStringCopyTo(hString, sourceIndex, arValue, valueLength, count)
NStringCopyToStrOrChars(hString, sourceIndex, arValue, valueLength, NFalse, count)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
hString	[in] Handle to NString [678] which contains characters to copy.
sourceIndex	[in] The index of the first character in this handle to copy.
arValue	[out] Pointer to memory buffer which contains copied characters.
valueLength	[in] Size of *arValue* memory buffer.
count	[in] The number of characters in *hString* to copy to *arValue*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.24 NStringCopyToString Macro

Copies a specified number of characters from a specified position in handle to a specified string.

C++

```
#define NStringCopyToString(hString, sourceIndex, szValue, valueSize, count)
NStringCopyToStrOrChars(hString, sourceIndex, szValue, valueSize, NTrue, count)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678] which contains characters to copy.
sourceIndex	[in] The index of the first character in this handle to copy.
szValue	[out] Pointer to NString [678] which contains copied characters.
valueSize	[in] Size of sz*Value* string.
count	[in] The number of characters in *hString* to copy to sz*Value*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.25 NStringCopyToStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringCopyToStrOrChars function.

C++

```
#define NStringCopyToStrOrChars N_FUNC_AW(NStringCopyToStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.26 NStringCreate Macro

Creates a string with the specified value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NStringCreate(szValue, phString) NStringCreateFromStrOrChars(szValue, -1, phString)
```

Parameters

Parameters	Description
szValue	[in] Pointer to the string value.
phString	[out] Pointer to a handle to the new string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.27 NStringCreateFromChar Macro

Picks a Unicode (W) or ASCII (A) version of the NStringCreateFromChar function.

C++

```
#define NStringCreateFromChar N_FUNC_AW(NStringCreateFromChar)
```

Module

NString Unit [678]

7.1.2.52.3.28 NStringCreateFromChars Macro

Creates a string from the specified array of characters.

C++

```
#define NStringCreateFromChars(arValue, length, phString)
NStringCreateFromStrOrChars(arValue, length, phString)
```

Parameters

Parameters	Description
arValue	[in] Array of chars.
length	[in] Size of *arValue* memory buffer.
phString	[out] Pointer to a handle to the new string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.29 NStringCreateFromStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringCreateFromStrOrChars function.

C++

```
#define NStringCreateFromStrOrChars N_FUNC_AW(NStringCreateFromStrOrChars)
```

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.3.30 NStringCreateWrapper Macro

Picks a Unicode (W) or ASCII (A) version of the NStringCreateWrapper function.

C++

```
#define NStringCreateWrapper N_FUNC_AW(NStringCreateWrapper)
```

Module

NString Unit [678]

7.1.2.52.3.31 NStringEndsWith Macro

Checks whether the end of the string matches the specified string.

C++

```
#define NStringEndsWith(hString, szValue, ignoreCase, pResult)
NStringEndsWithStrOrChars(hString, szValue, -1, ignoreCase, pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to string to check.
szValue	[in] Pointer to the string to compare to the *hString*.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.32 NStringEndsWithChars Macro

Checks whether the end of the string matches the specified string or charset.

C++

```
#define NStringEndsWithChars(hString, arValue, valueLength, ignoreCase, pResult)
NStringEndsWithStrOrChars(hString, arValue, valueLength, ignoreCase, pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to string to check.
arValue	[in] Pointer to the char array to compare to the *hString*.
valueLength	[in] Length of *arValue.*
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.3.33 NStringEndsWithStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringEndsWithStrOrChars function.

C++

```
#define NStringEndsWithStrOrChars N_FUNC_AW(NStringEndsWithStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.34 NStringEquals Macro

Checks whether two specified stings have the same value.

C++

```
#define NStringEquals(hString, szValue, ignoreCase, pResult)
NStringEqualsStrOrChars(hString, szValue, -1, ignoreCase, pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to the first string to compare.
szValue	[in] Pointer to the second string to compare.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.35 NStringEqualsChars Macro

Checks whether the specified string and the specified charset have the same value.

C++

```
#define NStringEqualsChars(hString, arValue, valueLength, ignoreCase, pResult)
NStringEqualsStrOrChars(hString, arValue, valueLength, ignoreCase, pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to the string to compare.
arValue	[in] Pointer to the char array to compare.
valueLength	[in] Length of *arValue.*
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pResult	[out] Strings check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.3.36 NStringEqualsStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringEqualsStrOrChars function.

C++

```
#define NStringEqualsStrOrChars N_FUNC_AW(NStringEqualsStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.37 NStringFormat Macro

Picks a Unicode (W) or ASCII (A) version of the NStringFormat function.

C++

```
#define NStringFormat N_FUNC_AW(NStringFormat)
```

Module

NString Unit [678]

7.1.2.52.3.38 NStringFormatVA Macro

Picks a Unicode (W) or ASCII (A) version of the NStringFormatVA function.

C++

```
#define NStringFormatVA N_FUNC_AW(NStringFormatVA)
```

Module

NString Unit [678]

7.1.2.52.3.39 NStringGetBuffer Macro

Picks a Unicode (W) or ASCII (A) version of the NStringGetBuffer function.

C++

```
#define NStringGetBuffer N_FUNC_AW(NStringGetBuffer)
```

Module

NString Unit [678]

7.1.2.52.3.40 NStringGetChar Macro

Picks a Unicode (W) or ASCII (A) version of the NStringGetChar function.

C++

```
#define NStringGetChar N_FUNC_AW(NStringGetChar)
```

Module

NString Unit [678]

7.1.2.52.3.41 NStringGetLength Macro

Picks a Unicode (W) or ASCII (A) version of the NStringGetLength function.

C++

```
#define NStringGetLength N_FUNC_AW(NStringGetLength)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NString Unit [678]

7.1.2.52.3.42 NStringIndexOf Macro

Picks a Unicode (W) or ASCII (A) version of the NStringIndexOf function.

C++

```
#define NStringIndexOf N_FUNC_AW(NStringIndexOf)
```

Module

NString Unit [678]

7.1.2.52.3.43 NStringIndexOfAny Macro

Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfAny function.

C++

```
#define NStringIndexOfAny N_FUNC_AW(NStringIndexOfAny)
```

Module

NString Unit [678]

7.1.2.52.3.44 NStringIndexOfAnyFrom Macro

Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfAnyFrom function.

C++

```
#define NStringIndexOfAnyFrom N_FUNC_AW(NStringIndexOfAnyFrom)
```

Module

NString Unit [678]

7.1.2.52.3.45 NStringIndexOfAnyInRange Macro

Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfAnyInRange function.

C++

```
#define NStringIndexOfAnyInRange N_FUNC_AW(NStringIndexOfAnyInRange)
```

Module

NString Unit [678]

7.1.2.52.3.46 NStringIndexOfChars Macro

Returns the zero-based index of the first occurrence of the specified character in the string.

C++

```
#define NStringIndexOfChars(hString, arValue, valueLength, ignoreCase, pIndex)
NStringIndexOfStrOrChars(hString, arValue, valueLength, ignoreCase, pIndex)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
arValue	[in] NChar [1005] array which contains chars to check in *hString*.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

valueLength	[in] Size of *arValue* memory buffer.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.47 NStringIndexOfCharsFrom Macro

Returns the zero-based index of the first occurrence of the specified character in the string.

C++

```
#define NStringIndexOfCharsFrom(hString, arValue, valueLength, startIndex, ignoreCase,
pIndex) NStringIndexOfStrOrCharsFrom(hString, arValue, valueLength, startIndex, ignoreCase,
pIndex)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
arValue	[in] NChar [1005] array which contains chars to check in *hString*.
valueLength	[in] Size of *arValue* memory buffer.
startIndex	[in] Seek start index.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.48 NStringIndexOfCharsInRange Macro

Returns the zero-based index of the first occurrence of the specified character in the string.

C++

```
#define NStringIndexOfCharsInRange(hString, arValue, valueLength, startIndex, count,
ignoreCase, pIndex) NStringIndexOfStrOrCharsInRange(hString, arValue, valueLength,
startIndex, count, ignoreCase, pIndex)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
arValue	[in] NChar [1005] array which contains chars to check in *hString*.
valueLength	[in] Size of *arValue* memory buffer.
startIndex	[in] Seek start index.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

count	[in] Range to perform search.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.49 NStringIndexOfFrom Macro

Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfFrom function.

C++

```
#define NStringIndexOfFrom N_FUNC_AW(NStringIndexOfFrom)
```

Module

NString Unit [678]

7.1.2.52.3.50 NStringIndexOfInRange Macro

Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfInRange function.

C++

```
#define NStringIndexOfInRange N_FUNC_AW(NStringIndexOfInRange)
```

Module

NString Unit [678]

7.1.2.52.3.51 NStringIndexOfStr Macro

Returns the zero-based index of the first occurrence of the specified string in the string.

C++

```
#define NStringIndexOfStr(hString, szValue, ignoreCase, pIndex)
NStringIndexOfStrOrChars(hString, szValue, -1, ignoreCase, pIndex)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
szValue	[in] Pointer to a string to check in *hString*.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.3.52 NStringIndexOfStrFrom Macro

Returns the zero-based index of the first occurrence of the specified string in the string.

C++

```
#define NStringIndexOfStrFrom(hString, szValue, startIndex, ignoreCase, pIndex)
NStringIndexOfStrOrCharsFrom(hString, szValue, -1, startIndex, ignoreCase, pIndex)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
szValue	[in] Pointer to a string to check in *hString*.
startIndex	[in] Seek start index.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.53 NStringIndexOfStrFromN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrFromN function.

C++

```
#define NStringIndexOfStrFromN N_FUNC_AW(NStringIndexOfStrFromN)
```

Module

NString Unit [678]

7.1.2.52.3.54 NStringIndexOfStrInRange Macro

Returns the zero-based index of the first occurrence of the specified string in the string.

C++

```
#define NStringIndexOfStrInRange(hString, szValue, startIndex, count, ignoreCase, pIndex)
NStringIndexOfStrOrCharsInRange(hString, szValue, -1, startIndex, count, ignoreCase, pIndex)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
arValue	[in] Pointer to a string to check in *hString*.
startIndex	[in] Seek start index.
count	[in] Range to perform search.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NString Unit [678]

7.1.2.52.3.55 NStringIndexOfStrInRangeN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrInRangeN function.

C++

```
#define NStringIndexOfStrInRangeN N_FUNC_AW(NStringIndexOfStrInRangeN)
```

Module

NString Unit [678]

7.1.2.52.3.56 NStringIndexOfStrN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrN function.

C++

```
#define NStringIndexOfStrN N_FUNC_AW(NStringIndexOfStrN)
```

Module

NString Unit [678]

7.1.2.52.3.57 NStringIndexOfStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrOrChars function.

C++

```
#define NStringIndexOfStrOrChars N_FUNC_AW(NStringIndexOfStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.58 NStringIndexOfStrOrCharsFrom Macro

Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrOrCharsFrom function.

C++

```
#define NStringIndexOfStrOrCharsFrom N_FUNC_AW(NStringIndexOfStrOrCharsFrom)
```

Module

NString Unit [678]

7.1.2.52.3.59 NStringIndexOfStrOrCharsInRange Macro

Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrOrCharsInRange function.

C++

```
#define NStringIndexOfStrOrCharsInRange N_FUNC_AW(NStringIndexOfStrOrCharsInRange)
```

Module

NString Unit [678]

7.1.2.52.3.60 NStringInsert Macro

Returns a new string in which a specified string is inserted at a specified index position.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NStringInsert(hString, index, szValue, phResultString)
NStringInsertStrOrChars(hString, index, szValue, -1, phResultString)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
index	[in] The zero-based index position of the insertion.
szValue	[in] Pointer to the string to insert.
phResultString	[out] Pointer to new string that is equivalent to *hString*, but with *szValue* inserted at position *index*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.61 NStringInsertChars Macro

Returns a new string in which a specified characters are inserted at a specified index position.

C++

```
#define NStringInsertChars(hString, index, arValue, valueLength, phResultString)
NStringInsertStrOrChars(hString, index, arValue, valueLength, phResultString)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
index	[in] The zero-based index position of the insertion.
arValue	[in] NChar [1005] array containing chars to insert.
valueLength	[in] Size of *arValue* memory buffer.
phResultString	[out] Pointer to new string that is equivalent to *hString*, but with *arValue* inserted at position *index*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.62 NStringInsertN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringInsertN function.

C++

```
#define NStringInsertN N_FUNC_AW(NStringInsertN)
```

Module

NString Unit [678]

7.1.2.52.3.63 NStringInsertStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringInsertStrOrChars function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NStringInsertStrOrChars N_FUNC_AW(NStringInsertStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.64 NStringJoinArray Macro

Picks a Unicode (W) or ASCII (A) version of the NStringJoinArray function.

C++

```
#define NStringJoinArray N_FUNC_AW(NStringJoinArray)
```

Module

NString Unit [678]

7.1.2.52.3.65 NStringJoinArrayVN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringJoinArrayVN function.

C++

```
#define NStringJoinArrayVN N_FUNC_AW(NStringJoinArrayVN)
```

Module

NString Unit [678]

7.1.2.52.3.66 NStringJoinMany Macro

Picks a Unicode (W) or ASCII (A) version of the NStringJoinMany function.

C++

```
#define NStringJoinMany N_FUNC_AW(NStringJoinMany)
```

Module

NString Unit [678]

7.1.2.52.3.67 NStringJoinManyVAVN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringJoinManyVAVN function.

C++

```
#define NStringJoinManyVAVN N_FUNC_AW(NStringJoinManyVAVN)
```

Module

NString Unit [678]

7.1.2.52.3.68 NStringJoinManyVN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringJoinManyVN function.

C++

```
#define NStringJoinManyVN N_FUNC_AW(NStringJoinManyVN)
```

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.3.69 NStringLastIndexOf Macro

Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOf function.

C++

```
#define NStringLastIndexOf N_FUNC_AW(NStringLastIndexOf)
```

Module

NString Unit [678]

7.1.2.52.3.70 NStringLastIndexOfAny Macro

Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfAny function.

C++

```
#define NStringLastIndexOfAny N_FUNC_AW(NStringLastIndexOfAny)
```

Module

NString Unit [678]

7.1.2.52.3.71 NStringLastIndexOfAnyFrom Macro

Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfAnyFrom function.

C++

```
#define NStringLastIndexOfAnyFrom N_FUNC_AW(NStringLastIndexOfAnyFrom)
```

Module

NString Unit [678]

7.1.2.52.3.72 NStringLastIndexOfAnyInRange Macro

Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfAnyInRange function.

C++

```
#define NStringLastIndexOfAnyInRange N_FUNC_AW(NStringLastIndexOfAnyInRange)
```

Module

NString Unit [678]

7.1.2.52.3.73 NStringLastIndexOfChars Macro

Returns the zero-based index of the last occurrence of the specified character in the string.

C++

```
#define NStringLastIndexOfChars(hString, arValue, valueLength, ignoreCase, pIndex)
NStringLastIndexOfStrOrChars(hString, arValue, valueLength, ignoreCase, pIndex)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
arValue	[in] NChar [1005] array which contains chars to check in *hString*.
valueLength	[in] Size of *arValue* memory buffer.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.74 NStringLastIndexOfCharsFrom Macro

Returns the zero-based index of the last occurrence of the specified character in the string.

C++

```
#define NStringLastIndexOfCharsFrom(hString, arValue, valueLength, startIndex, ignoreCase,
pIndex) NStringLastIndexOfStrOrCharsFrom(hString, arValue, valueLength, startIndex,
ignoreCase, pIndex)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
arValue	[in] NChar [1005] array which contains chars to check in *hString*.
valueLength	[in] Size of *arValue* memory buffer.
startIndex	[in] Seek start index.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.75 NStringLastIndexOfCharsInRange Macro

Returns the zero-based index of the last occurrence of the specified character in the string.

C++

```
#define NStringLastIndexOfCharsInRange(hString, arValue, valueLength, startIndex, count,
ignoreCase, pIndex) NStringLastIndexOfStrOrCharsInRange(hString, arValue, valueLength,
startIndex, count, ignoreCase, pIndex)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
arValue	[in] NChar [1005] array which contains chars to check in *hString*.
valueLength	[in] Size of *arValue* memory buffer.
startIndex	[in] Seek start index.
count	[in] Range to perform search.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.76 NStringLastIndexOfFrom Macro

Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfFrom function.

C++

```
#define NStringLastIndexOfFrom N_FUNC_AW(NStringLastIndexOfFrom)
```

Module

NString Unit [678]

7.1.2.52.3.77 NStringLastIndexOfInRange Macro

Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfInRange function.

C++

```
#define NStringLastIndexOfInRange N_FUNC_AW(NStringLastIndexOfInRange)
```

Module

NString Unit [678]

7.1.2.52.3.78 NStringLastIndexOfStr Macro

Returns the zero-based index of the last occurrence of the specified string in the string.

C++

```
#define NStringLastIndexOfStr(hString, szValue, ignoreCase, pIndex)
NStringLastIndexOfStrOrChars(hString, szValue, -1, ignoreCase, pIndex)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
szValue	[in] Pointer to a string to check in *hString*.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.79 NStringLastIndexOfStrFrom Macro

Returns the zero-based index of the last occurrence of the specified string in the string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NStringLastIndexOfStrFrom(hString, szValue, startIndex, ignoreCase, pIndex)
NStringLastIndexOfStrOrCharsFrom(hString, szValue, -1, startIndex, ignoreCase, pIndex)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
szValue	[in] Pointer to a string to check in *hString*.
startIndex	[in] Seek start index.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.80 NStringLastIndexOfStrFromN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrFromN function.

C++

```
#define NStringLastIndexOfStrFromN N_FUNC_AW(NStringLastIndexOfStrFromN)
```

Module

NString Unit [678]

7.1.2.52.3.81 NStringLastIndexOfStrInRange Macro

Returns the zero-based index of the last occurrence of the specified string in the string.

C++

```
#define NStringLastIndexOfStrInRange(hString, szValue, startIndex, count, ignoreCase,
pIndex) NStringLastIndexOfStrOrCharsInRange(hString, szValue, -1, startIndex, count,
ignoreCase, pIndex)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
arValue	[in] Pointer to a string to check in *hString*.
startIndex	[in] Seek start index.
count	[in] Range to perform search.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pIndex	[out] The zero-based index position of NString [678] if found, or -1 if it is not. Note: Index numbering starts from 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.3.82 NStringLastIndexOfStrInRangeN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrInRangeN function.

C++

```
#define NStringLastIndexOfStrInRangeN N_FUNC_AW(NStringLastIndexOfStrInRangeN)
```

Module

NString Unit [678]

7.1.2.52.3.83 NStringLastIndexOfStrN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrN function.

C++

```
#define NStringLastIndexOfStrN N_FUNC_AW(NStringLastIndexOfStrN)
```

Module

NString Unit [678]

7.1.2.52.3.84 NStringLastIndexOfStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrOrChars function.

C++

```
#define NStringLastIndexOfStrOrChars N_FUNC_AW(NStringLastIndexOfStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.85 NStringLastIndexOfStrOrCharsFrom Macro

Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrOrCharsFrom function.

C++

```
#define NStringLastIndexOfStrOrCharsFrom N_FUNC_AW(NStringLastIndexOfStrOrCharsFrom)
```

Module

NString Unit [678]

7.1.2.52.3.86 NStringLastIndexOfStrOrCharsInRange Macro

Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrOrCharsInRange function.

C++

```
#define NStringLastIndexOfStrOrCharsInRange N_FUNC_AW(NStringLastIndexOfStrOrCharsInRange)
```

Module

NString Unit [678]

7.1.2.52.3.87 NStringPadLeft Macro

Picks a Unicode (W) or ASCII (A) version of the NStringPadLeft function.

C++

```
#define NStringPadLeft N_FUNC_AW(NStringPadLeft)
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NString Unit [678]

7.1.2.52.3.88 NStringPadLeftWith Macro

Picks a Unicode (W) or ASCII (A) version of the NStringPadLeftWith function.

C++

```
#define NStringPadLeftWith N_FUNC_AW(NStringPadLeftWith)
```

Module

NString Unit [678]

7.1.2.52.3.89 NStringPadRight Macro

Picks a Unicode (W) or ASCII (A) version of the NStringPadRight function.

C++

```
#define NStringPadRight N_FUNC_AW(NStringPadRight)
```

Module

NString Unit [678]

7.1.2.52.3.90 NStringPadRightWith Macro

Picks a Unicode (W) or ASCII (A) version of the NStringPadRightWith function.

C++

```
#define NStringPadRightWith N_FUNC_AW(NStringPadRightWith)
```

Module

NString Unit [678]

7.1.2.52.3.91 NStringPrepend Macro

Prepends a string to another string.

C++

```
#define NStringPrepend(hString, szValue, phResultString) NStringPrependStrOrChars(hString,
szValue, -1, phResultString)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678] which should be prepended.
szValue	[in] Pointer to a string that will be prepended at the beginning of *hString*.
phResultString	[out] Result string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.3.92 NStringPrependChars Macro

Prepends a charset to another string.

C++

```
#define NStringPrependChars(hString, arValue, valueLength, phResultString)
NStringPrependStrOrChars(hString, arValue, valueLength, phResultString)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678] which should be prepended.
arValue	[in] Array with a sequence of characters. These characters will be prepended at the beginning of *hString*.
valueLength	[in] Size of *arValue* memory buffer.
phResultString	[out] Result string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.93 NStringPrependStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringPrependStrOrChars function.

C++

```
#define NStringPrependStrOrChars N_FUNC_AW(NStringPrependStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.94 NStringPromoteBuffer Macro

Picks a Unicode (W) or ASCII (A) version of the NStringPromoteBuffer function.

C++

```
#define NStringPromoteBuffer N_FUNC_AW(NStringPromoteBuffer)
```

Module

NString Unit [678]

7.1.2.52.3.95 NStringRemove Macro

Picks a Unicode (W) or ASCII (A) version of the NStringRemove function.

C++

```
#define NStringRemove N_FUNC_AW(NStringRemove)
```

Module

NString Unit [678]

7.1.2.52.3.96 NStringRemoveFrom Macro

Picks a Unicode (W) or ASCII (A) version of the NStringRemoveFrom function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NStringRemoveFrom N_FUNC_AW(NStringRemoveFrom)
```

Module

NString Unit [678]

7.1.2.52.3.97 NStringReplace Macro

Picks a Unicode (W) or ASCII (A) version of the NStringReplace function.

C++

```
#define NStringReplace N_FUNC_AW(NStringReplace)
```

Module

NString Unit [678]

7.1.2.52.3.98 NStringReplaceChars Macro

Replaces a part of a char array with new values.

C++

```
#define NStringReplaceChars(hString, arOldValue, oldValueLength, arNewValue,
newValueLength, phResultString) NStringReplaceStrOrChars(hString, arOldValue,
oldValueLength, arNewValue, newValueLength, phResultString)
```

Parameters

Parameters	Description
hString	[in] Handle to the original NString [678].
arOldValue	[in] Char array representing values to be replaced.
oldValueLength	[in] Length of the old value array.
arNewValue	[in] Char array representing replacement values.
newValueLength	[in] Length of the new value array.
phResultString	[out] Pointer to a handle to the modified NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.99 NStringReplaceStr Macro

Replaces a part of a string with new values.

C++

```
#define NStringReplaceStr(hString, szOldValue, szNewValue, phResultString)
NStringReplaceStrOrChars(hString, szOldValue, -1, szNewValue, -1, phResultString)
```

Parameters

Parameters	Description
hString	[in] Handle to the original NString [678].
arOldValue	[in] Char array representing values to be replaced.
oldValueLength	[in] Length of the old value array.
arNewValue	[in] Char array representing replacement values.
newValueLength	[in] Length of the new value array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

phResultString	[out] Pointer to a handle to the modified NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.100 NStringReplaceStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringReplaceStrOrChars function.

C++

```
#define NStringReplaceStrOrChars N_FUNC_AW(NStringReplaceStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.101 NStringSplit Macro

Returns a string array that contains the substrings that are delimited by elements of a specified character array.

C++

```
#define NStringSplit(hString, arSeparators, separatorCount, removeEmptyEntries, parhValues,
pValueCount) NStringSplitWithCount(hString, arSeparators, separatorCount, N_INT_MAX,
removeEmptyEntries, parhValues, pValueCount)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
arSeparators	[in] An array of characters that delimit the substrings in *hString.*
separatorCount	[in] Separators count in *arSeparators*.
removeEmptyEntries	[in] NTrue [1064] - empty array elements from the array returned are omitted; NFalse [1041] - empty array elements in the array returned are included.
parhValues	[out] An array whose elements contain the substrings in *hString* that are delimited by one or more chars in separator.
pValueCount	[out] *parhValues* array size.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.102 NStringSplitWithCount Macro

Picks a Unicode (W) or ASCII (A) version of the NStringSplitWithCount function.

C++

```
#define NStringSplitWithCount N_FUNC_AW(NStringSplitWithCount)
```

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.3.103 NStringSplitWithStrs Macro

Returns a string array that contains the substrings that are delimited by elements of a specified character array.

C++

```
#define NStringSplitWithStrs(hString, arszSeparators, separatorCount, removeEmptyEntries,
parhValues, pValueCount) NStringSplitWithStrsAndCount(hString, arszSeparators,
separatorCount, N_INT_MAX, removeEmptyEntries, parhValues, pValueCount)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
arszSeparators	[in] A pointer to an array of characters that delimit the substrings in *hString*.
separatorCount	[in] Separators count in *arszSeparators*.
removeEmptyEntries	[in] NTrue [1064] - empty array elements from the array returned are omitted; NFalse [1041] - empty array elements in the array returned are included.
parhValues	[out] An array whose elements contain the substrings in *hString* that are delimited by one or more chars in separator.
pValueCount	[out] *parhValues* array size.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.104 NStringSplitWithStrsAndCount Macro

Picks a Unicode (W) or ASCII (A) version of the NStringSplitWithStrsAndCount function.

C++

```
#define NStringSplitWithStrsAndCount N_FUNC_AW(NStringSplitWithStrsAndCount)
```

Module

NString Unit [678]

7.1.2.52.3.105 NStringSplitWithStrsN Macro

Returns a string array that contains the substrings that are delimited by elements of a specified string array.

C++

```
#define NStringSplitWithStrsN(hString, arhSeparators, separatorCount, removeEmptyEntries,
parhValues, pValueCount) NStringSplitWithStrsAndCountN(hString, arhSeparators,
separatorCount, N_INT_MAX, removeEmptyEntries, parhValues, pValueCount)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
arhSeparators	[in] A pointer to an array of string handles that delimit the substrings in *hString*.
separatorCount	[in] Separators count in *arhSeparators*.
removeEmptyEntries	[in] NTrue [1064] - empty array elements from the array returned are omitted; NFalse [1041] - empty array elements in the array returned are included.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

parhValues	[out] An array whose elements contain the substrings in *hString* that are delimited by one or more chars in separator.
pValueCount	[out] *parhValues* array size.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.106 NStringStartsWith Macro

Checks whether the beginning of *hString* matches the specified string.

C++

```
#define NStringStartsWith(hString, szValue, ignoreCase, pResult)
NStringStartsWithStrOrChars(hString, szValue, -1, ignoreCase, pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
szValue	[in] Pointer to a string to compare.
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pResult	[out] Check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.107 NStringStartsWithChars Macro

Checks whether the beginning of *hString* matches the specified charset.

C++

```
#define NStringStartsWithChars(hString, arValue, valueLength, ignoreCase, pResult)
NStringStartsWithStrOrChars(hString, arValue, valueLength, ignoreCase, pResult)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
arValue	[in] Pointer to a char array to compare.
valueLength	[in] Length of *arValue.*
ignoreCase	[in] If NTrue [1064] strings comparison is not case sensitive.
pResult	[out] Check result.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.3.108 NStringStartsWithStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringStartsWithStrOrChars function.

C++

```
#define NStringStartsWithStrOrChars N_FUNC_AW(NStringStartsWithStrOrChars)
```

Module

NString Unit [678]

7.1.2.52.3.109 NStringSubstring Macro

Picks a Unicode (W) or ASCII (A) version of the NStringSubstring function.

C++

```
#define NStringSubstring N_FUNC_AW(NStringSubstring)
```

Module

NString Unit [678]

7.1.2.52.3.110 NStringSubstringFrom Macro

Picks a Unicode (W) or ASCII (A) version of the NStringSubstringFrom function.

C++

```
#define NStringSubstringFrom N_FUNC_AW(NStringSubstringFrom)
```

Module

NString Unit [678]

7.1.2.52.3.111 NStringToCharArray Macro

Converts a string to character array.

C++

```
#define NStringToCharArray(hString, parValue, pLength) NStringToStrOrCharArray(hString,
NFalse, parValue, pLength)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
parValue	[out] Pointer to a pointer to the resulting char array.
pLength	[out] Pointer to the length of the resulting char array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.112 NStringToCharArrayRange Macro

Converts a string to a part of character array.

C++

```
#define NStringToCharArrayRange(hString, startIndex, length, parValue, pLength)
NStringToStrOrCharArrayRange(hString, NFalse, startIndex, length, parValue, pLength)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
nullTerminate	[in] NTrue [1064] if null values are to be terminated; else NFalse [1041].
startIndex	[in] Starting index in the array, at which the converted characters are being inserted.
length	[in] Length of the range into which the converted characters are being inserted.
parValue	[out] Pointer to a pointer to the resulting char array.
pLength	[out] Pointer to the length of the resulting char array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.113 NStringToNewString Macro

Converts a string to another string.

C++

```
#define NStringToNewString(hString, pszValue, pLength) NStringToStrOrCharArray(hString,
NTrue, pszValue, pLength)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
pszValues	[out] Pointer to a pointer to the resulting string.
pLength	[out] Pointer to the length of the resulting string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.114 NStringToNewStringRange Macro

Converts a string to another string or a part of character array.

C++

```
#define NStringToNewStringRange(hString, startIndex, length, pszValue, pLength)
NStringToStrOrCharArrayRange(hString, NTrue, startIndex, length, pszValue, pLength)
```

Parameters

Parameters	Description
hString	[in] Handle to NString [678].
startIndex	[in] Starting index in the string, at which the converted characters are being inserted.
length	[in] Length of the range into which the converted characters are being inserted.
pszValues	[out] Pointer to a pointer to the resulting string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

pLength	[out] Pointer to the length of the resulting string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NString Unit [678]

7.1.2.52.3.115 NStringToString Macro

Picks a Unicode (W) or ASCII (A) version of the NStringToString function.

C++

```
#define NStringToString N_FUNC_AW(NStringToString)
```

Module

NString Unit [678]

7.1.2.52.3.116 NStringToStringOld Macro

Picks a Unicode (W) or ASCII (A) version of the NStringToStringOld function.

C++

```
#define NStringToStringOld N_FUNC_AW(NStringToStringOld)
```

Module

NString Unit [678]

7.1.2.52.3.117 NStringToStrOrCharArray Macro

Picks a Unicode (W) or ASCII (A) version of the NStringToStrOrCharArray function.

C++

```
#define NStringToStrOrCharArray N_FUNC_AW(NStringToStrOrCharArray)
```

Module

NString Unit [678]

7.1.2.52.3.118 NStringToStrOrCharArrayRange Macro

Picks a Unicode (W) or ASCII (A) version of the NStringToStrOrCharArrayRange function.

C++

```
#define NStringToStrOrCharArrayRange N_FUNC_AW(NStringToStrOrCharArrayRange)
```

Module

NString Unit [678]

7.1.2.52.3.119 NStringTrimAny Macro

Picks a Unicode (W) or ASCII (A) version of the NStringTrimAny function.

C++

```
#define NStringTrimAny N_FUNC_AW(NStringTrimAny)
```

Module

NString Unit [678]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.52.3.120 NStringTrimEndAny Macro

Picks a Unicode (W) or ASCII (A) version of the NStringTrimEndAny function.

C++

```
#define NStringTrimEndAny N_FUNC_AW(NStringTrimEndAny)
```

Module

NString Unit [678]

7.1.2.52.3.121 NStringTrimStartAny Macro

Picks a Unicode (W) or ASCII (A) version of the NStringTrimStartAny function.

C++

```
#define NStringTrimStartAny N_FUNC_AW(NStringTrimStartAny)
```

Module

NString Unit [678]

7.1.2.52.4 Files

7.1.2.52.4.1 NString.h

Header file for the NString [678] unit.

Functions

	Name	Description
	NStringAppendStrOrChars [687]	Extends specified string content by adding an additional string or chars at its end.
	NStringClone [687]	Clones specified NString [678].
	NStringCompareRangeNA [688]	Compares substrings of the two specified strings and retrieves their relative position in the sort order.
	NStringCompareRangeNW [688]	Compares substrings of the two specified strings and retrieves their relative position in the sort order.
	NStringCompareRangeStrOrCharsA [689]	Compares substrings of a string and a string or a charset and retrieves their relative position in the sort order.
	NStringCompareRangeStrOrCharsW [689]	Compares substrings of a string and a string or a charset and retrieves their relative position in the sort order.
	NStringConcatArray [690]	Concatenates strings specified by array.
	NStringConcatMany [690]	Concatenates specified strings.
	NStringConcatManyVA [691]	Concatenates specified strings.
	NStringConcatN [691]	Concatenates strings specified by handles.
	NStringConcatStrOrChars [692]	Concatenates strings or charsets specified by handles.
	NStringContainsStrN [692]	Checks whether specified string occurs within other string.
	NStringContainsStrOrChars [692]	Returns a value indicating whether specified chars or string occurs within string specified by handle.
	NStringCopyToStrOrChars [693]	Copies a specified number of characters from a specified position in handle to a specified array.
	NStringCreateFromChar [693]	Creates string from specified character.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NStringCreateFromStrOrChars [694]	Creates string from specified array of characters.
	NStringCreateWrapper [694]	Creates a new string handle based on the specified string.
	NStringEndsWithN [694]	Checks whether the end of the string matches the specified string.
	NStringEndsWithStrOrCharsA [695]	Checks whether the end of the string matches the specified string or charset.
	NStringEndsWithStrOrCharsW [695]	Checks whether the end of the string matches the specified string or charset.
	NStringEqualsN [696]	Checks whether two specified strings have the same value.
	NStringEqualsStrOrCharsA [696]	Checks whether two specified strings or charsets have the same value.
	NStringEqualsStrOrCharsW [696]	Checks whether two specified strings or charsets have the same value.
	NStringFormat [697]	Returns a string representing the specified format and parameters.
	NStringFormatVA [697]	Returns a string representing the specified format and arguments.
	NStringFree [698]	Deletes the string specified by handle. After the string is deleted the specified handle is no longer valid.
	NStringFreeArray [698]	Deletes array of strings specified by handles. After the strings is deleted the specified object handles are no longer valid.
	NStringFreeElements [698]	Deletes strings specified by handles in array. After the strings is deleted the specified string handles are no longer valid.
	NStringGetBuffer [698]	Gets characters buffer (string) from specified handle to NString [678].
	NStringGetEmpty [699]	Gets empty string.
	NStringGetHashCode [699]	Gets the hash code
	NStringGetLength [699]	Gets length of specified string.
	NStringHasEmbeddedNull [700]	Checked whether the specified string has an embedded null value.
	NStringIndexOf [700]	Returns the zero-based index of the first occurrence of the specified character in the string.
	NStringIndexOfAny [701]	Returns the zero-based index of the first occurrence in the string of any character in a specified array of characters.
	NStringIndexOfAnyFrom [701]	Returns the zero-based index of the first occurrence in the string of any character in a specified array of characters.
	NStringIndexOfAnyInRange [701]	Returns the zero-based index of the first occurrence in the string of any character in a specified array of characters.
	NStringIndexOfFrom [702]	Returns the zero-based index of the first occurrence of the specified character in the string starting from specified index.
	NStringIndexOfInRange [702]	Returns the zero-based index of the first character's occurrence in string's range.
	NStringIndexOfStrFromN [703]	Returns the zero-based index of the first occurrence of the specified string in the string starting from specified index.
	NStringIndexOfStrInRangeN [703]	Returns the zero-based index of the first occurrence of the specified string in the string's range.
	NStringIndexOfStrN [704]	Returns the zero-based index of the first occurrence of the specified NString [678] in the string.
	NStringIndexOfStrOrChars [704]	Returns the zero-based index of the first occurrence of the specified string or character in the string.
	NStringIndexOfStrOrCharsFrom [705]	Returns the zero-based index of the first occurrence of the specified string or character in the string.
	NStringIndexOfStrOrCharsInRange [705]	Returns the zero-based index of the first occurrence of the specified string or character in the string.
	NStringInsertN [706]	Returns a new string in which a specified string is inserted at a specified index position.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NStringInsertStrOrChars [706]	Returns a new string in which a specified string or characters is inserted at a specified index position.
	NStringIsA [706]	Checks whether specified string contains ANSI characters.
	NStringIsEmpty [707]	Checks whether specified string is empty string.
	NStringIsW [707]	Checks whether specified string contains Unicode characters.
	NStringIsWhiteSpace [707]	Checks whether specified string consists only of white-space characters.
	NStringJoinArray [708]	Concatenates all the elements of a string array, using the specified separator between each element.
	NStringJoinArrayVN [708]	Concatenates all the elements of a string array, using the specified separator between each element.
	NStringJoinMany [709]	Concatenates all the elements of a string arrays, using the specified separator between each element.
	NStringJoinManyVA [709]	Concatenates all the elements of a string arrays, using the specified separator between each element.
	NStringJoinManyVAVN [709]	Concatenates all the elements of a string arrays, using the specified separator between each element.
	NStringJoinManyVN [710]	Concatenates all the elements of a string arrays, using the specified separator between each element.
	NStringLastIndexOf [710]	Returns the zero-based index position of the last occurrence of a specified character within string.
	NStringLastIndexOfAny [710]	Returns the zero-based index position of the last occurrence in the string of one or more characters specified in an array.
	NStringLastIndexOfAnyFrom [711]	Returns the zero-based index position of the last occurrence in the string of one or more characters specified in an array.
	NStringLastIndexOfAnyInRange [711]	Returns the zero-based index position of the last occurrence in the string's range of one or more characters specified in an array.
	NStringLastIndexOfFrom [712]	Returns the zero-based index of the last occurrence of the specified character in the string starting from specified index.
	NStringLastIndexOfInRange [712]	Returns the zero-based index of the last character's occurrence in string's range.
	NStringLastIndexOfStrFromN [713]	Returns the zero-based index of the last occurrence of the specified string in the NString [678] starting from specified index.
	NStringLastIndexOfStrInRangeN [713]	Returns the zero-based index of the last occurrence of the specified string in the NString [678] in specified range.
	NStringLastIndexOfStrN [714]	Returns the zero-based index of the last occurrence of the specified NString [678] in the string.
	NStringLastIndexOfStrOrChars [714]	Returns the zero-based index of the last occurrence of the specified string or character in the string.
	NStringLastIndexOfStrOrCharsFrom [715]	Returns the zero-based index of the last occurrence of the specified string or character in the string.
	NStringLastIndexOfStrOrCharsInRange [715]	Returns the zero-based index of the last occurrence of the specified string or character in the string.
	NStringPadLeft [716]	Returns a new string that right-aligns the characters by padding them with spaces on the left, for a specified total length.
	NStringPadLeftWith [716]	Returns a new string that right-aligns the characters by padding them on the left with a specified character, for a specified total length.
	NStringPadRight [717]	Returns a new string that left-aligns the characters by padding them with spaces on the right, for a specified total length.
	NStringPadRightWith [717]	Returns a new string that left-aligns the characters by padding them on the right with a specified character, for a specified total length.
	NStringPrependStrOrChars [718]	Prepends a string or a charset to another string.
	NStringPromoteBuffer [718]	Creates a new NString [678] which takes ownership of the memory buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NStringRemove [718]	Returns a new string in which all the characters in *hString*, beginning at a specified position and continuing through the last position, have been deleted.
	NStringRemoveFrom [719]	Returns a new string in which all the characters in *hString*, beginning at a specified position have been deleted.
	NStringReplace [719]	Returns a new string in which all occurrences of a specified character in *hString* are replaced with another specified character.
	NStringReplaceStrOrChars [720]	Replaces a part of a string or a char array with new values.
	NStringSet [720]	Sets the value of the specified string.
	NStringSplitWithCount [720]	Returns a string array that contains the substrings that are delimited by elements of a specified character array. A parameter specifies the maximum number of substrings to return.
	NStringSplitWithStrsAndCount [721]	Returns a string array that contains the substrings that are delimited by elements of a specified character array. A parameter specifies the maximum number of substrings to return.
	NStringSplitWithStrsAndCountN [721]	Returns a string array that contains the substrings that are delimited by elements of a specified string array. A parameter specifies the maximum number of substrings to return.
	NStringStartsWithN [722]	Checks whether the beginning of *hString* matches the specified string.
	NStringStartsWithStrOrCharsA [722]	Checks whether the beginning of *hString* matches the specified string or chars.
	NStringStartsWithStrOrCharsW [723]	Checks whether the beginning of *hString* matches the specified string or chars.
	NStringSubstring [723]	Gets a substring from *hString*. The substring starts at a specified character position.
	NStringSubstringFrom [724]	Gets a substring from *hString*. The substring starts at a specified character position.
	NStringToLower [724]	Converts specified string to lowercase.
	NStringToString [724]	Converts the specified string to another string with specified format.
	NStringToStringOld [725]	Converts the specified string to charset.
	NStringToStrOrCharArray [725]	Converts a string to another string or a character array.
	NStringToStrOrCharArrayRange [726]	Converts a string to another string or a part of character array.
	NStringToUpper [726]	Converts specified string to uppercase.
	NStringTrim [726]	Removes all leading and trailing white-space characters from specified string.
	NStringTrimAny [727]	Removes all leading and trailing occurrences of a sets of characters from specified string.
	NStringTrimEnd [727]	Removes all trailing characters from *hString*.
	NStringTrimEndAny [727]	Removes all trailing occurrences of a set of characters specified in an array from *hString*.
	NStringTrimStart [728]	Removes all trailing characters from *hString*.
	NStringTrimStartAny [728]	Removes all leading occurrences of a set of characters specified in an array from *hString*.

Macros

Name	Description
N_NEW_LINE [729]	Picks a Unicode (W) or ASCII (A) version of the N_NEW_LINE macro.
N_STRING_SIZE [729]	Defines the string size.
NStringAppend [729]	Extends specified string content by adding an additional string at its end.
NStringAppendChars [729]	Extends specified string content by adding additional chars at its end.
NStringAppendStrOrChars [730]	Picks a Unicode (W) or ASCII (A) version of the NStringAppendStrOrChars function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringCompare [730]	Compares two strings and retrieves their relative position in the sort order.
NStringCompareChars [730]	Compares a string with a char array and retrieves their relative position in the sort order.
NStringCompareN [731]	Compares two strings and retrieves their relative position in the sort order.
NStringCompareRange [731]	Compares substrings of the two specified strings and retrieves their relative position in the sort order.
NStringCompareRangeChars [732]	Compares substrings of a string and a charset and retrieves their relative position in the sort order.
NStringCompareRangeN [732]	Picks a Unicode (W) or ASCII (A) version of the NStringCompareRangeN function.
NStringCompareRangeStrOrChars [732]	Picks a Unicode (W) or ASCII (A) version of the NStringCompareRangeStrOrChars function.
NStringCompareStrOrChars [732]	Compares a string with a string or char array and retrieves their relative position in the sort order.
NStringConcat [733]	Concatenates strings specified by handles.
NStringConcatArray [733]	Picks a Unicode (W) or ASCII (A) version of the NStringConcatArray function.
NStringConcatChars [733]	Concatenates charsets specified by handles.
NStringConcatMany [734]	Picks a Unicode (W) or ASCII (A) version of the NStringConcatMany function.
NStringConcatManyVA [734]	Picks a Unicode (W) or ASCII (A) version of the NStringConcatManyVA function.
NStringConcatStrOrChars [734]	Picks a Unicode (W) or ASCII (A) version of the NStringConcatStrOrChars function.
NStringContainsChars [734]	Returns a value indicating whether specified chars occur within string specified by handle.
NStringContainsStr [735]	Returns a value indicating whether specified string occurs within string specified by handle.
NStringContainsStrOrChars [735]	Picks a Unicode (W) or ASCII (A) version of the NStringContainsStrOrChars function.
NStringCopyTo [735]	Copies a specified number of characters from a specified position in handle to a specified array.
NStringCopyToString [736]	Copies a specified number of characters from a specified position in handle to a specified string.
NStringCopyToStrOrChars [736]	Picks a Unicode (W) or ASCII (A) version of the NStringCopyToStrOrChars function.
NStringCreate [736]	Creates a string with the specified value.
NStringCreateFromChar [737]	Picks a Unicode (W) or ASCII (A) version of the NStringCreateFromChar function.
NStringCreateFromChars [737]	Creates a string from the specified array of characters.
NStringCreateFromStrOrChars [737]	Picks a Unicode (W) or ASCII (A) version of the NStringCreateFromStrOrChars function.
NStringCreateWrapper [738]	Picks a Unicode (W) or ASCII (A) version of the NStringCreateWrapper function.
NStringEndsWith [738]	Checks whether the end of the string matches the specified string.
NStringEndsWithChars [738]	Checks whether the end of the string matches the specified string or charset.
NStringEndsWithStrOrChars [739]	Picks a Unicode (W) or ASCII (A) version of the NStringEndsWithStrOrChars function.
NStringEquals [739]	Checks whether two specified stings have the same value.
NStringEqualsChars [739]	Checks whether the specified string and the specified charset have the same value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringEqualsStrOrChars [740]	Picks a Unicode (W) or ASCII (A) version of the NStringEqualsStrOrChars function.
NStringFormat [740]	Picks a Unicode (W) or ASCII (A) version of the NStringFormat function.
NStringFormatVA [740]	Picks a Unicode (W) or ASCII (A) version of the NStringFormatVA function.
NStringGetBuffer [740]	Picks a Unicode (W) or ASCII (A) version of the NStringGetBuffer function.
NStringGetChar [740]	Picks a Unicode (W) or ASCII (A) version of the NStringGetChar function.
NStringGetLength [740]	Picks a Unicode (W) or ASCII (A) version of the NStringGetLength function.
NStringIndexOf [741]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOf function.
NStringIndexOfAny [741]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfAny function.
NStringIndexOfAnyFrom [741]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfAnyFrom function.
NStringIndexOfAnyInRange [741]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfAnyInRange function.
NStringIndexOfChars [741]	Returns the zero-based index of the first occurrence of the specified character in the string.
NStringIndexOfCharsFrom [742]	Returns the zero-based index of the first occurrence of the specified character in the string.
NStringIndexOfCharsInRange [742]	Returns the zero-based index of the first occurrence of the specified character in the string.
NStringIndexOfFrom [743]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfFrom function.
NStringIndexOfInRange [743]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfInRange function.
NStringIndexOfStr [743]	Returns the zero-based index of the first occurrence of the specified string in the string.
NStringIndexOfStrFrom [744]	Returns the zero-based index of the first occurrence of the specified string in the string.
NStringIndexOfStrFromN [744]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrFromN function.
NStringIndexOfStrInRange [744]	Returns the zero-based index of the first occurrence of the specified string in the string.
NStringIndexOfStrInRangeN [745]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrInRangeN function.
NStringIndexOfStrN [745]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrN function.
NStringIndexOfStrOrChars [745]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrOrChars function.
NStringIndexOfStrOrCharsFrom [745]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrOrCharsFrom function.
NStringIndexOfStrOrCharsInRange [745]	Picks a Unicode (W) or ASCII (A) version of the NStringIndexOfStrOrCharsInRange function.
NStringInsert [745]	Returns a new string in which a specified string is inserted at a specified index position.
NStringInsertChars [746]	Returns a new string in which a specified characters are inserted at a specified index position.
NStringInsertN [746]	Picks a Unicode (W) or ASCII (A) version of the NStringInsertN function.
NStringInsertStrOrChars [746]	Picks a Unicode (W) or ASCII (A) version of the NStringInsertStrOrChars function.
NStringJoinArray [747]	Picks a Unicode (W) or ASCII (A) version of the NStringJoinArray function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringJoinArrayVN [747]	Picks a Unicode (W) or ASCII (A) version of the NStringJoinArrayVN function.
NStringJoinMany [747]	Picks a Unicode (W) or ASCII (A) version of the NStringJoinMany function.
NStringJoinManyVAVN [747]	Picks a Unicode (W) or ASCII (A) version of the NStringJoinManyVAVN function.
NStringJoinManyVN [747]	Picks a Unicode (W) or ASCII (A) version of the NStringJoinManyVN function.
NStringLastIndexOf [748]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOf function.
NStringLastIndexOfAny [748]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfAny function.
NStringLastIndexOfAnyFrom [748]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfAnyFrom function.
NStringLastIndexOfAnyInRange [748]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfAnyInRange function.
NStringLastIndexOfChars [748]	Returns the zero-based index of the last occurrence of the specified character in the string.
NStringLastIndexOfCharsFrom [749]	Returns the zero-based index of the last occurrence of the specified character in the string.
NStringLastIndexOfCharsInRange [749]	Returns the zero-based index of the last occurrence of the specified character in the string.
NStringLastIndexOfFrom [750]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfFrom function.
NStringLastIndexOfInRange [750]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfInRange function.
NStringLastIndexOfStr [750]	Returns the zero-based index of the last occurrence of the specified string in the string.
NStringLastIndexOfStrFrom [750]	Returns the zero-based index of the last occurrence of the specified string in the string.
NStringLastIndexOfStrFromN [751]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrFromN function.
NStringLastIndexOfStrInRange [751]	Returns the zero-based index of the last occurrence of the specified string in the string.
NStringLastIndexOfStrInRangeN [752]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrInRangeN function.
NStringLastIndexOfStrN [752]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrN function.
NStringLastIndexOfStrOrChars [752]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrOrChars function.
NStringLastIndexOfStrOrCharsFrom [752]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrOrCharsFrom function.
NStringLastIndexOfStrOrCharsInRange [752]	Picks a Unicode (W) or ASCII (A) version of the NStringLastIndexOfStrOrCharsInRange function.
NStringPadLeft [752]	Picks a Unicode (W) or ASCII (A) version of the NStringPadLeft function.
NStringPadLeftWith [753]	Picks a Unicode (W) or ASCII (A) version of the NStringPadLeftWith function.
NStringPadRight [753]	Picks a Unicode (W) or ASCII (A) version of the NStringPadRight function.
NStringPadRightWith [753]	Picks a Unicode (W) or ASCII (A) version of the NStringPadRightWith function.
NStringPrepend [753]	Prepends a string to another string.
NStringPrependChars [754]	Prepends a charset to another string.
NStringPrependStrOrChars [754]	Picks a Unicode (W) or ASCII (A) version of the NStringPrependStrOrChars function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringPromoteBuffer [754]	Picks a Unicode (W) or ASCII (A) version of the NStringPromoteBuffer function.
NStringRemove [754]	Picks a Unicode (W) or ASCII (A) version of the NStringRemove function.
NStringRemoveFrom [754]	Picks a Unicode (W) or ASCII (A) version of the NStringRemoveFrom function.
NStringReplace [755]	Picks a Unicode (W) or ASCII (A) version of the NStringReplace function.
NStringReplaceChars [755]	Replaces a part of a char array with new values.
NStringReplaceStr [755]	Replaces a part of a string with new values.
NStringReplaceStrOrChars [756]	Picks a Unicode (W) or ASCII (A) version of the NStringReplaceStrOrChars function.
NStringSplit [756]	Returns a string array that contains the substrings that are delimited by elements of a specified character array.
NStringSplitWithCount [756]	Picks a Unicode (W) or ASCII (A) version of the NStringSplitWithCount function.
NStringSplitWithStrs [757]	Returns a string array that contains the substrings that are delimited by elements of a specified character array.
NStringSplitWithStrsAndCount [757]	Picks a Unicode (W) or ASCII (A) version of the NStringSplitWithStrsAndCount function.
NStringSplitWithStrsN [757]	Returns a string array that contains the substrings that are delimited by elements of a specified string array.
NStringStartsWith [758]	Checks whether the beginning of *hString* matches the specified string.
NStringStartsWithChars [758]	Checks whether the beginning of *hString* matches the specified charset.
NStringStartsWithStrOrChars [759]	Picks a Unicode (W) or ASCII (A) version of the NStringStartsWithStrOrChars function.
NStringSubstring [759]	Picks a Unicode (W) or ASCII (A) version of the NStringSubstring function.
NStringSubstringFrom [759]	Picks a Unicode (W) or ASCII (A) version of the NStringSubstringFrom function.
NStringToCharArray [759]	Converts a string to character array.
NStringToCharArrayRange [759]	Converts a string to a part of character array.
NStringToNewString [760]	Converts a string to another string.
NStringToNewStringRange [760]	Converts a string to another string or a part of character array.
NStringToString [761]	Picks a Unicode (W) or ASCII (A) version of the NStringToString function.
NStringToStringOld [761]	Picks a Unicode (W) or ASCII (A) version of the NStringToStringOld function.
NStringToStrOrCharArray [761]	Picks a Unicode (W) or ASCII (A) version of the NStringToStrOrCharArray function.
NStringToStrOrCharArrayRange [761]	Picks a Unicode (W) or ASCII (A) version of the NStringToStrOrCharArrayRange function.
NStringTrimAny [761]	Picks a Unicode (W) or ASCII (A) version of the NStringTrimAny function.
NStringTrimEndAny [762]	Picks a Unicode (W) or ASCII (A) version of the NStringTrimEndAny function.
NStringTrimStartAny [762]	Picks a Unicode (W) or ASCII (A) version of the NStringTrimStartAny function.

Module

NString Unit [678]

7.1.2.53 NStringBuilder Unit

Represents a mutable string of characters.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NStringBuilderAppend [776]	Appends a the copy of the specified string to the specified string builder.
	NStringBuilderAppendBoolean [776]	Appends a the string representation of specified Boolean value to the specified string builder.
	NStringBuilderAppendByte [776]	Appends a the string representation of specified unsigned byte integer to the specified string builder.
	NStringBuilderAppendChar [777]	Appends a the string representation of specified char to the specified string builder.
	NStringBuilderAppendChars [777]	Appends a the string representation of specified char arry to the specified string builder.
	NStringBuilderAppendDouble [777]	Appends a the string representation of specified double integer to the specified string builder.
	NStringBuilderAppendEmptyLine [778]	Appends the default line terminator to the end of the specified NStringBuilder object.
	NStringBuilderAppendFormat [778]	Appends a format string with specified parameters to the specified string builder.
	NStringBuilderAppendFormatN [778]	Appends a format string with specified parameters to the specified string builder.
	NStringBuilderAppendFormatVA [779]	Appends a format string with specified arguments to the specified string builder.
	NStringBuilderAppendFormatVAN [779]	Appends a format string with specified arguments to the specified string builder.
	NStringBuilderAppendInt16 [780]	Appends a the string representation of specified short integer to the specified string builder.
	NStringBuilderAppendInt32 [780]	Appends a the string representation of specified integer to the specified string builder.
	NStringBuilderAppendInt64 [780]	Appends a the string representation of specified long integer to the specified string builder.
	NStringBuilderAppendLine [781]	Appends a the copy of the specified string followed by the default line terminator to the end of the specified string builder.
	NStringBuilderAppendLineN [781]	Appends a the copy of the specified string followed by the default line terminator to the end of the specified string builder.
	NStringBuilderAppendN [781]	Appends a the copy of the specified string to the specified string builder.
	NStringBuilderAppendObject [782]	Appends a the string representation of specified object to the specified string builder.
	NStringBuilderAppendPointer [782]	Appends a the string representation of specified pointer to the specified string builder.
	NStringBuilderAppendResult [782]	Appends a string representation of a NResult [1007] value to the specified string builder.
	NStringBuilderAppendSByte [783]	Appends a the string representation of specified integer to the specified string builder.
	NStringBuilderAppendSingle [783]	Appends a the string representation of specified floating point number to the specified string builder.
	NStringBuilderAppendSizeType [783]	Appends a the string representation of specified SizeType to the specified string builder.
	NStringBuilderAppendSSizeType [784]	Appends a the string representation of specified SSizeType to the specified string builder.
	NStringBuilderAppendUInt16 [784]	Appends a the string representation of specified unsigned short integer to the specified string builder.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NStringBuilderAppendUInt32 [784]	Appends a the string representation of specified unsigned integer to the specified string builder.
	NStringBuilderAppendUInt64 [785]	Appends a the string representation of specified unsigned long integer to the specified string builder.
	NStringBuilderAppendValue [785]	Appends a string representation of a value of specified type to the specified string builder.
	NStringBuilderAppendValueP [785]	Appends a the string representation of specified value to the specified string builder.
	NStringBuilderClear [786]	Removes all characters from the specified NStringBuilder.
	NStringBuilderCopyToStrOrChars [786]	Copies the characters from a specified segment of the NStringBuilder to a specified segment of a destination Char array.
	NStringBuilderDetachStringN [786]	Detaches a string from the NStringBuilder.
	NStringBuilderDetachStrOrChars [787]	Detaches a string or a char array from the NStringBuilder.
	NStringBuilderDispose [787]	Releases the unused resources of the specified NStringBuilder.
	NStringBuilderGetCapacity [787]	Gets the capacity of the specified NStringBuilder.
	NStringBuilderGetChar [788]	Gets the character at the specified position in the specified NStringBuilder.
	NStringBuilderGetLength [788]	Gets the length of the specified NStringBuilder.
	NStringBuilderGetStringWrapper [788]	Gets the string wrapper of the specified NStringBuilder.
	NStringBuilderGetStrOrCharsBuffer [789]	Gets the string or character array buffer of the specified NStringBuilder.
	NStringBuilderInit [789]	Creates a new NStringBuilder.
	NStringBuilderInitEx [790]	Creates a new NStringBuilder with specified capacity, maximum capacity and growth delta.
	NStringBuilderInitWithCapacity [790]	Creates a new NStringBuilder using the specified capacity.
	NStringBuilderInitWithStringN [790]	Creates a new NStringBuilder using the specified string.
	NStringBuilderInitWithStrOrChars [791]	Creates a new NStringBuilder using the specified string or char array.
	NStringBuilderInsert [791]	Inserts a string into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertBoolean [791]	
	NStringBuilderInsertByte [792]	Inserts a string representation of the specified unsigned byte integer into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertChar [792]	Inserts a string representation of the specified char into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertChars [792]	Inserts a string representation of the specified char array into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertDouble [793]	Inserts a string representation of the specified double integer into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertInt16 [793]	Inserts a string representation of the specified short integer into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertInt32 [794]	Inserts a string representation of the specified integer into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertInt64 [794]	Inserts a string representation of the specified long integer into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertN [794]	Inserts a string into the specified NStringBuilder at the specified character position.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NStringBuilderInsertObject [795]	Inserts a string representation of the specified object into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertPointer [795]	Inserts a string representation of the specified pointer into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertResult [796]	Inserts a string representation of a NResult [1007] value into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertSByte [796]	Inserts a string representation of the specified byte integer into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertSingle [796]	Inserts a string representation of the specified floating point number into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertSizeType [797]	Inserts a string representation of the specified SizeType into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertSSizeType [797]	Inserts a string representation of the specified SSizeType into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertUInt16 [797]	Inserts a string representation of the specified unsigned short integer into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertUInt32 [798]	Inserts a string representation of the specified unsigned integer into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertUInt64 [798]	Inserts a string representation of the specified long integer into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertValue [799]	Inserts a string representation of a value of specified type into the specified NStringBuilder at the specified character position.
	NStringBuilderInsertValueP [799]	Inserts a string representation of the specified value into the specified NStringBuilder at the specified character position.
	NStringBuilderRemove [799]	Removes the specified range of characters from the specified NStringBuilder.
	NStringBuilderReplace [800]	Replaces all occurrences of a specified character in the specified NStringBuilder with another specified character.
	NStringBuilderReplaceInRange [800]	Replaces, within a substring of the specified NStringBuilder, all occurrences of a specified character with another specified character.
	NStringBuilderReplaceStringInRangeN [801]	Replaces, within a substring of the specified NStringBuilder, all occurrences of a specified string with another specified string.
	NStringBuilderReplaceStringN [801]	Replaces all occurrences of a specified string in the specified NStringBuilder with another specified character.
	NStringBuilderReplaceStrOrChars [801]	Replaces all occurrences of a specified string or char array in the specified NStringBuilder with another specified string or char array.
	NStringBuilderReplaceStrOrCharsInRange [802]	Replaces, within a substring of the specified NStringBuilder, all occurrences of a specified string or char array with another specified string or char array.
	NStringBuilderSetCapacity [802]	Sets the capacity of the specified NStringBuilder.
	NStringBuilderSetChar [803]	Sets the value of the character at the specified position in the specified NStringBuilder.
	NStringBuilderSetLength [803]	Sets the length of the specified NStringBuilder.
	NStringBuilderToString [803]	Converts the specified NStringBuilder to a string.
	NStringBuilderToStringRange [804]	Converts the specified NStringBuilder to a string range.

Macros

Name	Description
N_STRING_BUILDER_SIZE [804]	Defines the NStringBuilder size.
NStringBuilderAppend [804]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppend function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringBuilderAppendBoolean [805]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendBoolean function.
NStringBuilderAppendByte [805]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendByte function.
NStringBuilderAppendChar [805]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendChar function.
NStringBuilderAppendChars [805]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendChars function.
NStringBuilderAppendDouble [805]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendDouble function.
NStringBuilderAppendEmptyLine [805]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendEmptyLine function.
NStringBuilderAppendFormat [806]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendFormat function.
NStringBuilderAppendFormatN [806]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendFormatN function.
NStringBuilderAppendFormatVA [806]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendFormatVA function.
NStringBuilderAppendFormatVAN [806]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendFormatVAN function.
NStringBuilderAppendInt16 [806]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendInt16 function.
NStringBuilderAppendInt32 [807]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendInt32 function.
NStringBuilderAppendInt64 [807]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendInt64 function.
NStringBuilderAppendLine [807]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendLine function.
NStringBuilderAppendLineN [807]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendLineN function.
NStringBuilderAppendN [807]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendN function.
NStringBuilderAppendObject [807]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendObject function.
NStringBuilderAppendPointer [808]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendPointer function.
NStringBuilderAppendResult [808]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendResult function.
NStringBuilderAppendSByte [808]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendSByte function.
NStringBuilderAppendSingle [808]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendSingle function.
NStringBuilderAppendSizeType [808]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendSizeType function.
NStringBuilderAppendSSizeType [808]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendSSizeType function.
NStringBuilderAppendUInt16 [809]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendUInt16 function.
NStringBuilderAppendUInt32 [809]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendUInt32 function.
NStringBuilderAppendUInt64 [809]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendUInt64 function.
NStringBuilderAppendValue [809]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendValue function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringBuilderAppendValueP [809]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendValueP function.
NStringBuilderClear [810]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderClear function.
NStringBuilderCopyTo [810]	Copies the characters from a specified segment of the NStringBuilder to a specified segment of a destination Char array.
NStringBuilderCopyToString [810]	Copies the characters from a specified segment of the NStringBuilder to a specified segment of a destination string.
NStringBuilderCopyToStrOrChars [811]	Picks a Unicode (W) or ASCII (A) version of the NStringCopyToStrOrChars [693] function.
NStringBuilderDetachCharArray [811]	Detaches a char array from the NStringBuilder.
NStringBuilderDetachString [811]	Detaches a string from the NStringBuilder.
NStringBuilderDetachStrOrChars [811]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderDetachStrOrChars function.
NStringBuilderDispose [812]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderDispose function.
NStringBuilderGetBuffer [812]	Gets the string buffer of the specified NStringBuilder.
NStringBuilderGetCapacity [812]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderGetCapacity function.
NStringBuilderGetChar [812]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderGetChar function.
NStringBuilderGetCharsBuffer [813]	Gets the character array buffer of the specified NStringBuilder.
NStringBuilderGetLength [813]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderGetLength function.
NStringBuilderGetStringWrapper [813]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderGetStringWrapper function.
NStringBuilderGetStrOrCharsBuffer [813]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderGetStrOrCharsBuffer function.
NStringBuilderInit [813]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInit function.
NStringBuilderInitEx [814]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInitEx function.
NStringBuilderInitWithCapacity [814]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInitWithCapacity function.
NStringBuilderInitWithChars [814]	Creates a new NStringBuilder using the specified char array.
NStringBuilderInitWithString [814]	Creates a new NStringBuilder using the specified string.
NStringBuilderInitWithStringN [815]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInitWithStringN function.
NStringBuilderInitWithStrOrChars [815]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInitWithStrOrChars function.
NStringBuilderInsert [815]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsert function.
NStringBuilderInsertBoolean [815]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertBoolean function.
NStringBuilderInsertByte [815]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertByte function.
NStringBuilderInsertChar [816]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertChar function.
NStringBuilderInsertChars [816]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertChars function.
NStringBuilderInsertDouble [816]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertDouble function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringBuilderInsertInt16 [816]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertInsertInt16 function.
NStringBuilderInsertInt32 [816]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertInt32 function.
NStringBuilderInsertInt64 [816]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertInt64 function.
NStringBuilderInsertN [817]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertN function.
NStringBuilderInsertObject [817]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertObject function.
NStringBuilderInsertPointer [817]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertPointer function.
NStringBuilderInsertResult [817]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertResult function.
NStringBuilderInsertSByte [817]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertSByte function.
NStringBuilderInsertSingle [817]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertSingle function.
NStringBuilderInsertSizeType [818]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertSizeType function.
NStringBuilderInsertSSizeType [818]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertSSizeType function.
NStringBuilderInsertUInt16 [818]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertUInt16 function.
NStringBuilderInsertUInt32 [818]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertUInt32 function.
NStringBuilderInsertUInt64 [818]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertUInt64 function.
NStringBuilderRemove [819]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderRemove function.
NStringBuilderReplace [819]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderReplace function.
NStringBuilderReplaceChars [819]	Replaces all occurrences of a specified char array in the specified NStringBuilder with another specified char array.
NStringBuilderReplaceCharsInRange [819]	Replaces, within a substring of the specified NStringBuilder, all occurrences of a specified char array with another specified char array.
NStringBuilderReplaceInRange [820]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderReplaceInRange function.
NStringBuilderReplaceString [820]	Replaces all occurrences of a specified string in the specified NStringBuilder with another specified character.
NStringBuilderReplaceStringInRange [820]	Replaces, within a substring of the specified NStringBuilder, all occurrences of a specified string with another specified string.
NStringBuilderReplaceStringN [821]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderReplaceStringN function.
NStringBuilderReplaceStrOrChars [821]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderReplaceStrOrChars function.
NStringBuilderReplaceStrOrCharsInRange [821]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderReplaceStrOrCharsInRange function.
NStringBuilderSetCapacity [821]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderSetCapacity function.
NStringBuilderSetChar [822]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderSetChar function.
NStringBuilderSetLength [822]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderSetLength function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringBuilderToString [822]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderToString function.
NStringBuilderToStringRange [822]	Picks a Unicode (W) or ASCII (A) version of the NStringBuilderToStringRange function.

7.1.2.53.1 Functions

7.1.2.53.1.1 NStringBuilderAppend Function

Appends a the copy of the specified string to the specified string builder.

C++

```
NResult N_API NStringBuilderAppend(NStringBuilder * pStringBuilder, const NChar * szValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
const NChar * szValue	[in] Pointer to the string to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.2 NStringBuilderAppendBoolean Function

Appends a the string representation of specified Boolean value to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendBoolean(NStringBuilder * pStringBuilder, NBool value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NBool value	[in] The NBool [1004] to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.3 NStringBuilderAppendByte Function

Appends a the string representation of specified unsigned byte integer to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendByte(NStringBuilder * pStringBuilder, NByte value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NByte value	[in] NByte [1005] to append.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.4 NStringBuilderAppendChar Function

Appends a the string representation of specified char to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendChar(NStringBuilder * pStringBuilder, NChar value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NChar value	[in] NChar [1005] to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.5 NStringBuilderAppendChars Function

Appends a the string representation of specified char arry to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendChars(NStringBuilder * pStringBuilder, const NChar *
arValue, NInt valueLength);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
const NChar * arValue	[in] Pointer to the char array to append.
NInt valueLength	[in] Length of the char array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.6 NStringBuilderAppendDouble Function

Appends a the string representation of specified double integer to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendDouble(NStringBuilder * pStringBuilder, NDouble value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NDouble value	[in] The NDouble [1005] to append.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.7 NStringBuilderAppendEmptyLine Function

Appends the default line terminator to the end of the specified NStringBuilder [769] object.

C++

```
NResult N_API NStringBuilderAppendEmptyLine(NStringBuilder * pStringBuilder);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.8 NStringBuilderAppendFormat Function

Appends a format string with specified parameters to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendFormat(NStringBuilder * pStringBuilder, const NChar *
szFormat, ... multipleParameters);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
const NChar * szFormat	[in] Pointer to the composite format string.
... multipleParameters	[in] Format parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NStringBuilder Unit [769]

7.1.2.53.1.9 NStringBuilderAppendFormatN Function

Appends a format string with specified parameters to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendFormatN(NStringBuilder * pStringBuilder, HNString
hFormat, ... multipleParameters);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
HNString hFormat	[in] Handle to the composite format string.
... multipleParameters	[in] Format parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.10 NStringBuilderAppendFormatVA Function

Appends a format string with specified arguments to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendFormatVA(NStringBuilder * pStringBuilder, const NChar *
szFormat, va_list args);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
const NChar * szFormat	[in] Pointer to the composite format string.
va_list args	[in] Format argument list.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NStringBuilder Unit [769]

7.1.2.53.1.11 NStringBuilderAppendFormatVAN Function

Appends a format string with specified arguments to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendFormatVAN(NStringBuilder * pStringBuilder, HNString
hFormat, va_list args);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
HNString hFormat	[in] Handle to the composite format string.
va_list args	[in] Format argument list.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.1.12 NStringBuilderAppendInt16 Function

Appends a the string representation of specified short integer to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendInt16(NStringBuilder * pStringBuilder, NShort value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NShort value	[in] NShort [1008] to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.13 NStringBuilderAppendInt32 Function

Appends a the string representation of specified integer to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendInt32(NStringBuilder * pStringBuilder, NInt value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt value	[in] NInt [1006] to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.14 NStringBuilderAppendInt64 Function

Appends a the string representation of specified long integer to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendInt64(NStringBuilder * pStringBuilder, NLong value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NLong value	[in] NLong [1007] to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.1.15 NStringBuilderAppendLine Function

Appends a the copy of the specified string followed by the default line terminator to the end of the specified string builder.

C++

```
NResult N_API NStringBuilderAppendLine(NStringBuilder * pStringBuilder, const NChar *
szValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
const NChar * szValue	[in] Pointer to the string to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.16 NStringBuilderAppendLineN Function

Appends a the copy of the specified string followed by the default line terminator to the end of the specified string builder.

C++

```
NResult N_API NStringBuilderAppendLineN(NStringBuilder * pStringBuilder, HNString hValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
HNString hValue	[in] Handle to the string to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.17 NStringBuilderAppendN Function

Appends a the copy of the specified string to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendN(NStringBuilder * pStringBuilder, HNString hValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
HNString hValue	[in] Handle to the string to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.1.18 NStringBuilderAppendObject Function

Appends a the string representation of specified object to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendObject(NStringBuilder * pStringBuilder, HNObject hObject);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
HNObject hObject	[in] Handle to the object to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.19 NStringBuilderAppendPointer Function

Appends a the string representation of specified pointer to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendPointer(NStringBuilder * pStringBuilder, const void *
value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
const void * value	[in] The pointer to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.20 NStringBuilderAppendResult Function

Appends a string representation of a NResult [1007] value to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendResult(NStringBuilder * pStringBuilder, NResult value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NResult value	[in] The NResult [1007] value to be appended.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.1.21 NStringBuilderAppendSByte Function

Appends a the string representation of specified integer to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendSByte(NStringBuilder * pStringBuilder, NSByte value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NSByte value	[in] NSByte [1008] to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.22 NStringBuilderAppendSingle Function

Appends a the string representation of specified floating point number to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendSingle(NStringBuilder * pStringBuilder, NFloat value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NFloat value	[in] NFloat [1005] to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.23 NStringBuilderAppendSizeType Function

Appends a the string representation of specified SizeType to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendSizeType(NStringBuilder * pStringBuilder, NSizeType
value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NSizeType value	[in] SizeType to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.1.24 NStringBuilderAppendSSizeType Function

Appends a the string representation of specified SSizeType to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendSSizeType(NStringBuilder * pStringBuilder, NSSizeType value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NSSizeType value	[in] SSizeType to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.25 NStringBuilderAppendUInt16 Function

Appends a the string representation of specified unsigned short integer to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendUInt16(NStringBuilder * pStringBuilder, NUShort value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NUShort value	[in] NUShort [1010] to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.26 NStringBuilderAppendUInt32 Function

Appends a the string representation of specified unsigned integer to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendUInt32(NStringBuilder * pStringBuilder, NUInt value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NUInt value	[in] NUInt [1009] to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.1.27 NStringBuilderAppendUInt64 Function

Appends a the string representation of specified unsigned long integer to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendUInt64(NStringBuilder * pStringBuilder, NULong value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NULong value	[in] NULong [1010] to append.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.28 NStringBuilderAppendValue Function

Appends a string representation of a value of specified type to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendValue(NStringBuilder * pStringBuilder, HNType hType,
const void * pValue, NSizeType valueSize);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
HNType hType	[in] Handle to the type of value to be appended.
const void * pValue	[in] Pointer to the value to be appended.
NSizeType valueSize	[in] Size of the value to be appended.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.29 NStringBuilderAppendValueP Function

Appends a the string representation of specified value to the specified string builder.

C++

```
NResult N_API NStringBuilderAppendValueP(NStringBuilder * pStringBuilder, NTypeOfProc
pTypeOf, const void * pValue, NSizeType valueSize);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NTypeOfProc pTypeOf	[in] Type of the value.
const void * pValue	[in] Pointer to the value to append.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSizeType valueSize	[in] Size of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.30 NStringBuilderClear Function

Removes all characters from the specified NStringBuilder [769].

C++

```
NResult N_API NStringBuilderClear(NStringBuilder * pStringBuilder);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.31 NStringBuilderCopyToStrOrChars Function

Copies the characters from a specified segment of the NStringBuilder [769] to a specified segment of a destination Char array.

C++

```
NResult N_API NStringBuilderCopyToStrOrChars(NStringBuilder * pStringBuilder, NInt
sourceIndex, NChar * arValue, NInt valueLength, NBool nullTerminate, NInt count);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt sourceIndex	[in] Index in the NStringBuilder [769] at which the copying will start.
NChar * arValue	[out] The destination char array.
NInt valueLength	[in] Char array length.
NBool nullTerminate	[in] If NTrue [1064], null terminated string will be copied.
NInt count	[in] The number of characters to be copied.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.32 NStringBuilderDetachStringN Function

Detaches a string from the NStringBuilder [769].

C++

```
NResult N_API NStringBuilderDetachStringN(NStringBuilder * pStringBuilder, HNString *
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
phValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
HNString * phValue	[out] Pointer to a handle to the detached string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.33 NStringBuilderDetachStrOrChars Function

Detaches a string or a char array from the NStringBuilder [769].

C++

```
NResult N_API NStringBuilderDetachStrOrChars(NStringBuilder * pStringBuilder, NBool
nullTerminate, NChar * * parValue, NInt * pValueLength);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NBool nullTerminate	[in] If NTrue [1064], null terminated string will be returned.
NChar * * parValue	[out] Pointer to a detached char array.
NInt * pValueLength	[out] Detached char array length.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.34 NStringBuilderDispose Function

Releases the unused resources of the specified NStringBuilder [769].

C++

```
NResult N_API NStringBuilderDispose(NStringBuilder * pStringBuilder);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.35 NStringBuilderGetCapacity Function

Gets the capacity of the specified NStringBuilder [769].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStringBuilderGetCapacity(NStringBuilder * pStringBuilder, NInt * pValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt * pValue	[out] Pointer to the capacity of the NStringBuilder [769].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.36 NStringBuilderGetChar Function

Gets the character at the specified position in the specified NStringBuilder [769].

C++

```
NResult N_API NStringBuilderGetChar(NStringBuilder * pStringBuilder, NInt index, NChar *
pValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] Index of the character.
NChar * pValue	[out] Pointer to the character.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.37 NStringBuilderGetLength Function

Gets the length of the specified NStringBuilder [769].

C++

```
NResult N_API NStringBuilderGetLength(NStringBuilder * pStringBuilder, NInt * pValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt * pValue	[out] Pointer to the length of the NStringBuilder [769].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.38 NStringBuilderGetStringWrapper Function

Gets the string wrapper of the specified NStringBuilder [769].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStringBuilderGetStringWrapper(NStringBuilder * pStringBuilder, NStringHeader * pStringHeader, HNString * phValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NStringHeader * pStringHeader	[out] Pointer to the string header.
HNString * phValue	[out] Pointer to the string wrapper.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.39 NStringBuilderGetStrOrCharsBuffer Function

Gets the string or character array buffer of the specified NStringBuilder [769].

C++

```
NResult N_API NStringBuilderGetStrOrCharsBuffer(NStringBuilder * pStringBuilder, NBool nullTerminate, NInt * pValueLength, NChar * * parValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NBool nullTerminate	[in] If NTrue [1064], null terminated string will be returned.
NInt * pValueLength	[out] Pointer to the length of the char array.
NChar * * parValue	[out] Pointer to the char array buffer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.40 NStringBuilderInit Function

Creates a new NStringBuilder [769].

C++

```
NResult N_API NStringBuilderInit(NStringBuilder * pStringBuilder);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Pointer to a handle to the new NStringBuilder [769].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.53.1.41 NStringBuilderInitEx Function

Creates a new NStringBuilder [769] with specified capacity, maximum capacity and growth delta.

C++

```
NResult N_API NStringBuilderInitEx(NStringBuilder * pStringBuilder, NInt capacity, NInt
maxCapacity, NInt growthDelta);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Pointer to a handle to the new NStringBuilder [769].
NInt capacity	[in] Capacity value.
NInt maxCapacity	[in] Maximum capacity value.
NInt growthDelta	[in] Growth delta.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.42 NStringBuilderInitWithCapacity Function

Creates a new NStringBuilder [769] using the specified capacity.

C++

```
NResult N_API NStringBuilderInitWithCapacity(NStringBuilder * pStringBuilder, NInt
capacity);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Pointer to a handle to the new NStringBuilder [769].
NInt capacity	[in] Capacity value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.43 NStringBuilderInitWithStringN Function

Creates a new NStringBuilder [769] using the specified string.

C++

```
NResult N_API NStringBuilderInitWithStringN(NStringBuilder * pStringBuilder, HNString
hString);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Pointer to a handle to the new NStringBuilder [769].
HNString hString	[in] Handle to the string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.1.44 NStringBuilderInitWithStrOrChars Function

Creates a new NStringBuilder [769] using the specified string or char array.

C++

```
NResult N_API NStringBuilderInitWithStrOrChars(NStringBuilder * pStringBuilder, const NChar
* arValue, NInt valueLength);
```

Parameters

Parameters	Description
const NChar * arValue	[in] Pointer to the char array.
NInt valueLength	[in] Length of the char array.
phStringBuilder	[out] Pointer to a handle to the new NStringBuilder [769].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.45 NStringBuilderInsert Function

Inserts a string into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsert(NStringBuilder * pStringBuilder, NInt index, const NChar
* szValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
const NChar * szValue	[in] Pointer to the string to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.46 NStringBuilderInsertBoolean Function

C++

```
NResult N_API NStringBuilderInsertBoolean(NStringBuilder * pStringBuilder, NInt index,
NBool value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBool value	[in] NBool [1004] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.47 NStringBuilderInsertByte Function

Inserts a string representation of the specified unsigned byte integer into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertByte(NStringBuilder * pStringBuilder, NInt index, NByte value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NByte value	[in] NByte [1005] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.48 NStringBuilderInsertChar Function

Inserts a string representation of the specified char into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertChar(NStringBuilder * pStringBuilder, NInt index, NChar value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NChar value	[in] NChar [1005] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.49 NStringBuilderInsertChars Function

Inserts a string representation of the specified char array into the specified NStringBuilder [769] at the specified character position.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStringBuilderInsertChars(NStringBuilder * pStringBuilder, NInt index, const NChar * arValue, NInt valueLength);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
const NChar * arValue	[in] Pointer to the char array to insert.
NInt valueLength	[in] Length of the char array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.50 NStringBuilderInsertDouble Function

Inserts a string representation of the specified double integer into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertDouble(NStringBuilder * pStringBuilder, NInt index, NDouble value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NDouble value	[in] NDouble [1005] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.51 NStringBuilderInsertInt16 Function

Inserts a string representation of the specified short integer into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertInt16(NStringBuilder * pStringBuilder, NInt index, NShort value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NShort value	[in] NShort [1008] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.1.52 NStringBuilderInsertInt32 Function

Inserts a string representation of the specified integer into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertInt32(NStringBuilder * pStringBuilder, NInt index, NInt value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NInt value	[in] NInt [1006] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.53 NStringBuilderInsertInt64 Function

Inserts a string representation of the specified long integer into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertInt64(NStringBuilder * pStringBuilder, NInt index, NLong value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NLong value	[in] NLong [1007] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.54 NStringBuilderInsertN Function

Inserts a string into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertN(NStringBuilder * pStringBuilder, NInt index, HNString hValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
HNString hValue	[in] Handle to the string to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.55 NStringBuilderInsertObject Function

Inserts a string representation of the specified object into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertObject(NStringBuilder * pStringBuilder, NInt index,
HNObject hObject);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
HNObject hObject	[in] Handle to the object to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.56 NStringBuilderInsertPointer Function

Inserts a string representation of the specified pointer into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertPointer(NStringBuilder * pStringBuilder, NInt index,
const void * value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
const void * value	[in] The pointer to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.53.1.57 NStringBuilderInsertResult Function

Inserts a string representation of a NResult [1007] value into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertResult(NStringBuilder * pStringBuilder, NInt index,
NResult value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NResult value	[in] The NResult [1007] value to be inserted.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.58 NStringBuilderInsertSByte Function

Inserts a string representation of the specified byte integer into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertSByte(NStringBuilder * pStringBuilder, NInt index, NSByte
value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NSByte value	[in] NSByte [1008] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.59 NStringBuilderInsertSingle Function

Inserts a string representation of the specified floating point number into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertSingle(NStringBuilder * pStringBuilder, NInt index,
NFloat value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFloat value	[in] NFloat [1005] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.60 NStringBuilderInsertSizeType Function

Inserts a string representation of the specified SizeType into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertSizeType(NStringBuilder * pStringBuilder, NInt index,
NSizeType value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NSizeType value	[in] NSizeType [1008] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.61 NStringBuilderInsertSSizeType Function

Inserts a string representation of the specified SSizeType into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertSSizeType(NStringBuilder * pStringBuilder, NInt index,
NSSizeType value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NSSizeType value	[in] NSSizeType [1009] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.62 NStringBuilderInsertUInt16 Function

Inserts a string representation of the specified unsigned short integer into the specified NStringBuilder [769] at the specified character position.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStringBuilderInsertUInt16(NStringBuilder * pStringBuilder, NInt index,
NUShort value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NUShort value	[in] NUShort [1010] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.63 NStringBuilderInsertUInt32 Function

Inserts a string representation of the specified unsigned integer into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertUInt32(NStringBuilder * pStringBuilder, NInt index, NUInt
value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NUInt value	[in] NUInt [1009] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.64 NStringBuilderInsertUInt64 Function

Inserts a string representation of the specified long integer into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertUInt64(NStringBuilder * pStringBuilder, NInt index,
NULong value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NULong value	[in] NULong [1010] to insert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.1.65 NStringBuilderInsertValue Function

Inserts a string representation of a value of specified type into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertValue(NStringBuilder * pStringBuilder, NInt index, HNType hType, const void * pValue, NSizeType valueSize);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
HNType hType	[in] Handle to the type of value to be appended.
const void * pValue	[in] Pointer to the value to be appended.
NSizeType valueSize	[in] Size of the value to be appended.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.66 NStringBuilderInsertValueP Function

Inserts a string representation of the specified value into the specified NStringBuilder [769] at the specified character position.

C++

```
NResult N_API NStringBuilderInsertValueP(NStringBuilder * pStringBuilder, NInt index, NTypeOfProc pTypeOf, const void * pValue, NSizeType valueSize);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] The position in the string builder where insertion begins.
NTypeOfProc pTypeOf	[in] Type of the value.
const void * pValue	[in] Pointer to the value to insert.
NSizeType valueSize	[in] Size of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.67 NStringBuilderRemove Function

Removes the specified range of characters from the specified NStringBuilder [769].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStringBuilderRemove(NStringBuilder * pStringBuilder, NInt startIndex, NInt count);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt startIndex	[in] The zero-based position in this instance where removal begins.
NInt count	[in] The number of characters to remove.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.68 NStringBuilderReplace Function

Replaces all occurrences of a specified character in the specified NStringBuilder [769] with another specified character.

C++

```
NResult N_API NStringBuilderReplace(NStringBuilder * pStringBuilder, NChar oldValue, NChar newValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NChar oldValue	[in] The character to replace.
NChar newValue	[in] The character that replaces *oldValue*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.69 NStringBuilderReplaceInRange Function

Replaces, within a substring of the specified NStringBuilder [769], all occurrences of a specified character with another specified character.

C++

```
NResult N_API NStringBuilderReplaceInRange(NStringBuilder * pStringBuilder, NChar oldValue, NChar newValue, NInt startIndex, NInt count);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NChar oldValue	[in] The character to replace.
NChar newValue	[in] The character that replaces *oldValue*.
NInt startIndex	[in] The position in the NStringBuilder [769] where the substring begins.
NInt count	[in] The length of the substring.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.70 NStringBuilderReplaceStringInRangeN Function

Replaces, within a substring of the specified NStringBuilder [769], all occurrences of a specified string with another specified string.

C++

```
NResult N_API NStringBuilderReplaceStringInRangeN(NStringBuilder * pStringBuilder, HNString hOldValue, HNString hNewValue, NInt startIndex, NInt count);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
HNString hOldValue	[in] Handle to the string to replace.
HNString hNewValue	[in] The string that replaces *hOldValue*.
NInt startIndex	[in] The position in the NStringBuilder [769] where the substring begins.
NInt count	[in] The length of the substring.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.71 NStringBuilderReplaceStringN Function

Replaces all occurrences of a specified string in the specified NStringBuilder [769] with another specified character.

C++

```
NResult N_API NStringBuilderReplaceStringN(NStringBuilder * pStringBuilder, HNString hOldValue, HNString hNewValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
HNString hOldValue	[in] Handle to the string to replace.
HNString hNewValue	[in] The string that replaces *hOldValue*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.72 NStringBuilderReplaceStrOrChars Function

Replaces all occurrences of a specified string or char array in the specified NStringBuilder [769] with another specified string or char array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStringBuilderReplaceStrOrChars(NStringBuilder * pStringBuilder, const NChar
* arOldValue, NInt oldValueLength, const NChar * arNewValue, NInt newValueLength);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
const NChar * arOldValue	[in] Pointer to the char array to replace.
NInt oldValueLength	[in] *arOldValue* array length.
const NChar * arNewValue	[in] Pointer to the char array that replaces *arOldValue.*
NInt newValueLength	[in] *arNewValue* array length.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.73 NStringBuilderReplaceStrOrCharsInRange Function

Replaces, within a substring of the specified NStringBuilder [769], all occurrences of a specified string or char array with another specified string or char array.

C++

```
NResult N_API NStringBuilderReplaceStrOrCharsInRange(NStringBuilder * pStringBuilder, const
NChar * arOldValue, NInt oldValueLength, const NChar * arNewValue, NInt newValueLength,
NInt startIndex, NInt count);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
const NChar * arOldValue	[in] Pointer to the char array to replace.
NInt oldValueLength	[in] *arOldValue* array length.
const NChar * arNewValue	[in] Pointer to the char array that replaces *arOldValue.*
NInt newValueLength	[in] *arNewValue* array length.
NInt startIndex	[in] The position in the NStringBuilder [769] where the substring begins.
NInt count	[in] The length of the substring.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.74 NStringBuilderSetCapacity Function

Sets the capacity of the specified NStringBuilder [769].

C++

```
NResult N_API NStringBuilderSetCapacity(NStringBuilder * pStringBuilder, NInt value);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt value	[in] The new capacity value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.75 NStringBuilderSetChar Function

Sets the value of the character at the specified position in the specified NStringBuilder [769].

C++

```
NResult N_API NStringBuilderSetChar(NStringBuilder * pStringBuilder, NInt index, NChar value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt index	[in] Index of the character.
NChar value	[in] New character value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.76 NStringBuilderSetLength Function

Sets the length of the specified NStringBuilder [769].

C++

```
NResult N_API NStringBuilderSetLength(NStringBuilder * pStringBuilder, NInt value);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt value	[in] The new length value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.77 NStringBuilderToString Function

Converts the specified NStringBuilder [769] to a string.

C++

```
NResult N_API NStringBuilderToString(NStringBuilder * pStringBuilder, HNString * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
HNString * phValue	[out] Pointer to a handle to the string representing the NStringBuilder [769].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.1.78 NStringBuilderToStringRange Function

Converts the specified NStringBuilder [769] to a string range.

C++

```
NResult N_API NStringBuilderToStringRange(NStringBuilder * pStringBuilder, NInt startIndex,
NInt length, HNString * phValue);
```

Parameters

Parameters	Description
NStringBuilder * pStringBuilder	[in/out] Handle to the NStringBuilder [769].
NInt startIndex	[in] Starting index of the string range.
NInt length	[in] Length of the string range.
HNString * phValue	[out] Pointer to a handle to the string representing the NStringBuilder [769].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.2 Macros

7.1.2.53.2.1 N_STRING_BUILDER_SIZE Macro

Defines the NStringBuilder [769] size.

C++

```
#define N_STRING_BUILDER_SIZE 20
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.2 NStringBuilderAppend Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppend function.

C++

```
#define NStringBuilderAppend N_FUNC_AW(NStringBuilderAppend)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.2.3 NStringBuilderAppendBoolean Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendBoolean function.

C++

```
#define NStringBuilderAppendBoolean N_FUNC_AW(NStringBuilderAppendBoolean)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.4 NStringBuilderAppendByte Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendByte function.

C++

```
#define NStringBuilderAppendByte N_FUNC_AW(NStringBuilderAppendByte)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.5 NStringBuilderAppendChar Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendChar function.

C++

```
#define NStringBuilderAppendChar N_FUNC_AW(NStringBuilderAppendChar)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.6 NStringBuilderAppendChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendChars function.

C++

```
#define NStringBuilderAppendChars N_FUNC_AW(NStringBuilderAppendChars)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.7 NStringBuilderAppendDouble Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendDouble function.

C++

```
#define NStringBuilderAppendDouble N_FUNC_AW(NStringBuilderAppendDouble)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.8 NStringBuilderAppendEmptyLine Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendEmptyLine function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NStringBuilderAppendEmptyLine N_FUNC_AW(NStringBuilderAppendEmptyLine)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.9 NStringBuilderAppendFormat Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendFormat function.

C++

```
#define NStringBuilderAppendFormat N_FUNC_AW(NStringBuilderAppendFormat)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.10 NStringBuilderAppendFormatN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendFormatN function.

C++

```
#define NStringBuilderAppendFormatN N_FUNC_AW(NStringBuilderAppendFormatN)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.11 NStringBuilderAppendFormatVA Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendFormatVA function.

C++

```
#define NStringBuilderAppendFormatVA N_FUNC_AW(NStringBuilderAppendFormatVA)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.12 NStringBuilderAppendFormatVAN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendFormatVAN function.

C++

```
#define NStringBuilderAppendFormatVAN N_FUNC_AW(NStringBuilderAppendFormatVAN)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.13 NStringBuilderAppendInt16 Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendInt16 function.

C++

```
#define NStringBuilderAppendInt16 N_FUNC_AW(NStringBuilderAppendInt16)
```

Module

NStringBuilder Unit [769]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.53.2.14 NStringBuilderAppendInt32 Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendInt32 function.

C++

```
#define NStringBuilderAppendInt32 N_FUNC_AW(NStringBuilderAppendInt32)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.15 NStringBuilderAppendInt64 Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendInt64 function.

C++

```
#define NStringBuilderAppendInt64 N_FUNC_AW(NStringBuilderAppendInt64)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.16 NStringBuilderAppendLine Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendLine function.

C++

```
#define NStringBuilderAppendLine N_FUNC_AW(NStringBuilderAppendLine)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.17 NStringBuilderAppendLineN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendLineN function.

C++

```
#define NStringBuilderAppendLineN N_FUNC_AW(NStringBuilderAppendLineN)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.18 NStringBuilderAppendN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendN function.

C++

```
#define NStringBuilderAppendN N_FUNC_AW(NStringBuilderAppendN)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.19 NStringBuilderAppendObject Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendObject function.

C++

```
#define NStringBuilderAppendObject N_FUNC_AW(NStringBuilderAppendObject)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.2.20 NStringBuilderAppendPointer Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendPointer function.

C++

```
#define NStringBuilderAppendPointer N_FUNC_AW(NStringBuilderAppendPointer)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.21 NStringBuilderAppendResult Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendResult function.

C++

```
#define NStringBuilderAppendResult N_FUNC_AW(NStringBuilderAppendResult)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.22 NStringBuilderAppendSByte Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendSByte function.

C++

```
#define NStringBuilderAppendSByte N_FUNC_AW(NStringBuilderAppendSByte)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.23 NStringBuilderAppendSingle Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendSingle function.

C++

```
#define NStringBuilderAppendSingle N_FUNC_AW(NStringBuilderAppendSingle)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.24 NStringBuilderAppendSizeType Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendSizeType function.

C++

```
#define NStringBuilderAppendSizeType N_FUNC_AW(NStringBuilderAppendSizeType)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.25 NStringBuilderAppendSSizeType Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendSSizeType function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NStringBuilderAppendSSizeType N_FUNC_AW(NStringBuilderAppendSSizeType)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.26 NStringBuilderAppendUInt16 Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendUInt16 function.

C++

```
#define NStringBuilderAppendUInt16 N_FUNC_AW(NStringBuilderAppendUInt16)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.27 NStringBuilderAppendUInt32 Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendUInt32 function.

C++

```
#define NStringBuilderAppendUInt32 N_FUNC_AW(NStringBuilderAppendUInt32)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.28 NStringBuilderAppendUInt64 Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendUInt64 function.

C++

```
#define NStringBuilderAppendUInt64 N_FUNC_AW(NStringBuilderAppendUInt64)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.29 NStringBuilderAppendValue Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendValue function.

C++

```
#define NStringBuilderAppendValue N_FUNC_AW(NStringBuilderAppendValue)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.30 NStringBuilderAppendValueP Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderAppendValueP function.

C++

```
#define NStringBuilderAppendValueP N_FUNC_AW(NStringBuilderAppendValueP)
```

Module

NStringBuilder Unit [769]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.53.2.31 NStringBuilderClear Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderClear function.

C++

```
#define NStringBuilderClear N_FUNC_AW(NStringBuilderClear)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.32 NStringBuilderCopyTo Macro

Copies the characters from a specified segment of the NStringBuilder [769] to a specified segment of a destination Char array.

C++

```
#define NStringBuilderCopyTo(pStringBuilder, sourceIndex, arValue, valueLength, count)
NStringBuilderCopyToStrOrChars(pStringBuilder, sourceIndex, arValue, valueLength, NFalse,
count)
```

Parameters

Parameters	Description
pStringBuilder	[in/out] Handle to the NStringBuilder [769].
sourceIndex	[in] Index in the NStringBuilder [769] at which the copying will start.
arValue	[in] Destination char array.
valueLength	[in] Length of the char array.
count	[in] The number of characters to be copied.

Returns

Module

NStringBuilder Unit [769]

7.1.2.53.2.33 NStringBuilderCopyToString Macro

Copies the characters from a specified segment of the NStringBuilder [769] to a specified segment of a destination string.

C++

```
#define NStringBuilderCopyToString(pStringBuilder, sourceIndex, szValue, valueSize, count)
NStringBuilderCopyToStrOrChars(pStringBuilder, sourceIndex, szValue, valueSize, NTrue,
count)
```

Parameters

Parameters	Description
pStringBuilder	[in/out] Handle to the NStringBuilder [769].
sourceIndex	[in] Index in the NStringBuilder [769] at which the copying will start.
szValue	[in] Destination string.
valueSize	[in] Size of the string.
count	[in] The number of characters to be copied.

Module

NStringBuilder Unit [769]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.53.2.34 NStringBuilderCopyToStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringCopyToStrOrChars [693] function.

C++

```
#define NStringBuilderCopyToStrOrChars N_FUNC_AW(NStringBuilderCopyToStrOrChars)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.35 NStringBuilderDetachCharArray Macro

Detaches a char array from the NStringBuilder [769].

C++

```
#define NStringBuilderDetachCharArray(pStringBuilder, parValue, pValueLength)
NStringBuilderDetachStrOrChars(pStringBuilder, NFalse, parValue, pValueLength)
```

Parameters

Parameters	Description
pStringBuilder	[in/out] Handle to the NStringBuilder [769].
nullTerminate	[in] If NTrue [1064], null terminated string will be returned.
parValue	[out] Pointer to a detached char array.
pValueLength	[out] Detached char array length.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.2.36 NStringBuilderDetachString Macro

Detaches a string from the NStringBuilder [769].

C++

```
#define NStringBuilderDetachString(pStringBuilder, pszValue, pValueLength)
NStringBuilderDetachStrOrChars(pStringBuilder, NTrue, pszValue, pValueLength)
```

Parameters

Parameters	Description
pStringBuilder	[in/out] Handle to the NStringBuilder [769].
pszValue	[out] Pointer to the detached string.
pValueLength	[out] Length of the detached string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.2.37 NStringBuilderDetachStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderDetachStrOrChars function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NStringBuilderDetachStrOrChars N_FUNC_AW(NStringBuilderDetachStrOrChars)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.38 NStringBuilderDispose Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderDispose function.

C++

```
#define NStringBuilderDispose N_FUNC_AW(NStringBuilderDispose)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.39 NStringBuilderGetBuffer Macro

Gets the string buffer of the specified NStringBuilder [769].

C++

```
#define NStringBuilderGetBuffer(pStringBuilder, pValueLength, pszValue)
NStringBuilderGetStrOrCharsBuffer(pStringBuilder, NTrue, pValueLength, pszValue)
```

Parameters

Parameters	Description
pStringBuilder	[in/out] Handle to the NStringBuilder [769].
pValueLength	[out] Pointer to the length of the char array.
pszValue	[out] Pointer to the string buffer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.2.40 NStringBuilderGetCapacity Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderGetCapacity function.

C++

```
#define NStringBuilderGetCapacity N_FUNC_AW(NStringBuilderGetCapacity)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.41 NStringBuilderGetChar Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderGetChar function.

C++

```
#define NStringBuilderGetChar N_FUNC_AW(NStringBuilderGetChar)
```

Module

NStringBuilder Unit [769]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.53.2.42 NStringBuilderGetCharsBuffer Macro

Gets the character array buffer of the specified NStringBuilder [769].

C++

```
#define NStringBuilderGetCharsBuffer(pStringBuilder, pValueLength, parValue)
NStringBuilderGetStrOrCharsBuffer(pStringBuilder, NFalse, pValueLength, parValue)
```

Parameters

Parameters	Description
pStringBuilder	[in/out] Handle to the NStringBuilder [769].
pValueLength	[out] Pointer to the length of the char array.
parValue	[out] Pointer to the char array buffer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.2.43 NStringBuilderGetLength Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderGetLength function.

C++

```
#define NStringBuilderGetLength N_FUNC_AW(NStringBuilderGetLength)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.44 NStringBuilderGetStringWrapper Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderGetStringWrapper function.

C++

```
#define NStringBuilderGetStringWrapper N_FUNC_AW(NStringBuilderGetStringWrapper)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.45 NStringBuilderGetStrOrCharsBuffer Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderGetStrOrCharsBuffer function.

C++

```
#define NStringBuilderGetStrOrCharsBuffer N_FUNC_AW(NStringBuilderGetStrOrCharsBuffer)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.46 NStringBuilderInit Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInit function.

C++

```
#define NStringBuilderInit N_FUNC_AW(NStringBuilderInit)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.2.47 NStringBuilderInitEx Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInitEx function.

C++

```
#define NStringBuilderInitEx N_FUNC_AW(NStringBuilderInitEx)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.48 NStringBuilderInitWithCapacity Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInitWithCapacity function.

C++

```
#define NStringBuilderInitWithCapacity N_FUNC_AW(NStringBuilderInitWithCapacity)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.49 NStringBuilderInitWithChars Macro

Creates a new NStringBuilder [769] using the specified char array.

C++

```
#define NStringBuilderInitWithChars(pStringBuilder, arValue, valueLength)
NStringBuilderInitWithStrOrChars(pStringBuilder, arValue, valueLength)
```

Parameters

Parameters	Description
arValue	[in] Pointer to the char array.
valueLength	[in] Length of the array.
phStringBuilder	[out] Pointer to a handle to the new NStringBuilder [769].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.2.50 NStringBuilderInitWithString Macro

Creates a new NStringBuilder [769] using the specified string.

C++

```
#define NStringBuilderInitWithString(pStringBuilder, szValue)
NStringBuilderInitWithStrOrChars(pStringBuilder, szValue, -1)
```

Parameters

Parameters	Description
szValue	[in] Pointer to the string.
phStringBuilder	[out] Pointer to a handle to the new NStringBuilder [769].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.2.51 NStringBuilderInitWithStringN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInitWithStringN function.

C++

```
#define NStringBuilderInitWithStringN N_FUNC_AW(NStringBuilderInitWithStringN)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.52 NStringBuilderInitWithStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInitWithStrOrChars function.

C++

```
#define NStringBuilderInitWithStrOrChars N_FUNC_AW(NStringBuilderInitWithStrOrChars)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.53 NStringBuilderInsert Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsert function.

C++

```
#define NStringBuilderInsert N_FUNC_AW(NStringBuilderInsert)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.54 NStringBuilderInsertBoolean Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertBoolean function.

C++

```
#define NStringBuilderInsertBoolean N_FUNC_AW(NStringBuilderInsertBoolean)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.55 NStringBuilderInsertByte Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertByte function.

C++

```
#define NStringBuilderInsertByte N_FUNC_AW(NStringBuilderInsertByte)
```

Module

NStringBuilder Unit [769]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.53.2.56 NStringBuilderInsertChar Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertChar function.

C++

```
#define NStringBuilderInsertChar N_FUNC_AW(NStringBuilderInsertChar)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.57 NStringBuilderInsertChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertChars function.

C++

```
#define NStringBuilderInsertChars N_FUNC_AW(NStringBuilderInsertChars)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.58 NStringBuilderInsertDouble Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertDouble function.

C++

```
#define NStringBuilderInsertDouble N_FUNC_AW(NStringBuilderInsertDouble)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.59 NStringBuilderInsertInt16 Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertInsertInt16 function.

C++

```
#define NStringBuilderInsertInt16 N_FUNC_AW(NStringBuilderInsertInt16)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.60 NStringBuilderInsertInt32 Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertInt32 function.

C++

```
#define NStringBuilderInsertInt32 N_FUNC_AW(NStringBuilderInsertInt32)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.61 NStringBuilderInsertInt64 Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertInt64 function.

C++

```
#define NStringBuilderInsertInt64 N_FUNC_AW(NStringBuilderInsertInt64)
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NStringBuilder Unit [769]

7.1.2.53.2.62 NStringBuilderInsertN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertN function.

C++

```
#define NStringBuilderInsertN N_FUNC_AW(NStringBuilderInsertN)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.63 NStringBuilderInsertObject Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertObject function.

C++

```
#define NStringBuilderInsertObject N_FUNC_AW(NStringBuilderInsertObject)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.64 NStringBuilderInsertPointer Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertPointer function.

C++

```
#define NStringBuilderInsertPointer N_FUNC_AW(NStringBuilderInsertPointer)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.65 NStringBuilderInsertResult Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertResult function.

C++

```
#define NStringBuilderInsertResult N_FUNC_AW(NStringBuilderInsertResult)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.66 NStringBuilderInsertSByte Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertSByte function.

C++

```
#define NStringBuilderInsertSByte N_FUNC_AW(NStringBuilderInsertSByte)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.67 NStringBuilderInsertSingle Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertSingle function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NStringBuilderInsertSingle N_FUNC_AW(NStringBuilderInsertSingle)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.68 NStringBuilderInsertSizeType Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertSizeType function.

C++

```
#define NStringBuilderInsertSizeType N_FUNC_AW(NStringBuilderInsertSizeType)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.69 NStringBuilderInsertSSizeType Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertSSizeType function.

C++

```
#define NStringBuilderInsertSSizeType N_FUNC_AW(NStringBuilderInsertSSizeType)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.70 NStringBuilderInsertUInt16 Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertUInt16 function.

C++

```
#define NStringBuilderInsertUInt16 N_FUNC_AW(NStringBuilderInsertUInt16)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.71 NStringBuilderInsertUInt32 Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertUInt32 function.

C++

```
#define NStringBuilderInsertUInt32 N_FUNC_AW(NStringBuilderInsertUInt32)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.72 NStringBuilderInsertUInt64 Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderInsertUInt64 function.

C++

```
#define NStringBuilderInsertUInt64 N_FUNC_AW(NStringBuilderInsertUInt64)
```

Module

NStringBuilder Unit [769]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.53.2.73 NStringBuilderRemove Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderRemove function.

C++

```
#define NStringBuilderRemove N_FUNC_AW(NStringBuilderRemove)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.74 NStringBuilderReplace Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderReplace function.

C++

```
#define NStringBuilderReplace N_FUNC_AW(NStringBuilderReplace)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.75 NStringBuilderReplaceChars Macro

Replaces all occurrences of a specified char array in the specified NStringBuilder [769] with another specified char array.

C++

```
#define NStringBuilderReplaceChars(pStringBuilder, arOldValue, oldValueLength, arNewValue,
newValueLength) NStringBuilderReplaceStrOrChars(pStringBuilder, arOldValue, oldValueLength,
arNewValue, newValueLength)
```

Parameters

Parameters	Description
pStringBuilder	[in/out] Handle to the NStringBuilder [769].
arOldValue	[in] Pointer to the char array to replace.
oldValueLength	[in] *arOldValue* array length.
arNewValue	[in] Pointer to the char array that replaces *arOldValue.*
newValueLength	[in] *arNewValue* array length.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.2.76 NStringBuilderReplaceCharsInRange Macro

Replaces, within a substring of the specified NStringBuilder [769], all occurrences of a specified char array with another specified char array.

C++

```
#define NStringBuilderReplaceCharsInRange(pStringBuilder, arOldValue, oldValueLength,
arNewValue, newValueLength, startIndex, count)
NStringBuilderReplaceStrOrCharsInRange(pStringBuilder, arOldValue, oldValueLength,
arNewValue, newValueLength, startIndex, count)
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
pStringBuilder	[in/out] Handle to the NStringBuilder [769].
arOldValue	[in] Pointer to the char array to replace.
oldValueLength	[in] *arOldValue* array length.
arNewValue	[in] Pointer to the char array that replaces *arOldValue.*
newValueLength	[in] *arNewValue* array length.
startIndex	[in] The position in the NStringBuilder [769] where the substring begins.
count	[in] The length of the substring.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.2.77 NStringBuilderReplaceInRange Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderReplaceInRange function.

C++

```
#define NStringBuilderReplaceInRange N_FUNC_AW(NStringBuilderReplaceInRange)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.78 NStringBuilderReplaceString Macro

Replaces all occurrences of a specified string in the specified NStringBuilder [769] with another specified character.

C++

```
#define NStringBuilderReplaceString(pStringBuilder, szOldValue, szNewValue)
NStringBuilderReplaceStrOrChars(pStringBuilder, szOldValue, -1, szNewValue, -1)
```

Parameters

Parameters	Description
pStringBuilder	[in/out] Pointer to the NStringBuilder [769].
szOldValue	[in] Pointer to the string to replace.
szNewValue	[in] The string that replaces *szOldValue*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.2.79 NStringBuilderReplaceStringInRange Macro

Replaces, within a substring of the specified NStringBuilder [769], all occurrences of a specified string with another specified string.

C++

```
#define NStringBuilderReplaceStringInRange(pStringBuilder, szOldValue, szNewValue,
startIndex, count) NStringBuilderReplaceStrOrCharsInRange(pStringBuilder, szOldValue, -1,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
szNewValue, -1, startIndex, count)
```

Parameters

Parameters	Description
szStringBuilder	[in] Pointer to the NStringBuilder [769].
szOldValue	[in] Pointer to the string to replace.
hNewValue	[in] The string that replaces *szOldValue*.
startIndex	[in] The position in the NStringBuilder [769] where the substring begins.
count	[in] The length of the substring.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringBuilder Unit [769]

7.1.2.53.2.80 NStringBuilderReplaceStringN Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderReplaceStringN function.

C++

```
#define NStringBuilderReplaceStringN N_FUNC_AW(NStringBuilderReplaceStringN)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.81 NStringBuilderReplaceStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderReplaceStrOrChars function.

C++

```
#define NStringBuilderReplaceStrOrChars N_FUNC_AW(NStringBuilderReplaceStrOrChars)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.82 NStringBuilderReplaceStrOrCharsInRange Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderReplaceStrOrCharsInRange function.

C++

```
#define NStringBuilderReplaceStrOrCharsInRange
N_FUNC_AW(NStringBuilderReplaceStrOrCharsInRange)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.83 NStringBuilderSetCapacity Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderSetCapacity function.

C++

```
#define NStringBuilderSetCapacity N_FUNC_AW(NStringBuilderSetCapacity)
```

Module

NStringBuilder Unit [769]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.53.2.84 NStringBuilderSetChar Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderSetChar function.

C++

```
#define NStringBuilderSetChar N_FUNC_AW(NStringBuilderSetChar)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.85 NStringBuilderSetLength Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderSetLength function.

C++

```
#define NStringBuilderSetLength N_FUNC_AW(NStringBuilderSetLength)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.86 NStringBuilderToString Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderToString function.

C++

```
#define NStringBuilderToString N_FUNC_AW(NStringBuilderToString)
```

Module

NStringBuilder Unit [769]

7.1.2.53.2.87 NStringBuilderToStringRange Macro

Picks a Unicode (W) or ASCII (A) version of the NStringBuilderToStringRange function.

C++

```
#define NStringBuilderToStringRange N_FUNC_AW(NStringBuilderToStringRange)
```

Module

NStringBuilder Unit [769]

7.1.2.54 NStringReader Unit

Implements a TextReader that reads from a string.

Functions

	Name	Description
	NStringReaderCreateFromStrOrChars [823]	Creates a new NStringReader that reads from the specified string or char array.
	NStringReaderCreateN [823]	Creates a new NStringReader that reads from the specified NString [678].

Macros

Name	Description
NStringReaderCreate [823]	Creates a new NStringReader that reads from the specified string.
NStringReaderCreateFromChars [824]	Creates a new NStringReader that reads from the specified char array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringReaderCreateFromStrOrChars [824]	Picks a Unicode (W) or ASCII (A) version of the NStringReaderCreateFromStrOrChars function.

7.1.2.54.1 Functions

7.1.2.54.1.1 NStringReaderCreateFromStrOrChars Function

Creates a new NStringReader [822] that reads from the specified string or char array.

C++

```
NResult N_API NStringReaderCreateFromStrOrChars(const NChar * arValue, NInt valueLength,
HNStringReader * phReader);
```

Parameters

Parameters	Description
const NChar * arValue	[in] Pointer to the char array or string.
NInt valueLength	[in] Length of the char array.
HNStringReader * phReader	[out] Pointer to a handle to the new NStringReader [822].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringReader Unit [822]

7.1.2.54.1.2 NStringReaderCreateN Function

Creates a new NStringReader [822] that reads from the specified NString [678].

C++

```
NResult N_API NStringReaderCreateN(HNString hValue, HNStringReader * phReader);
```

Parameters

Parameters	Description
HNString hValue	[in] Handle to the NString [678].
HNStringReader * phReader	[out] Pointer to a handle to the new NStringReader [822].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringReader Unit [822]

7.1.2.54.2 Macros

7.1.2.54.2.1 NStringReaderCreate Macro

Creates a new NStringReader [822] that reads from the specified string.

C++

```
#define NStringReaderCreate(szValue, phReader) NStringReaderCreateFromStrOrChars(szValue,
-1, phReader)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
szValue	[in] Pointer to the string.
phReader	[out] Pointer to a handle to the new NStringReader [822].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringReader Unit [822]

7.1.2.54.2.2 NStringReaderCreateFromChars Macro

Creates a new NStringReader [822] that reads from the specified char array.

C++

```
#define NStringReaderCreateFromChars(arValue, valueLength, phReader)
NStringReaderCreateFromStrOrChars(arValue, valueLength, phReader)
```

Parameters

Parameters	Description
arValue	[in] Pointer to the char array or string.
valueLength	[in] Length of the char array.
phReader	[out] Pointer to a handle to the new NStringReader [822].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringReader Unit [822]

7.1.2.54.2.3 NStringReaderCreateFromStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NStringReaderCreateFromStrOrChars function.

C++

```
#define NStringReaderCreateFromStrOrChars N_FUNC_AW(NStringReaderCreateFromStrOrChars)
```

Module

NStringReader Unit [822]

7.1.2.55 NStringWriter Unit

Implements a TextWriter for writing information to a string. The information is stored in an underlying StringBuilder.

Files

Name	Description
NStringWriter.h [826]	Header file for the NStringWriter module.

Functions

	Name	Description
	NStringWriterCreate [825]	Creates a new NStringWriter.
	NStringWriterCreateEx [825]	Creates a new NStringWriter with specified capacity and growth delta.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NStringWriterCreateWithCapacity [825]	Creates a new NStringWriter with specified capacity.
	NStringWriterDetachString [826]	Detaches the specified string from the specified string writer.

7.1.2.55.1 Functions

7.1.2.55.1.1 NStringWriterCreate Function

Creates a new NStringWriter [824].

C++

```
NResult N_API NStringWriterCreate(HNStringWriter * phWriter);
```

Parameters

Parameters	Description
HNStringWriter * phWriter	[out] Pointer to a handle to the new NStringWriter [824].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

A new NStringBuilder [769] is automatically created and associated with the new NStringWriter [824].

Module

NStringWriter Unit [824]

7.1.2.55.1.2 NStringWriterCreateEx Function

Creates a new NStringWriter [824] with specified capacity and growth delta.

C++

```
NResult N_API NStringWriterCreateEx(NInt capacity, NInt maxCapacity, NInt growthDelta,
HNStringWriter * phWriter);
```

Parameters

Parameters	Description
NInt capacity	[in] Capacity value of the new string writer.
NInt maxCapacity	[in] Maximum capacity value of the new string writer.
NInt growthDelta	[in] Growth delta of the new string writer.
HNStringWriter * phWriter	[out] Pointer to a handle to the new NStringWriter [824].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

A new NStringBuilder [769] is automatically created and associated with the new NStringWriter [824].

Module

NStringWriter Unit [824]

7.1.2.55.1.3 NStringWriterCreateWithCapacity Function

Creates a new NStringWriter [824] with specified capacity.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NStringWriterCreateWithCapacity(NInt capacity, HNStringWriter * phWriter);
```

Parameters

Parameters	Description
NInt capacity	[in] Capacity value of the new string writer.
HNStringWriter * phWriter	[out] Pointer to a handle to the new NStringWriter [824].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

A new NStringBuilder [769] is automatically created and associated with the new NStringWriter [824].

Module

NStringWriter Unit [824]

7.1.2.55.1.4 NStringWriterDetachString Function

Detaches the specified string from the specified string writer.

C++

```
NResult N_API NStringWriterDetachString(HNStringWriter hWriter, HNString * phValue);
```

Parameters

Parameters	Description
HNStringWriter hWriter	[in] Handle to the string writer.
HNString * phValue	[out] Pointer to a handle to the string to be detached.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NStringWriter Unit [824]

7.1.2.55.2 Files

7.1.2.55.2.1 NStringWriter.h

Header file for the NStringWriter [824] module.

Functions

	Name	Description
	NStringWriterCreate [825]	Creates a new NStringWriter [824].
	NStringWriterCreateEx [825]	Creates a new NStringWriter [824] with specified capacity and growth delta.
	NStringWriterCreateWithCapacity [825]	Creates a new NStringWriter [824] with specified capacity.
	NStringWriterDetachString [826]	Detaches the specified string from the specified string writer.

Module

NStringWriter Unit [824]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.56 NSyncEvent Unit

Cross platform synchronization primitive. It allows for threads to wait for event without consuming CPU. The other thread(s) would notify waiting thread(s) of the event by setting it to signalled state. See the NSyncEventCreate [827] function for more details.

Functions

	Name	Description
	NSyncEventCreate [827]	Creates and initializes a sync event object.
	NSyncEventCreateFromOSHandle [828]	Creates a new NSyncEvent for the specified sync event handle.
	NSyncEventGetOSHandle [828]	Gets OS handle for the specified sync event.
	NSyncEventSet [828]	Post event to the waiting thread(s), i.e. make the event object signalled.

Macros

Name	Description
NSyncEventReset [829]	Resets the specified sync event.

7.1.2.56.1 Functions

7.1.2.56.1.1 NSyncEventCreate Function

Creates and initializes a sync event object.

C++

```
NResult N_API NSyncEventCreate(NBool manualReset, NBool initialState, HNSyncEvent *
phSyncEvent);
```

Parameters

Parameters	Description
NBool manualReset	[in] Should the event be of manual-reset type. Auto-reset event will be created if parameter is NFalse [1041].
NBool initialState	[in] Initial state: NTrue [1064] - signalled, NFalse [1041] - nonsignalled.
HNSyncEvent * phSyncEvent	[out] Pointer to the variable that will receive the handle of the created object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

The NSyncEvent [827]'s state can be characterized by the flag, which can be set (NSyncEventSet [828]), cleared (NSyncEventReset [829]) or waited to be set (NWaitObjectWaitFor [1123]/NWaitObjectTimedWaitFor). The event will be delivered to expecting thread even if it is set to signalled state before expecting thread starts to wait.

Two kinds of NSyncEvent [827] are possible: auto-reset and manual-reset. Auto-reset event is reset to nonsignalled state automatically when waiting thread gets the event. As a result the event is always delivered to only single expecting thread. If other waiting threads are present they would remain waiting until event is signalled again and delivered to single waiting thread.

Manual-reset event must to be manually reset to nonsignalled state. As result all the waiting threads will receive the event and continue execution. Also subsequent wait operations will complete immediately until the event is cleared with NSyncEventReset [829] function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Notes

Note that event object doesn't buffer the events, ie. the set operation on already signalled event has no effect. Check NSemaphore [654] for alternative synchronization object.

Module

NSyncEvent Unit [827]

7.1.2.56.1.2 NSyncEventCreateFromOSHandle Function

Creates a new NSyncEvent [827] for the specified sync event handle.

C++

```
NResult N_API NSyncEventCreateFromOSHandle(NHandle handle, NBool ownsHandle, HNSyncEvent *
phSyncEvent);
```

Parameters

Parameters	Description
NHandle handle	[in] A sync event handle for the sync event that the current NSyncEvent [827] will encapsulate.
NBool ownsHandle	[in] NTrue [1064] if the sync event handle will be owned by this NSyncEvent [827]; otherwise, NFalse [1041].
HNSyncEvent * phSyncEvent	[out] A pointer to the created NSyncEvent [827].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NSyncEventGetOSHandle [828]

Module

NSyncEvent Unit [827]

7.1.2.56.1.3 NSyncEventGetOSHandle Function

Gets OS handle for the specified sync event.

C++

```
NResult N_API NSyncEventGetOSHandle(HNSyncEvent hSyncEvent, NHandle * phValue);
```

Parameters

Parameters	Description
HNSyncEvent hSyncEvent	[in] A handle to NSyncEvent [827].
NHandle * phValue	[out] OS handle to a sync event.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSyncEvent Unit [827]

7.1.2.56.1.4 NSyncEventSet Function

Post event to the waiting thread(s), i.e. make the event object signalled.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NSyncEventSet(HNSyncEvent hSyncEvent);
```

Parameters

Parameters	Description
HNSyncEvent hSyncEvent	[in] Handle of the sync event.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

The effect of this operation depends on event kind. For auto-reset event this would unblock at most single thread waiting for the event. For manual-reset event all waiting threads will resume untill the event flag is cleared with NSyncEventReset [829]. See allso NSyncEventCreate [827]

Notes

The events are not buffered. If the NSyncEvent [827] is already in signalled state, the operation has no effect. So if we have event of auto-reset kind and no threads were waiting for event, when it is set two times, only the first thread that will wait for event will continue uninterrupted. The second thread would block until event is set again. Consider using the NSemaphore [654] where it is important to preserve count of events set.

Module

NSyncEvent Unit [827]

7.1.2.56.2 Macros

7.1.2.56.2.1 NSyncEventReset Macro

Resets the specified sync event.

C++

```
#define NSyncEventReset(hSyncEvent) NObjectReset(hSyncEvent)
```

Parameters

Parameters	Description
hSyncEvent	[in] Handle to the sync event.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSyncEvent Unit [827]

7.1.2.57 NTextReader Unit

Represents a reader that can read a sequential series of characters.

Functions

	Name	Description
	NTextReaderGetNull [830]	Retrieves a NTextReader with no data to read from.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NTextReaderPeekChar [830]	Reads the next character without changing the state of the reader or the character source. Returns the next available character without actually reading it from the reader.
	NTextReaderReadChar [831]	Reads the next character from the text reader and advances the character position by one character.
	NTextReaderReadChars [831]	Reads a specified maximum number of characters from the current reader and writes the data to a buffer.
	NTextReaderReadLine [831]	Reads a line of characters from the text reader and returns the data as NString [678].
	NTextReaderReadToEnd [832]	Reads all characters from the current position to the end of the text reader and returns them as NString [678].
	NTextReaderSynchronized [832]	Creates a thread-safe wrapper around the specified NTextReader.

Macros

Name	Description
NTextReaderPeekChar [832]	Picks a Unicode (W) or ASCII (A) version of the NTextReaderPeekChar function.
NTextReaderReadChar [833]	Picks a Unicode (W) or ASCII (A) version of the NTextReaderReadChar function.
NTextReaderReadChars [833]	Picks a Unicode (W) or ASCII (A) version of the NTextReaderReadChars function.

7.1.2.57.1 Functions

7.1.2.57.1.1 NTextReaderGetNull Function

Retrieves a NTextReader [829] with no data to read from.

C++

```
NResult N_API NTextReaderGetNull(HNTextReader * phValue);
```

Parameters

Parameters	Description
HNTextReader * phValue	[out] Pointer to a handle to the NTextReader [829] with no data to read from.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextReader Unit [829]

7.1.2.57.1.2 NTextReaderPeekChar Function

Reads the next character without changing the state of the reader or the character source. Returns the next available character without actually reading it from the reader.

C++

```
NResult N_API NTextReaderPeekChar(HNTextReader hReader, NInt * pValue);
```

Parameters

Parameters	Description
HNTextReader hReader	[in] Handle to the NTextReader [829].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt * pValue	[out] Pointer to the integer representing the next character to be read.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextReader Unit [829]

7.1.2.57.1.3 NTextReaderReadChar Function

Reads the next character from the text reader and advances the character position by one character.

C++

```
NResult N_API NTextReaderReadChar(HNTextReader hReader, NInt * pValue);
```

Parameters

Parameters	Description
HNTextReader hReader	[in] Handle to the NTextReader [829].
NInt * pValue	[out] Pointer to the next character from the text reader

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextReader Unit [829]

7.1.2.57.1.4 NTextReaderReadChars Function

Reads a specified maximum number of characters from the current reader and writes the data to a buffer.

C++

```
NResult N_API NTextReaderReadChars(HNTextReader hReader, NChar * szBuffer, NInt count, NInt
* pCharsRead);
```

Parameters

Parameters	Description
HNTextReader hReader	[in] Handle to the NTextReader [829].
NChar * szBuffer	[out] Pointer to the buffer containing the characters read.
NInt count	[in] The maximum number of characters to read.
NInt * pCharsRead	[out] Pointer to the number of characters read.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextReader Unit [829]

7.1.2.57.1.5 NTextReaderReadLine Function

Reads a line of characters from the text reader and returns the data as NString [678].

C++

```
NResult N_API NTextReaderReadLine(HNTextReader hReader, HNString * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNTextReader hReader	[in] Handle to the NTextReader [829].
HNString * phValue	[out] Pointer to a handle to NString [678] containing the line of characters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextReader Unit [829]

7.1.2.57.1.6 NTextReaderReadToEnd Function

Reads all characters from the current position to the end of the text reader and returns them as NString [678].

C++

```
NResult N_API NTextReaderReadToEnd(HNTextReader hReader, HNString * phValue);
```

Parameters

Parameters	Description
HNTextReader hReader	[in] Handle to the NTextReader [829].
HNString * phValue	[out] Pointer to a handle to NString [678] containing the characters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextReader Unit [829]

7.1.2.57.1.7 NTextReaderSynchronized Function

Creates a thread-safe wrapper around the specified NTextReader [829].

C++

```
NResult N_API NTextReaderSynchronized(HNTextReader hReader, HNTextReader * phValue);
```

Parameters

Parameters	Description
HNTextReader hReader	[in] Handle to the NTextReader [829].
HNTextReader * phValue	[out] Pointer to a handle to a NTextReader [829] wrapper.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextReader Unit [829]

7.1.2.57.2 Macros

7.1.2.57.2.1 NTextReaderPeekChar Macro

Picks a Unicode (W) or ASCII (A) version of the NTextReaderPeekChar function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NTextReaderPeekChar N_FUNC_AW(NTextReaderPeekChar)
```

Module

NTextReader Unit [829]

7.1.2.57.2.2 NTextReaderReadChar Macro

Picks a Unicode (W) or ASCII (A) version of the NTextReaderReadChar function.

C++

```
#define NTextReaderReadChar N_FUNC_AW(NTextReaderReadChar)
```

Module

NTextReader Unit [829]

7.1.2.57.2.3 NTextReaderReadChars Macro

Picks a Unicode (W) or ASCII (A) version of the NTextReaderReadChars function.

C++

```
#define NTextReaderReadChars N_FUNC_AW(NTextReaderReadChars)
```

Module

NTextReader Unit [829]

7.1.2.58 NTextWriter Unit

Functions

	Name	Description
	NTextWriterFlush [835]	Clears all buffers for the specified writer and causes any buffered data to be written to the underlying device.
	NTextWriterGetEncoding [835]	Retrieves the encoding of the specified NTextWriter.
	NTextWriterGetNewLine [835]	Sets the line terminator string used by the current NTextWriter.
	NTextWriterGetNull [836]	Creates a NTextWriter with no backing store that can be written to, but not read from.
	NTextWriterSetNewLine [836]	Sets the line terminator string used by the current NTextWriter.
	NTextWriterSynchronized [836]	Creates a thread-safe wrapper around the specified NTextWriter.
	NTextWriterWrite [837]	Writes a character to the text string or stream.
	NTextWriterWriteBoolean [837]	Writes a Boolean to the text string or stream.
	NTextWriterWriteBooleanLine [837]	Writes a Boolean followed by a line terminator to the text string or stream.
	NTextWriterWriteChar [838]	Writes a character to the text string or stream.
	NTextWriterWriteCharLine [838]	Writes a character followed by a line terminator to the text string or stream.
	NTextWriterWriteChars [838]	Writes a character array to the text string or stream.
	NTextWriterWriteCharsLine [839]	Writes a character array followed by a line terminator to the text string or stream.
	NTextWriterWriteDouble [839]	Writes a double to the text string or stream.
	NTextWriterWriteDoubleLine [839]	Writes a double followed by a line terminator to the text string or stream.
	NTextWriterWriteEmptyLine [840]	Writes a line terminator to the text string or stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NTextWriterWriteFormat [840]	Creates a format string with the specified parameters and writes it to the text string or stream.
	NTextWriterWriteFormatLine [840]	Creates a format string with the specified parameters and writes it followed by a line terminator to the text string or stream.
	NTextWriterWriteFormatLineVA [841]	Creates a format string with the specified arguments and writes it followed by a line terminator to the text string or stream.
	NTextWriterWriteFormatVA [841]	Creates a format string with the specified arguments and writes it to the text string or stream.
	NTextWriterWriteInt32 [842]	Writes a Int32 to the text string or stream.
	NTextWriterWriteInt32Line [842]	Writes a Int32 followed by a line terminator to the text string or stream.
	NTextWriterWriteInt64 [842]	Writes a Int64 to the text string or stream.
	NTextWriterWriteInt64Line [843]	Writes a Int64 followed by a line terminator to the text string or stream.
	NTextWriterWriteLine [843]	Writes a character followed by a line terminator to the text string or stream.
	NTextWriterWriteObject [843]	Writes an object to the text string or stream.
	NTextWriterWriteObjectLine [844]	Writes an object followed by a line terminator to the text string or stream.
	NTextWriterWritePointer [844]	Writes a pointer to the text string or stream.
	NTextWriterWritePointerLine [844]	Writes a pointer followed by a line terminator to the text string or stream.
	NTextWriterWriteSingle [845]	Writes a single to the text string or stream.
	NTextWriterWriteSingleLine [845]	Writes a single followed by a line terminator to the text string or stream.
	NTextWriterWriteSizeType [845]	Writes a SizeType to the text string or stream.
	NTextWriterWriteSizeTypeLine [846]	Writes a SizeType followed by a line terminator to the text string or stream.
	NTextWriterWriteSSizeType [846]	Writes a SSizeType to the text string or stream.
	NTextWriterWriteSSizeTypeLine [846]	Writes a SSizeType followed by a line terminator to the text string or stream.
	NTextWriterWriteUInt32 [847]	Writes a UInt32 to the text string or stream.
	NTextWriterWriteUInt32Line [847]	Writes a UInt32 followed by a line terminator to the text string or stream.
	NTextWriterWriteUInt64 [847]	Writes a UInt64 to the text string or stream.
	NTextWriterWriteUInt64Line [848]	Writes a UInt64 followed by a line terminator to the text string or stream.
	NTextWriterWriteValueLineP [848]	Writes a specified value followed by a line terminator to the text string or stream.
	NTextWriterWriteValueP [848]	Writes a specified value to the text string or stream.

Macros

Name	Description
NTextWriterSetNewLine [849]	Picks a Unicode (W) or ASCII (A) version of the NTextWriterSetNewLine function.
NTextWriterWrite [849]	Picks a Unicode (W) or ASCII (A) version of the NTextWriterWrite function.
NTextWriterWriteChar [849]	Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteChar function.
NTextWriterWriteCharLine [849]	Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteCharLine function.
NTextWriterWriteChars [850]	Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteChars function.
NTextWriterWriteCharsLine [850]	Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteCharsLine function.
NTextWriterWriteFormat [850]	Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteFormat function.
NTextWriterWriteFormatLine [850]	Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteFormatLine function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTextWriterWriteFormatLineVA [850]	Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteFormatLineVA function.
NTextWriterWriteFormatVA [850]	Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteFormatVA function.
NTextWriterWriteLine [851]	Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteLine function.

7.1.2.58.1 Functions

7.1.2.58.1.1 NTextWriterFlush Function

Clears all buffers for the specified writer and causes any buffered data to be written to the underlying device.

C++

```
NResult N_API NTextWriterFlush(HNTextWriter hWriter);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.2 NTextWriterGetEncoding Function

Retrieves the encoding of the specified NTextWriter [833].

C++

```
NResult N_API NTextWriterGetEncoding(HNTextWriter hWriter, NEncoding * pValue);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NEncoding * pValue	[out] Pointer to the encoding value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.3 NTextWriterGetNewLine Function

Sets the line terminator string used by the current NTextWriter [833].

C++

```
NResult N_API NTextWriterGetNewLine(HNTextWriter hWriter, HNString * phValue);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
HNString * phValue	[in] Pointer to a handle to the line terminator string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.4 NTextWriterGetNull Function

Creates a NTextWriter [833] with no backing store that can be written to, but not read from.

C++

```
NResult N_API NTextWriterGetNull(HNTextWriter * phValue);
```

Parameters

Parameters	Description
HNTextWriter * phValue	[out] Pointer to a handle to the NTextWriter [833] with no backing store that can be written to, but not read from..

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.5 NTextWriterSetNewLine Function

Sets the line terminator string used by the current NTextWriter [833].

C++

```
NResult N_API NTextWriterSetNewLine(HNTextWriter hWriter, const NChar * szValue);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
const NChar * szValue	[in] Pointer to new line terminator string value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.6 NTextWriterSynchronized Function

Creates a thread-safe wrapper around the specified NTextWriter [833].

C++

```
NResult N_API NTextWriterSynchronized(HNTextWriter hWriter, HNTextWriter * phValue);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
HNTextWriter * phValue	[out] Pointer to a handle to the thread-safe wrapper.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.7 NTextWriterWrite Function

Writes a character to the text string or stream.

C++

```
NResult N_API NTextWriterWrite(HNTextWriter hWriter, const NChar * szValue);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
const NChar * szValue	[in] Pointer to the character to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.8 NTextWriterWriteBoolean Function

Writes a Boolean to the text string or stream.

C++

```
NResult N_API NTextWriterWriteBoolean(HNTextWriter hWriter, NBool value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NBool value	[in] The Boolean to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.9 NTextWriterWriteBooleanLine Function

Writes a Boolean followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteBooleanLine(HNTextWriter hWriter, NBool value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NBool value	[in] The Boolean to write to the text stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.10 NTextWriterWriteChar Function

Writes a character to the text string or stream.

C++

```
NResult N_API NTextWriterWriteChar(HNTextWriter hWriter, NChar value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NChar value	[in] Pointer to the character to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.11 NTextWriterWriteCharLine Function

Writes a character followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteCharLine(HNTextWriter hWriter, NChar value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NChar value	[in] Pointer to the character to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.12 NTextWriterWriteChars Function

Writes a character array to the text string or stream.

C++

```
NResult N_API NTextWriterWriteChars(HNTextWriter hWriter, const NChar * arValue, NInt
valueLength);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
const NChar * arValue	[in] Pointer to the character array to write to the text stream.
NInt valueLength	[in] Length of the char array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.13 NTextWriterWriteCharsLine Function

Writes a character array followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteCharsLine(HNTextWriter hWriter, const NChar * arValue, NInt
valueLength);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
const NChar * arValue	[in] Pointer to the character array to write to the text stream.
NInt valueLength	[in] Length of the char array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.14 NTextWriterWriteDouble Function

Writes a double to the text string or stream.

C++

```
NResult N_API NTextWriterWriteDouble(HNTextWriter hWriter, NDouble value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NDouble value	[in] The double to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.15 NTextWriterWriteDoubleLine Function

Writes a double followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteDoubleLine(HNTextWriter hWriter, NDouble value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NDouble value	[in] The double to write to the text stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.16 NTextWriterWriteEmptyLine Function

Writes a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteEmptyLine(HNTextWriter hWriter);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.17 NTextWriterWriteFormat Function

Creates a format string with the specified parameters and writes it to the text string or stream.

C++

```
NResult N_API NTextWriterWriteFormat(HNTextWriter hWriter, const NChar * szFormat, ...
multipleParameters);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
const NChar * szFormat	[in] Pointer to composite format string.
... multipleParameters	[in] Format parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTextWriter Unit [833]

7.1.2.58.1.18 NTextWriterWriteFormatLine Function

Creates a format string with the specified parameters and writes it followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteFormatLine(HNTextWriter hWriter, const NChar * szFormat, ...
multipleParameters);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
const NChar * szFormat	[in] Pointer to composite format string.
... multipleParameters	[in] Format parameters.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTextWriter Unit [833]

7.1.2.58.1.19 NTextWriterWriteFormatLineVA Function

Creates a format string with the specified arguments and writes it followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteFormatLineVA(HNTextWriter hWriter, const NChar * szFormat,
va_list args);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
const NChar * szFormat	[in] Pointer to composite format string.
va_list args	[in] Format arguments.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTextWriter Unit [833]

7.1.2.58.1.20 NTextWriterWriteFormatVA Function

Creates a format string with the specified arguments and writes it to the text string or stream.

C++

```
NResult N_API NTextWriterWriteFormatVA(HNTextWriter hWriter, const NChar * szFormat,
va_list args);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
const NChar * szFormat	[in] Pointer to composite format string.
va_list args	[in] Format arguments.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTextWriter Unit [833]

7.1.2.58.1.21 NTextWriterWriteInt32 Function

Writes a Int32 to the text string or stream.

C++

```
NResult N_API NTextWriterWriteInt32(HNTextWriter hWriter, NInt value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NInt value	[in] The Int32 to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.22 NTextWriterWriteInt32Line Function

Writes a Int32 followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteInt32Line(HNTextWriter hWriter, NInt value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NInt value	[in] The Int32 to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.23 NTextWriterWriteInt64 Function

Writes a Int64 to the text string or stream.

C++

```
NResult N_API NTextWriterWriteInt64(HNTextWriter hWriter, NLong value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NLong value	[in] The Int64 to write to the text stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.24 NTextWriterWriteInt64Line Function

Writes a Int64 followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteInt64Line(HNTextWriter hWriter, NLong value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NLong value	[in] The Int64 to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.25 NTextWriterWriteLine Function

Writes a character followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteLine(HNTextWriter hWriter, const NChar * szValue);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
const NChar * szValue	[in] Pointer to the character to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.26 NTextWriterWriteObject Function

Writes an object to the text string or stream.

C++

```
NResult N_API NTextWriterWriteObject(HNTextWriter hWriter, HNObject hObject);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
HNObject hObject	[in] Handle to the object to write to the text stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.27 NTextWriterWriteObjectLine Function

Writes an object followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteObjectLine(HNTextWriter hWriter, HNObject hObject);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
HNObject hObject	[in] Handle to the object to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.28 NTextWriterWritePointer Function

Writes a pointer to the text string or stream.

C++

```
NResult N_API NTextWriterWritePointer(HNTextWriter hWriter, const void * value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
const void * value	[in] The pointer to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.29 NTextWriterWritePointerLine Function

Writes a pointer followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWritePointerLine(HNTextWriter hWriter, const void * value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
const void * value	[in] The pointer to write to the text stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.30 NTextWriterWriteSingle Function

Writes a single to the text string or stream.

C++

```
NResult N_API NTextWriterWriteSingle(HNTextWriter hWriter, NFloat value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NFloat value	[in] The single to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.31 NTextWriterWriteSingleLine Function

Writes a single followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteSingleLine(HNTextWriter hWriter, NFloat value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NFloat value	[in] The single to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.32 NTextWriterWriteSizeType Function

Writes a SizeType to the text string or stream.

C++

```
NResult N_API NTextWriterWriteSizeType(HNTextWriter hWriter, NSizeType value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NSizeType value	[in] The SizeType to write to the text stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.33 NTextWriterWriteSizeTypeLine Function

Writes a SizeType followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteSizeTypeLine(HNTextWriter hWriter, NSizeType value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NSizeType value	[in] The SizeType to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.34 NTextWriterWriteSSizeType Function

Writes a SSizeType to the text string or stream.

C++

```
NResult N_API NTextWriterWriteSSizeType(HNTextWriter hWriter, NSSizeType value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NSSizeType value	[in] The SSizeType to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.35 NTextWriterWriteSSizeTypeLine Function

Writes a SSizeType followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteSSizeTypeLine(HNTextWriter hWriter, NSSizeType value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NSSizeType value	[in] The SSizeType to write to the text stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.36 NTextWriterWriteUInt32 Function

Writes a UInt32 to the text string or stream.

C++

```
NResult N_API NTextWriterWriteUInt32(HNTextWriter hWriter, NUInt value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NUInt value	[in] The UInt32 to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.37 NTextWriterWriteUInt32Line Function

Writes a UInt32 followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteUInt32Line(HNTextWriter hWriter, NUInt value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NUInt value	[in] The UInt32 to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.38 NTextWriterWriteUInt64 Function

Writes a UInt64 to the text string or stream.

C++

```
NResult N_API NTextWriterWriteUInt64(HNTextWriter hWriter, NULong value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NULong value	[in] The UInt64 to write to the text stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.39 NTextWriterWriteUInt64Line Function

Writes a UInt64 followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteUInt64Line(HNTextWriter hWriter, NULong value);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NULong value	[in] The UInt64 to write to the text stream.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.40 NTextWriterWriteValueLineP Function

Writes a specified value followed by a line terminator to the text string or stream.

C++

```
NResult N_API NTextWriterWriteValueLineP(HNTextWriter hWriter, NTypeOfProc pTypeOf, const
void * pValue, NSizeType valueSize);
```

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NTypeOfProc pTypeOf	_nt_
const void * pValue	[in] The value to write to the text stream.
NSizeType valueSize	_nt_

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.1.41 NTextWriterWriteValueP Function

Writes a specified value to the text string or stream.

C++

```
NResult N_API NTextWriterWriteValueP(HNTextWriter hWriter, NTypeOfProc pTypeOf, const void
* pValue, NSizeType valueSize);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNTextWriter hWriter	[in] Handle to the NTextWriter [833].
NTypeOfProc pTypeOf	[in] Type of the value.
const void * pValue	[in] The value to write to the text stream.
NSizeType valueSize	[in] Size of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTextWriter Unit [833]

7.1.2.58.2 Macros

7.1.2.58.2.1 NTextWriterSetNewLine Macro

Picks a Unicode (W) or ASCII (A) version of the NTextWriterSetNewLine function.

C++

```
#define NTextWriterSetNewLine N_FUNC_AW(NTextWriterSetNewLine)
```

Module

NTextWriter Unit [833]

7.1.2.58.2.2 NTextWriterWrite Macro

Picks a Unicode (W) or ASCII (A) version of the NTextWriterWrite function.

C++

```
#define NTextWriterWrite N_FUNC_AW(NTextWriterWrite)
```

Module

NTextWriter Unit [833]

7.1.2.58.2.3 NTextWriterWriteChar Macro

Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteChar function.

C++

```
#define NTextWriterWriteChar N_FUNC_AW(NTextWriterWriteChar)
```

Module

NTextWriter Unit [833]

7.1.2.58.2.4 NTextWriterWriteCharLine Macro

Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteCharLine function.

C++

```
#define NTextWriterWriteCharLine N_FUNC_AW(NTextWriterWriteCharLine)
```

Module

NTextWriter Unit [833]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.58.2.5 NTextWriterWriteChars Macro

Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteChars function.

C++

```
#define NTextWriterWriteChars N_FUNC_AW(NTextWriterWriteChars)
```

Module

NTextWriter Unit [833]

7.1.2.58.2.6 NTextWriterWriteCharsLine Macro

Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteCharsLine function.

C++

```
#define NTextWriterWriteCharsLine N_FUNC_AW(NTextWriterWriteCharsLine)
```

Module

NTextWriter Unit [833]

7.1.2.58.2.7 NTextWriterWriteFormat Macro

Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteFormat function.

C++

```
#define NTextWriterWriteFormat N_FUNC_AW(NTextWriterWriteFormat)
```

Module

NTextWriter Unit [833]

7.1.2.58.2.8 NTextWriterWriteFormatLine Macro

Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteFormatLine function.

C++

```
#define NTextWriterWriteFormatLine N_FUNC_AW(NTextWriterWriteFormatLine)
```

Module

NTextWriter Unit [833]

7.1.2.58.2.9 NTextWriterWriteFormatLineVA Macro

Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteFormatLineVA function.

C++

```
#define NTextWriterWriteFormatLineVA N_FUNC_AW(NTextWriterWriteFormatLineVA)
```

Module

NTextWriter Unit [833]

7.1.2.58.2.10 NTextWriterWriteFormatVA Macro

Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteFormatVA function.

C++

```
#define NTextWriterWriteFormatVA N_FUNC_AW(NTextWriterWriteFormatVA)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTextWriter Unit [833]

7.1.2.58.2.11 NTextWriterWriteLine Macro

Picks a Unicode (W) or ASCII (A) version of the NTextWriterWriteLine function.

C++

```
#define NTextWriterWriteLine N_FUNC_AW(NTextWriterWriteLine)
```

Module

NTextWriter Unit [833]

7.1.2.59 NThread Unit

Cross platform threading.

Files

Name	Description
NThread.h [855]	Header file for the NThread unit.

Functions

	Name	Description
	NThreadCreate [851]	Execute function in parallel in new thread.
	NThreadCreateFromOSHandle [852]	Creates a new NThread for the specified thread handle.
	NThreadCreateParameterized [852]	Execute function in parallel in new thread with specified additional parameters.
	NThreadGetCurrent [853]	Gets the currently running thread.
	NThreadGetCurrentId [853]	Gets the identifier of the currently running thread.
	NThreadGetId [853]	Gets the identifier of a specified NThread.
	NThreadGetOSHandle [854]	Gets OS handle for the specified thread.
	NThreadJoin [854]	Suspend the execution of current thread until the given thread terminates.
	NThreadSleep [854]	Suspend the execution of current thread for some time.

7.1.2.59.1 Functions

7.1.2.59.1.1 NThreadCreate Function

Execute function in parallel in new thread.

C++

```
NResult N_API NThreadCreate(NThreadStartProc pThreadStart, void * pThreadStartParam,
HNThread * phThread);
```

Parameters

Parameters	Description
NThreadStartProc pThreadStart	[in] Pointer to a function that will be executed in new thread.
HNThread * phThread	[out] Pointer to a variable that will receive handle of the created thread.
pParam	[in] Parameter that will be passed to pThreadStart function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NThread Unit [851]

7.1.2.59.1.2 NThreadCreateFromOSHandle Function

Creates a new NThread [851] for the specified thread handle.

C++

```
NResult N_API NThreadCreateFromOSHandle(NHandle handle, NBool ownsHandle, HNThread *
phThread);
```

Parameters

Parameters	Description
NHandle handle	[in] A thread handle for the thread that the current NThread [851] will encapsulate.
NBool ownsHandle	[in] NTrue [1064] if the thread handle will be owned by this NThread [851]; otherwise, NFalse [1041].
HNThread * phThread	[out] A pointer to the created NThread [851].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NThreadGetOSHandle [854]

Module

NThread Unit [851]

7.1.2.59.1.3 NThreadCreateParameterized Function

Execute function in parallel in new thread with specified additional parameters.

C++

```
NResult N_API NThreadCreateParameterized(NParameterizedThreadStartProc pThreadStart, void *
pThreadStartParam, HNObject hParameter, HNThread * phThread);
```

Parameters

Parameters	Description
NParameterizedThreadStartProc pThreadStart	[in] A thread handle for the thread that the current NThread [851] will encapsulate.
void * pThreadStartParam	[in] Parameter that will be passed to pThreadStart function.
HNObject hParameter	[in] Additional parameters.
HNThread * phThread	[out] Pointer to a variable that will receive handle of the created thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NThread Unit [851]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.59.1.4 NThreadGetCurrent Function

Gets the currently running thread.

C++

```
NResult N_API NThreadGetCurrent(HNThread * phValue);
```

Parameters

Parameters	Description
HNThread * phValue	[out] A pointer to the currently running thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NThread Unit [851]

7.1.2.59.1.5 NThreadGetCurrentId Function

Gets the identifier of the currently running thread.

C++

```
NResult N_API NThreadGetCurrentId(NSizeType * pValue);
```

Parameters

Parameters	Description
NSizeType * pValue	[out] A pointer to the identifier of the currently running thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NThread Unit [851]

7.1.2.59.1.6 NThreadGetId Function

Gets the identifier of a specified NThread [851].

C++

```
NResult N_API NThreadGetId(HNThread hThread, NSizeType * pValue);
```

Parameters

Parameters	Description
HNThread hThread	[in] A handle to NThread [851].
NSizeType * pValue	[out] A pointer to the identifier of NThread [851].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NThread Unit [851]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.59.1.7 NThreadGetOSHandle Function

Gets OS handle for the specified thread.

C++

```
NResult N_API NThreadGetOSHandle(HNThread hThread, NHandle * phValue);
```

Parameters

Parameters	Description
HNThread hThread	[in] A handle to NThread [851].
NHandle * phValue	[out] OS handle to a thread.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NThread Unit [851]

7.1.2.59.1.8 NThreadJoin Function

Suspend the execution of current thread until the given thread terminates.

C++

```
NResult N_API NThreadJoin(HNThread hThread);
```

Parameters

Parameters	Description
HNThread hThread	[in] Handle of the thread to wait for.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NThread Unit [851]

7.1.2.59.1.9 NThreadSleep Function

Suspend the execution of current thread for some time.

C++

```
NResult N_API NThreadSleep(NInt milliseconds);
```

Parameters

Parameters	Description
NInt milliseconds	[in] Number of milliseconds to wait before the calling thread can continue.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NThread Unit [851]

7.1.2.59.2 Files

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.59.2.1 NThread.h

Header file for the NThread [851] unit.

Functions

	Name	Description
	NThreadCreate [851]	Execute function in parallel in new thread.
	NThreadCreateFromOSHandle [852]	Creates a new NThread [851] for the specified thread handle.
	NThreadCreateParameterized [852]	Execute function in parallel in new thread with specified additional parameters.
	NThreadGetCurrent [853]	Gets the currently running thread.
	NThreadGetCurrentId [853]	Gets the identifier of the currently running thread.
	NThreadGetId [853]	Gets the identifier of a specified NThread [851].
	NThreadGetOSHandle [854]	Gets OS handle for the specified thread.
	NThreadJoin [854]	Suspend the execution of current thread until the given thread terminates.
	NThreadSleep [854]	Suspend the execution of current thread for some time.

Module

NThread Unit [851]

7.1.2.60 NTimeSpan Unit

Represents a time interval.

C++

```
typedef NTimeSpan_ NTimeSpan;
```

Functions

	Name	Description
	NTimeSpanAdd [856]	Adds two TimeSpan instances.
	NTimeSpanCompare [857]	Compares two instances of TimeSpan and returns an indication of their relative values.
	NTimeSpanCreate [857]	Initializes a new instance of NTimeSpan structure.
	NTimeSpanCreateFromDays [858]	Initializes a new instance of NTimeSpan structure with the number of days only specified.
	NTimeSpanCreateFromHours [858]	Initializes a new instance of NTimeSpan structure with the number of hours only specified.
	NTimeSpanCreateFromMilliseconds [858]	Initializes a new instance of NTimeSpan structure with the number of milliseconds only specified.
	NTimeSpanCreateFromMinutes [859]	Initializes a new instance of NTimeSpan structure with the number of minutes only specified.
	NTimeSpanCreateFromSeconds [859]	Initializes a new instance of NTimeSpan structure with the number of seconds only specified.
	NTimeSpanCreateFromTicks [859]	Initializes a new instance of NTimeSpan structure with the specified number of timer ticks.
	NTimeSpanDecode [860]	Decodes the specified TimeSpan instance.
	NTimeSpanDuration [860]	Returns a new TimeSpan object whose value is the absolute value of the current TimeSpan object.
	NTimeSpanGetDays [860]	Gets the days component of the specified TimeSpan.
	NTimeSpanGetHours [861]	Gets the hours component of the specified TimeSpan.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NTimeSpanGetMilliseconds [861]	Gets the milliseconds component of the specified TimeSpan.
	NTimeSpanGetMinutes [861]	Gets the minutes component of the specified TimeSpan.
	NTimeSpanGetSeconds [862]	Gets the seconds component of the specified TimeSpan.
	NTimeSpanGetTicks [862]	Gets the number of ticks that represent the value of the specified TimeSpan instance.
	NTimeSpanGetTotalDays [862]	Gets the value of the specified TimeSpan instance expressed in whole and fractional days.
	NTimeSpanGetTotalHours [863]	Gets the value of the specified TimeSpan instance expressed in whole and fractional hours.
	NTimeSpanGetTotalMilliseconds [863]	Gets the value of the specified TimeSpan instance expressed in whole and fractional milliseconds.
	NTimeSpanGetTotalMinutes [863]	Gets the value of the specified TimeSpan instance expressed in whole and fractional minutes.
	NTimeSpanGetTotalSeconds [864]	Gets the value of the specified TimeSpan instance expressed in whole and fractional seconds.
	NTimeSpanIsTimeValid [864]	Checks whether the specified time values are valid.
	NTimeSpanIsValid [865]	Checks whether the specified TimeSpan values are valid.
	NTimeSpanNegate [865]	Returns a new TimeSpan object whose value is the negated value of the specified instance.
	NTimeSpanSubtract [865]	Subtracts one TimeSpan instance from another.
	NTimeSpanToString [866]	Gets a string representation of the specified TimeSpan instance.

Macros

Name	Description
N_TIME_SPAN_MAX [866]	Maximum TimeSpan value.
N_TIME_SPAN_MIN [866]	Minimum TimeSpan value.
N_TIME_SPAN_TICKS_PER_DAY [867]	Number of timer ticks in one day.
N_TIME_SPAN_TICKS_PER_HOUR [867]	Number of timer ticks in one hour.
N_TIME_SPAN_TICKS_PER_MILLISECOND [867]	Number of timer ticks in one millisecond.
N_TIME_SPAN_TICKS_PER_MINUTE [867]	Number of timer ticks in one minute.
N_TIME_SPAN_TICKS_PER_SECOND [867]	Number of timer ticks in one second.
NTimeSpanToString [868]	Picks a Unicode (W) or ASCII (A) version of the NTimeSpanToString function.

Types

Name	Description
NTimeSpan_ [866]	The TimeSpan instance.

7.1.2.60.1 Functions

7.1.2.60.1.1 NTimeSpanAdd Function

Adds two TimeSpan instances.

C++

```
NResult N_API NTimeSpanAdd(NTimeSpan_ value1, NTimeSpan_ value2, NTimeSpan_ * pResult);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NTimeSpan_ value1	[in] The first TimeSpan instance.
NTimeSpan_ value2	[in] The second TimeSpan instance.
NTimeSpan_ * pResult	[out] The resulting TimeSpan instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.2 NTimeSpanCompare Function

Compares two instances of TimeSpan and returns an indication of their relative values.

C++

```
NResult N_API NTimeSpanCompare(NTimeSpan_ value1, NTimeSpan_ value2, NInt * pResult);
```

Parameters

Parameters	Description
NTimeSpan_ value1	[in] The first TimeSpan instance.
NTimeSpan_ value2	[in] The second TimeSpan instance.
NInt * pResult	[out] Pointer to an integer indicating the relative values of the instances.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.3 NTimeSpanCreate Function

Initializes a new instance of NTimeSpan [855] structure.

C++

```
NResult N_API NTimeSpanCreate(NInt days, NInt hours, NInt minutes, NInt seconds, NInt
milliseconds, NTimeSpan_ * pTimeSpan);
```

Parameters

Parameters	Description
NInt days	[in] The days value.
NInt hours	[in] The hours value.
NInt minutes	[in] The minutes value.
NInt seconds	[in] The seconds value.
NInt milliseconds	[in] The milliseconds value.
NTimeSpan_ * pTimeSpan	[out] Pointer to the new instance of NTimeSpan [855] structure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Type

NTimeSpan Unit [855]

7.1.2.60.1.4 NTimeSpanCreateFromDays Function

Initializes a new instance of NTimeSpan [855] structure with the number of days only specified.

C++

```
NResult N_API NTimeSpanCreateFromDays(NDouble value, NTimeSpan_ * pTimeSpan);
```

Parameters

Parameters	Description
NDouble value	[in] The days value.
NTimeSpan_ * pTimeSpan	[out] Pointer to the new instance of NTimeSpan [855] structure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.5 NTimeSpanCreateFromHours Function

Initializes a new instance of NTimeSpan [855] structure with the number of hours only specified.

C++

```
NResult N_API NTimeSpanCreateFromHours(NDouble value, NTimeSpan_ * pTimeSpan);
```

Parameters

Parameters	Description
NDouble value	[in] The hours value.
NTimeSpan_ * pTimeSpan	[out] Pointer to the new instance of NTimeSpan [855] structure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.6 NTimeSpanCreateFromMilliseconds Function

Initializes a new instance of NTimeSpan [855] structure with the number of milliseconds only specified.

C++

```
NResult N_API NTimeSpanCreateFromMilliseconds(NDouble value, NTimeSpan_ * pTimeSpan);
```

Parameters

Parameters	Description
NDouble value	[in] The milliseconds value.
NTimeSpan_ * pTimeSpan	[out] Pointer to the new instance of NTimeSpan [855] structure.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.7 NTimeSpanCreateFromMinutes Function

Initializes a new instance of NTimeSpan [855] structure with the number of minutes only specified.

C++

```
NResult N_API NTimeSpanCreateFromMinutes(NDouble value, NTimeSpan_ * pTimeSpan);
```

Parameters

Parameters	Description
NDouble value	[in] The minutes value.
NTimeSpan_ * pTimeSpan	[out] Pointer to the new instance of NTimeSpan [855] structure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.8 NTimeSpanCreateFromSeconds Function

Initializes a new instance of NTimeSpan [855] structure with the number of seconds only specified.

C++

```
NResult N_API NTimeSpanCreateFromSeconds(NDouble value, NTimeSpan_ * pTimeSpan);
```

Parameters

Parameters	Description
NDouble value	[in] The seconds value.
NTimeSpan_ * pTimeSpan	[out] Pointer to the new instance of NTimeSpan [855] structure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.9 NTimeSpanCreateFromTicks Function

Initializes a new instance of NTimeSpan [855] structure with the specified number of timer ticks.

C++

```
NResult N_API NTimeSpanCreateFromTicks(NLong value, NTimeSpan_ * pTimeSpan);
```

Parameters

Parameters	Description
NLong value	[in] The number of timer ticks.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTimeSpan_ * pTimeSpan	[out] Pointer to the new instance of NTimeSpan [855] structure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.10 NTimeSpanDecode Function

Decodes the specified TimeSpan instance.

C++

```
NResult N_API NTimeSpanDecode(NTimeSpan_ timeSpan, NInt * pDays, NInt * pHours, NInt *
pMinutes, NInt * pSeconds, NInt * pMilliseconds);
```

Parameters

Parameters	Description
NTimeSpan_ timeSpan	[in] The TimeSpan instance to be decoded.
NInt * pDays	[out] Pointer to the days value.
NInt * pHours	[out] Pointer to the hours value.
NInt * pMinutes	[out] Pointer to the minutes value.
NInt * pSeconds	[out] Pointer to the seconds value.
NInt * pMilliseconds	[out] Pointer to the milliseconds value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.11 NTimeSpanDuration Function

Returns a new TimeSpan object whose value is the absolute value of the current TimeSpan object.

C++

```
NResult N_API NTimeSpanDuration(NTimeSpan_ value, NTimeSpan_ * pResult);
```

Parameters

Parameters	Description
NTimeSpan_ value	[in] The original TimeSpan instance.
NTimeSpan_ * pResult	[out] Pointer to the new TimeSpan representing the absolute value of the original TimeSpan.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.12 NTimeSpanGetDays Function

Gets the days component of the specified TimeSpan.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NTimeSpanGetDays(NTimeSpan_ timeSpan, NInt * pValue);
```

Parameters

Parameters	Description
NTimeSpan_ timeSpan	[in] The TimeSpan instance.
NInt * pValue	[out] Pointer to the days value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.13 NTimeSpanGetHours Function

Gets the hours component of the specified TimeSpan.

C++

```
NResult N_API NTimeSpanGetHours(NTimeSpan_ timeSpan, NInt * pValue);
```

Parameters

Parameters	Description
NTimeSpan_ timeSpan	[in] The TimeSpan instance.
NInt * pValue	[out] Pointer to the hours value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.14 NTimeSpanGetMilliseconds Function

Gets the milliseconds component of the specified TimeSpan.

C++

```
NResult N_API NTimeSpanGetMilliseconds(NTimeSpan_ timeSpan, NInt * pValue);
```

Parameters

Parameters	Description
NTimeSpan_ timeSpan	[in] The TimeSpan instance.
NInt * pValue	[out] Pointer to the milliseconds value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.15 NTimeSpanGetMinutes Function

Gets the minutes component of the specified TimeSpan.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NTimeSpanGetMinutes(NTimeSpan_ timeSpan, NInt * pValue);
```

Parameters

Parameters	Description
NTimeSpan_ timeSpan	[in] The TimeSpan instance.
NInt * pValue	[out] Pointer to the minutes value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.16 NTimeSpanGetSeconds Function

Gets the seconds component of the specified TimeSpan.

C++

```
NResult N_API NTimeSpanGetSeconds(NTimeSpan_ timeSpan, NInt * pValue);
```

Parameters

Parameters	Description
NTimeSpan_ timeSpan	[in] The TimeSpan instance.
NInt * pValue	[out] Pointer to the seconds value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.17 NTimeSpanGetTicks Function

Gets the number of ticks that represent the value of the specified TimeSpan instance.

C++

```
NResult N_API NTimeSpanGetTicks(NTimeSpan_ timeSpan, NLong * pValue);
```

Parameters

Parameters	Description
NTimeSpan_ timeSpan	[in] The TimeSpan instance.
NLong * pValue	[out] Pointer to the number of ticks.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.18 NTimeSpanGetTotalDays Function

Gets the value of the specified TimeSpan instance expressed in whole and fractional days.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NTimeSpanGetTotalDays(NTimeSpan_ timeSpan, NDouble * pValue);
```

Parameters

Parameters	Description
NTimeSpan_ timeSpan	[in] The TimeSpan instance.
NDouble * pValue	[out] Pointer to the number of days.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.19 NTimeSpanGetTotalHours Function

Gets the value of the specified TimeSpan instance expressed in whole and fractional hours.

C++

```
NResult N_API NTimeSpanGetTotalHours(NTimeSpan_ timeSpan, NDouble * pValue);
```

Parameters

Parameters	Description
NTimeSpan_ timeSpan	[in] The TimeSpan instance.
NDouble * pValue	[out] Pointer to the number of hours.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.20 NTimeSpanGetTotalMilliseconds Function

Gets the value of the specified TimeSpan instance expressed in whole and fractional milliseconds.

C++

```
NResult N_API NTimeSpanGetTotalMilliseconds(NTimeSpan_ timeSpan, NDouble * pValue);
```

Parameters

Parameters	Description
NTimeSpan_ timeSpan	[in] The TimeSpan instance.
NDouble * pValue	[out] Pointer to the number of milliseconds.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.21 NTimeSpanGetTotalMinutes Function

Gets the value of the specified TimeSpan instance expressed in whole and fractional minutes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NTimeSpanGetTotalMinutes(NTimeSpan_ timeSpan, NDouble * pValue);
```

Parameters

Parameters	Description
NTimeSpan_ timeSpan	[in] The TimeSpan instance.
NDouble * pValue	[out] Pointer to the number of minutes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.22 NTimeSpanGetTotalSeconds Function

Gets the value of the specified TimeSpan instance expressed in whole and fractional seconds.

C++

```
NResult N_API NTimeSpanGetTotalSeconds(NTimeSpan_ timeSpan, NDouble * pValue);
```

Parameters

Parameters	Description
NTimeSpan_ timeSpan	[in] The TimeSpan instance.
NDouble * pValue	[out] Pointer to the number of seconds.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.23 NTimeSpanIsTimeValid Function

Checks whether the specified time values are valid.

C++

```
NBool N_API NTimeSpanIsTimeValid(NInt days, NInt hours, NInt minutes, NInt seconds, NInt
milliseconds);
```

Parameters

Parameters	Description
NInt days	[in] The days value.
NInt hours	[in] The hours value.
NInt minutes	[in] The minutes value.
NInt seconds	[in] The seconds value.
NInt milliseconds	[in] The milliseconds value.

Returns

NTrue [1064] if the time is valid; else NFalse [1041].

Type

NTimeSpan Unit [855]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.60.1.24 NTimeSpanIsValid Function

Checks whether the specified TimeSpan values are valid.

C++

```
NBool N_API NTimeSpanIsValid(NTimeSpan_ value);
```

Parameters

Parameters	Description
NTimeSpan_ value	[in] The TimeSpan to be checked.

Returns

NTrue [1064] if the TimeSpan is valid; else NFalse [1041].

Type

NTimeSpan Unit [855]

7.1.2.60.1.25 NTimeSpanNegate Function

Returns a new TimeSpan object whose value is the negated value of the specified instance.

C++

```
NResult N_API NTimeSpanNegate(NTimeSpan_ value, NTimeSpan_ * pResult);
```

Parameters

Parameters	Description
NTimeSpan_ value	[in] The original TimeSpan instance.
NTimeSpan_ * pResult	[out] Pointer to the negative TimeSpan instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.1.26 NTimeSpanSubtract Function

Subtracts one TimeSpan instance from another.

C++

```
NResult N_API NTimeSpanSubtract(NTimeSpan_ value1, NTimeSpan_ value2, NTimeSpan_ * pResult);
```

Parameters

Parameters	Description
NTimeSpan_ value1	[in] The first TimeSpan instance.
NTimeSpan_ value2	[in] The second TimeSpan instance.
NTimeSpan_ * pResult	[out] The resulting TimeSpan instance.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.60.1.27 NTimeSpanToString Function

Gets a string representation of the specified TimeSpan instance.

C++

```
NResult N_API NTimeSpanToString(NTimeSpan value, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
NTimeSpan value	[in] The TimeSpan instance.
const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] Pointer to a handle to the string representing the TimeSpan.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Type

NTimeSpan Unit [855]

7.1.2.60.2 Types

7.1.2.60.2.1 NTimeSpan_ Type

The TimeSpan instance.

C++

```
typedef NLong NTimeSpan_;
```

Type

NTimeSpan Unit [855]

7.1.2.60.3 Macros

7.1.2.60.3.1 N_TIME_SPAN_MAX Macro

Maximum TimeSpan value.

C++

```
#define N_TIME_SPAN_MAX N_LONG_MAX
```

Type

NTimeSpan Unit [855]

7.1.2.60.3.2 N_TIME_SPAN_MIN Macro

Minimum TimeSpan value.

C++

```
#define N_TIME_SPAN_MIN N_LONG_MIN
```

Type

NTimeSpan Unit [855]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.60.3.3 N_TIME_SPAN_TICKS_PER_DAY Macro

Number of timer ticks in one day.

C++

```
#define N_TIME_SPAN_TICKS_PER_DAY 864000000000LL
```

Type

NTimeSpan Unit [855]

7.1.2.60.3.4 N_TIME_SPAN_TICKS_PER_HOUR Macro

Number of timer ticks in one hour.

C++

```
#define N_TIME_SPAN_TICKS_PER_HOUR 36000000000LL
```

Type

NTimeSpan Unit [855]

7.1.2.60.3.5 N_TIME_SPAN_TICKS_PER_MILLISECOND Macro

Number of timer ticks in one millisecond.

C++

```
#define N_TIME_SPAN_TICKS_PER_MILLISECOND 10000LL
```

Type

NTimeSpan Unit [855]

7.1.2.60.3.6 N_TIME_SPAN_TICKS_PER_MINUTE Macro

Number of timer ticks in one minute.

C++

```
#define N_TIME_SPAN_TICKS_PER_MINUTE 600000000LL
```

Type

NTimeSpan Unit [855]

7.1.2.60.3.7 N_TIME_SPAN_TICKS_PER_SECOND Macro

Number of timer ticks in one second.

C++

```
#define N_TIME_SPAN_TICKS_PER_SECOND 10000000LL
```

Type

NTimeSpan Unit [855]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.60.3.8 NTimeSpanToString Macro

Picks a Unicode (W) or ASCII (A) version of the NTimeSpanToString function.

C++

```
#define NTimeSpanToString N_FUNC_AW(NTimeSpanToString)
```

Type

NTimeSpan Unit [868]

7.1.2.61 NTls Unit

Represents Thread Local Storage.

Functions

	Name	Description
	NTlsCreate [868]	Creates a new Thread Local Storage.
	NTlsCreateForObject [868]	Creates a new Thread Local Storage for the current object.
	NTlsFree [869]	Release the dynamic memory before the thread returns.
	NTlsGetObject [869]	Retrieves the object associated with the specified Thread Local Storage.
	NTlsGetValue [869]	Retrieves the value of the specified Thread Local Storage.
	NTlsSetObject [870]	Sets the object for the specified Thread Local Storage.
	NTlsSetValue [870]	Sets the Thread Local Storage value.

Variables

Name	Description
pValue [870]	Thread Local Storage value.

7.1.2.61.1 Functions

7.1.2.61.1.1 NTlsCreate Function

Creates a new Thread Local Storage.

C++

```
NResult N_API NTlsCreate(NTlsDisposeProc pDispose, HNTls * pHTls);
```

Parameters

Parameters	Description
NTlsDisposeProc pDispose	[in] Pointer to the NTls [868] dispose procedure.
HNTls * pHTls	[out] Pointer to a handle to the new NTls [868].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTls Unit [868]

7.1.2.61.1.2 NTlsCreateForObject Function

Creates a new Thread Local Storage for the current object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NTlsCreateForObject(HNTls * phTls);
```

Parameters

Parameters	Description
HNTls * phTls	[out] Pointer to a handle to the new NTls [868].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTls Unit [868]

7.1.2.61.1.3 NTlsFree Function

Release the dynamic memory before the thread returns.

C++

```
NResult N_API NTlsFree(HNTls hTls);
```

Parameters

Parameters	Description
HNTls hTls	[in] Handle to the NTls [868].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTls Unit [868]

7.1.2.61.1.4 NTlsGetObject Function

Retrieves the object associated with the specified Thread Local Storage.

C++

```
NResult N_API NTlsGetObject(HNTls hTls, HNObject * phValue);
```

Parameters

Parameters	Description
HNTls hTls	[in] Handle to the NTls [868].
HNObject * phValue	[out] Pointer to a handle to the NObject [536].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTls Unit [868]

7.1.2.61.1.5 NTlsGetValue Function

Retrieves the value of the specified Thread Local Storage.

C++

```
NResult N_API NTlsGetValue(HNTls hTls, void * * ppValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNTls hTls	[in] Handle to the NTls [868].
void * * ppValue	[out] Double pointer to a buffer containing NTls [868] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTls Unit [868]

7.1.2.61.1.6 NTlsSetObject Function

Sets the object for the specified Thread Local Storage.

C++

```
NResult N_API NTlsSetObject(HNTls hTls, HNObject hValue);
```

Parameters

Parameters	Description
HNTls hTls	[in] Handle to the NTls [868].
HNObject hValue	[in] Handle to the NObject [536].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTls Unit [868]

7.1.2.61.1.7 NTlsSetValue Function

Sets the Thread Local Storage value.

C++

```
NResult N_API NTlsSetValue(HNTls hTls, void * pValue);
```

Parameters

Parameters	Description
HNTls hTls	[in] Handle to the NTls [868].
void * pValue	[out] Pointer to a buffer containing the new NTls [868] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTls Unit [868]

7.1.2.61.2 Variables

7.1.2.61.2.1 pValue Variable

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Thread Local Storage value.

C++

```
void * pValue;
```

Module

NTls Unit [868]

7.1.2.62 NType Unit

Manages object types.

Files

Name	Description
NType.h [900]	Header file for the NType module.

Functions

	Name	Description
	NTypeAreValuesEqual [873]	Checks whether the values of the specified type are equal.
	NTypeCompareValues [873]	Compares two values of the specified type and returns an integer that indicates their relative position in the sort order.
	NTypeCopyPropertyValues [874]	Copies properties from one object to another object of the same type.
	NTypeCopyValue [874]	Copies the specified value.
	NTypeDisposeValue [875]	Disposes the specified value.
	NTypeDisposeValues [875]	Disposes the specified value.
	NTypeFreeValues [875]	Frees the specified values.
	NTypeGetBaseType [876]	Gets the base type of the specified type.
	NTypeGetModule [876]	Gets the module of the specified type.
	NTypeGetNameEx [876]	Gets the name of the specified type.
	NTypeGetNameN [877]	Gets the name of the specified type.
	NTypeGetOwnerType [877]	Gets the owner type of the specified type.
	NTypeGetParameterEx [877]	Retrieves the specified parameter value.
	NTypeGetPropertyValueN [878]	Retrieves the specified property value.
	NTypeGetPropertyValueP [878]	Retrieves the specified property value.
	NTypeGetRootType [879]	Retrieves the root type of the specified type.
	NTypeGetRootTypeCode [879]	Retrieves the code of the root type of the specified type.
	NTypeGetTypeCode [879]	Retrieves the type code for the specified type.
	NTypeGetUseInsteadType [880]	Gets the use-instead type of the specified type.
	NTypeGetValueHashCode [880]	Gets the hash code of the specified value.
	NTypeGetValueSize [881]	Gets the size of the specified value type.
	NTypeHasOwnerType [881]	Checks whether the specified type has an owner type.
	NTypeIdentifierToString [881]	Picks a Unicode (W) or ASCII (A) version of the NTypeIdentifierToString function.
	NTypeIdentifierToStringFN [881]	Picks a Unicode (W) or ASCII (A) version of the NTypeIdentifierToStringFN function.
	NTypeIdentifierToStringVN [882]	Picks a Unicode (W) or ASCII (A) version of the NTypeIdentifierToStringVN function.
	NTypeIsAbstract [882]	Checks whether the specified type is abstract.
	NTypeIsAssignableFrom [882]	Checks whether the specified type is abstract from another type.
	NTypeIsBasic [882]	Checks whether the specified value is basic.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NTypeIsCallback [883]	Checks whether the specified value is a callback.
	NTypeIsCloneable [883]	Checks whether the specified type is cloneable.
	NTypeIsComparable [884]	Checks whether the specified type is comparable.
	NTypeIsDeprecated [884]	Checks whether the specified type is deprecated.
	NTypeIsDisposable [884]	Checks whether the specified type is disposable.
	NTypeIsEnum [885]	Checks whether the specified type is an enumeration.
	NTypeIsEquatable [885]	Checks whether the specified type is equatable.
	NTypeIsFlagsEnum [885]	Checks whether the specified value is flag enumerated.
	NTypeIsHandle [886]	Checks whether the specified type is a handle.
	NTypeIsInstanceOfType [886]	Checks whether the specified object is an instance of the specified type.
	NTypeIsInstanceOfTypeP [886]	Checks whether the specified object is an instance of the specified type.
	NTypeIsMemorySerializable [887]	Checks whether the specified type is memory serializable.
	NTypeIsObject [887]	Checks whether the specified type is an object.
	NTypeIsParsable [887]	Checks whether the specified type is parsable.
	NTypeIsPrimitive [888]	Checks whether the specified type is primitive.
	NTypeIsPublic [888]	Checks whether the specified type is public.
	NTypeIsSealed [888]	Checks whether the specified type is sealed.
	NTypeIsSerializable [889]	Checks whether the specified type is serializable.
	NTypeIsSignNeutral [889]	Checks whether the specified type is sign neutral.
	NTypeIsStatic [889]	Checks whether the specified type is static.
	NTypeIsStruct [890]	Checks whether the specified type is a structure.
	NTypeIsSubclassOf [890]	Checks whether the specified type is a subclass of another type.
	NTypeParseValue [890]	Converts the specified string to a value of the specified type.
	NTypeParseValueVN [891]	Converts the specified string to a value of the specified type.
	NTypeReset [891]	Resets the type of the specified object.
	NTypeResetPropertyValue [892]	Resets the type and property of the specified object.
	NTypeSetParameterEx [892]	Sets the value of the specified parameter.
	NTypeSetPropertyValueN [892]	Sets the value of the specified property.
	NTypeSetPropertyValueP [893]	Sets the value of the specified property.
	NTypeSetValue [893]	Sets...
	NTypeTryParseValue [894]	Converts the specified string to a value of the specified type. Returns a Boolean value indicating whether the conversion succeeded.
	NTypeTryParseValueVN [894]	Converts the specified string to a value of the specified type. Returns a Boolean value indicating whether the conversion succeeded.
	NTypeValueToString [895]	Converts the specified value to string.
	NTypeValueToStringN [895]	Converts the specified value to string.

Macros

Name	Description
NTypeCopyParameters [897]	Copies parameters from one object to another object of the same type.
NTypeGetName [898]	Gets the name of the specified type.
NTypeGetNameEx [898]	Picks a Unicode (W) or ASCII (A) version of the NTypeGetNameEx function.
NTypeGetParameter [898]	Retrieves the specified parameter value.
NTypeGetPropertyValueP [898]	Picks a Unicode (W) or ASCII (A) version of the NTypeGetPropertyValueP function.
NTypeParseValue [899]	Picks a Unicode (W) or ASCII (A) version of the NTypeParseValue function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTypeParseValueVN [899]	Picks a Unicode (W) or ASCII (A) version of the NTypeParseValueVN function.
NTypeResetPropertyValueP [899]	Picks a Unicode (W) or ASCII (A) version of the NTypeResetPropertyValue [892] function.
NTypeSetParameter [899]	Sets the value of the specified parameter.
NTypeSetPropertyValueP [900]	Picks a Unicode (W) or ASCII (A) version of the NTypeSetPropertyValueP function.
NTypeTryParseValue [900]	Picks a Unicode (W) or ASCII (A) version of the NTypeTryParseValue function.
NTypeTryParseValueVN [900]	Picks a Unicode (W) or ASCII (A) version of the NTypeTryParseValueVN function.
NTypeValueToString [900]	Picks a Unicode (W) or ASCII (A) version of the NTypeValueToString function.

Structs, Records, Enums

	Name	Description
	NTypeCode_ [896]	Enumerates type codes.
	NTypeCode [896]	Enumerates type codes.

7.1.2.62.1 Functions

7.1.2.62.1.1 NTypeAreValuesEqual Function

Checks whether the values of the specified type are equal.

C++

```
NResult N_API NTypeAreValuesEqual(HNType hType, const void * pValue1, NSizeType value1Size,
const void * pValue2, NSizeType value2Size, NBool * pResult);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
const void * pValue1	[in] Pointer to the first value.
NSizeType value1Size	[in] Size of the first value.
const void * pValue2	[in] Pointer to the second value.
NSizeType value2Size	[in] Size of the second value.
NBool * pResult	[out] NTrue [1064] if the values are equal; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.2 NTypeCompareValues Function

Compares two values of the specified type and returns an integer that indicates their relative position in the sort order.

C++

```
NResult N_API NTypeCompareValues(HNType hType, const void * pValue1, NSizeType value1Size,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
const void * pValue2, NSizeType value2Size, NInt * pResult);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
const void * pValue1	[in] Pointer to the first value.
NSizeType value1Size	[in] Size of the first value.
const void * pValue2	[in] Pointer to the second value.
NSizeType value2Size	[in] Size of the second value.
NInt * pResult	[out] Pointer to an integer that indicates their relative position in the sort order.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.3 NTypeCopyPropertyValues Function

Copies properties from one object to another object of the same type.

C++

```
NResult N_API NTypeCopyPropertyValues(HNType hType, HNObject hDstObject, HNObject
hSrcObject);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
HNObject hDstObject	[in] Handle to the destination object.
HNObject hSrcObject	[in] Handle to the source object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.4 NTypeCopyValue Function

Copies the specified value.

C++

```
NResult N_API NTypeCopyValue(HNType hType, const void * pSrcValue, NSizeType srcValueSize,
void * pDstValue, NSizeType dstValueSize);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type of the value.
const void * pSrcValue	[in] Pointer to the original value.
NSizeType srcValueSize	[in] Size of the original value.
void * pDstValue	[out] Pointer to the copied value.
NSizeType dstValueSize	[in] Size of the copied value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.5 NTypeDisposeValue Function

Disposes the specified value.

C++

```
NResult N_API NTypeDisposeValue(HNType hType, void * pValue, NSizeType valueSize);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the value type.
void * pValue	[in/out] Pointer to the value to be disposed.
NSizeType valueSize	[in] Size of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.6 NTypeDisposeValues Function

Disposes the specified value.

C++

```
NResult N_API NTypeDisposeValues(HNType hType, void * arValues, NSizeType valuesSize, NInt
valuesLength);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the value type.
void * arValues	[in/out] Pointer to the array of values to be disposed.
NSizeType valuesSize	[in] Size of the value.
NInt valuesLength	[in] Length of the value array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.7 NTypeFreeValues Function

Frees the specified values.

C++

```
NResult N_API NTypeFreeValues(HNType hType, void * arValues, NSizeType valuesSize, NInt
valuesLength);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNType hType	[in] Handle to the value type.
void * arValues	[in/out] Pointer to the array of values to be disposed.
NSizeType valuesSize	[in] Size of the value.
NInt valuesLength	[in] Length of the value array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.8 NTypeGetBaseType Function

Gets the base type of the specified type.

C++

```
NResult N_API NTypeGetBaseType(HNType hType, HNType * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the specified type.
HNType * phValue	[out] Pointer to a handle to the base class.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.9 NTypeGetModule Function

Gets the module of the specified type.

C++

```
NResult N_API NTypeGetModule(HNType hType, HNModule * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
HNModule * phValue	[out] Pointer to a handle to the module of the type.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.10 NTypeGetNameEx Function

Gets the name of the specified type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NTypeGetNameEx(HNType hType, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NChar * szValue	[out] Pointer to the name of the type.
NInt valueSize	[in] Size of the name of the type.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.11 NTypeGetNameN Function

Gets the name of the specified type.

C++

```
NResult N_API NTypeGetNameN(HNType hType, HNString * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
HNString * phValue	[out] Pointer to a handle to the name of the type.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.12 NTypeGetOwnerType Function

Gets the owner type of the specified type.

C++

```
NResult N_API NTypeGetOwnerType(HNType hType, HNType * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the specified type.
HNType * phValue	[out] Pointer to a handle to the owner type.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.13 NTypeGetParameterEx Function

Retrieves the specified parameter value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NTypeGetParameterEx(HNType hType, HNObject hObject, NUShort partId, NUInt parameterId, NInt typeId, void * pValue, NSizeType valueSize);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
HNObject hObject	[in] Handle to the object.
NUShort partId	[in] Part ID.
NUInt parameterId	[in] Parameter ID.
NInt typeId	[in] Type ID.
void * pValue	[out] Pointer to the parameter value.
NSizeType valueSize	[in] Value size.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.14 NTypeGetPropertyValueN Function

Retrieves the specified property value.

C++

```
NResult N_API NTypeGetPropertyValueN(HNType hType, HNObject hObject, HNString hName, HNValue * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the object type.
HNObject hObject	[in] Handle to the object.
HNString hName	[in] Handle to the property name.
HNValue * phValue	[out] Pointer to a handle to the property value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.15 NTypeGetPropertyValueP Function

Retrieves the specified property value.

C++

```
NResult N_API NTypeGetPropertyValueP(HNType hType, HNObject hObject, const NChar * szName, NTypeOfProc pValueTypeOf, NAttributes attributes, void * arValues, NSizeType valuesSize, NInt valuesLength, NBool * pHasValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the object type.
HNObject hObject	[in] Handle to the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const NChar * szName	[in] Pointer to the property name.
NTypeOfProc pValueTypeOf	[in] Pointer to a function that retrieves the type of the property.
NAttributes attributes	[in] Property attributes.
void * arValues	[out] Pointer to an array of property values.
NSizeType valuesSize	[in] Size of the property value.
NInt valuesLength	[in] Length of the property value array.
NBool * pHasValue	[out] NTrue [1064] if the value array contains a value; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.16 NTypeGetRootType Function

Retrieves the root type of the specified type.

C++

```
NResult N_API NTypeGetRootType(HNType hType, HNType * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type.
HNType * phValue	[out] Pointer to a handle to the root type.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.17 NTypeGetRootTypeCode Function

Retrieves the code of the root type of the specified type.

C++

```
NResult N_API NTypeGetRootTypeCode(HNType hType, NTypeCode * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type.
NTypeCode * pValue	[out] Pointer to the code of the root type.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.18 NTypeGetTypeCode Function

Retrieves the type code for the specified type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NTypeGetTypeCode(HNType hType, NTypeCode * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type value.
NTypeCode * pValue	[out] Pointer to the code of the type.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.19 NTypeGetUseInsteadType Function

Gets the use-instead type of the specified type.

C++

```
NResult N_API NTypeGetUseInsteadType(HNType hType, HNType * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the specified type.
HNType * phValue	[out] Pointer to a handle to the use-instead class.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.20 NTypeGetValueHashCode Function

Gets the hash code of the specified value.

C++

```
NResult N_API NTypeGetValueHashCode(HNType hType, const void * pValue, NSizeType valueSize,
NInt * pResult);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
const void * pValue	[in] Pointer to the value.
NSizeType valueSize	[in] Value size.
NInt * pResult	[out] Pointer to the hash code.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.62.1.21 NTypeGetValueSize Function

Gets the size of the specified value type.

C++

```
NResult N_API NTypeGetValueSize(HNType hType, NSizeType * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NSizeType * pValue	[out] Pointer to the size of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.22 NTypeHasOwnerType Function

Checks whether the specified type has an owner type.

C++

```
NResult N_API NTypeHasOwnerType(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the type has an owner type; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.23 NTypeIdentifierToString Function

Picks a Unicode (W) or ASCII (A) version of the NTypeIdentifierToString function.

C++

```
NResult N_API NTypeIdentifierToString(const NChar * szValue, const NChar * szFormat,
HNString * phValue);
```

Module

NType Unit [871]

7.1.2.62.1.24 NTypeIdentifierToStringFN Function

Picks a Unicode (W) or ASCII (A) version of the NTypeIdentifierToStringFN function.

C++

```
NResult N_API NTypeIdentifierToStringFN(const NChar * szValue, HNString hFormat, HNString *
phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NType Unit [871]

7.1.2.62.1.25 NTypeIdentifierToStringVN Function

Picks a Unicode (W) or ASCII (A) version of the NTypeIdentifierToStringVN function.

C++

```
NResult N_API NTypeIdentifierToStringVN(HNString hValue, const NChar * szFormat, HNString *
phValue);
```

Module

NType Unit [871]

7.1.2.62.1.26 NTypeIsAbstract Function

Checks whether the specified type is abstract.

C++

```
NResult N_API NTypeIsAbstract(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the type is abstract; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.27 NTypeIsAssignableFrom Function

Checks whether the specified type is abstract from another type.

C++

```
NResult N_API NTypeIsAssignableFrom(HNType hType, HNType hOtherType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
HNType hOtherType	[in] Handle to the other NType [871].
NBool * pValue	[out] NTrue [1064] if the first type is assignable from the second type; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.28 NTypeIsBasic Function

Checks whether the specified value is basic.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NTypeIsBasic(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the value is basic; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.29 NTypeIsCallback Function

Checks whether the specified value is a callback.

C++

```
NResult N_API NTypeIsCallback(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the value is a callback; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.30 NTypeIsCloneable Function

Checks whether the specified type is cloneable.

C++

```
NResult N_API NTypeIsCloneable(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the type is cloneable; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.62.1.31 NTypeIsComparable Function

Checks whether the specified type is comparable.

C++

```
NResult N_API NTypeIsComparable(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the type is comparable; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.32 NTypeIsDeprecated Function

Checks whether the specified type is deprecated.

C++

```
NResult N_API NTypeIsDeprecated(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the type is deprecated; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.33 NTypeIsDisposable Function

Checks whether the specified type is disposable.

C++

```
NResult N_API NTypeIsDisposable(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type value.
NBool * pValue	[out] NTrue [1064] if the value is disposable; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.62.1.34 NTypeIsEnum Function

Checks whether the specified type is an enumeration.

C++

```
NResult N_API NTypeIsEnum(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type value.
NBool * pValue	[out] NTrue [1064] if the value is an enumeration; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.35 NTypeIsEquatable Function

Checks whether the specified type is equatable.

C++

```
NResult N_API NTypeIsEquatable(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the type is equatable; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.36 NTypeIsFlagsEnum Function

Checks whether the specified value is flag enumerated.

C++

```
NResult N_API NTypeIsFlagsEnum(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the value is flag enumerated; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.62.1.37 NTypeIsHandle Function

Checks whether the specified type is a handle.

C++

```
NResult N_API NTypeIsHandle(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type value.
NBool * pValue	[out] NTrue [1064] if the value is a handle; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.38 NTypeIsInstanceOfType Function

Checks whether the specified object is an instance of the specified type.

C++

```
NResult N_API NTypeIsInstanceOfType(HNType hType, HNObject hObject, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
HNObject hObject	[in] Handle to the object.
NBool * pValue	[out] NTrue [1064] if the object is an instance of the type; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.39 NTypeIsInstanceOfTypeP Function

Checks whether the specified object is an instance of the specified type.

C++

```
NResult N_API NTypeIsInstanceOfTypeP(NTypeOfProc pTypeOf, HNObject hObject, NBool * pValue);
```

Parameters

Parameters	Description
NTypeOfProc pTypeOf	[in] Pointer to a function that retrieves the NType [871].
HNObject hObject	[in] Handle to the object.
NBool * pValue	[out] NTrue [1064] if the object is an instance of the type; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NType Unit [871]

7.1.2.62.1.40 NTypeIsMemorySerializable Function

Checks whether the specified type is memory serializable.

C++

```
NResult N_API NTypeIsMemorySerializable(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the type is memory serializable; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.41 NTypeIsObject Function

Checks whether the specified type is an object.

C++

```
NResult N_API NTypeIsObject(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type value.
NBool * pValue	[out] NTrue [1064] if the value is an object; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.42 NTypeIsParsable Function

Checks whether the specified type is parsable.

C++

```
NResult N_API NTypeIsParsable(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the type is parsable; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.43 NTypeIsPrimitive Function

Checks whether the specified type is primitive.

C++

```
NResult N_API NTypeIsPrimitive(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type value.
NBool * pValue	[out] NTrue [1064] if the value is primitive; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.44 NTypeIsPublic Function

Checks whether the specified type is public.

C++

```
NResult N_API NTypeIsPublic(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the type is public; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.45 NTypeIsSealed Function

Checks whether the specified type is sealed.

C++

```
NResult N_API NTypeIsSealed(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the type is sealed; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.46 NTypeIsSerializable Function

Checks whether the specified type is serializable.

C++

```
NResult N_API NTypeIsSerializable(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the type is serializable; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.47 NTypeIsSignNeutral Function

Checks whether the specified type is sign neutral.

C++

```
NResult N_API NTypeIsSignNeutral(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type value.
NBool * pValue	[out] NTrue [1064] if the type is sign neutral; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.48 NTypeIsStatic Function

Checks whether the specified type is static.

C++

```
NResult N_API NTypeIsStatic(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
NBool * pValue	[out] NTrue [1064] if the type is static; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.49 NTypeIsStruct Function

Checks whether the specified type is a structure.

C++

```
NResult N_API NTypeIsStruct(HNType hType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type value.
NBool * pValue	[out] NTrue [1064] if the value is a structure; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.50 NTypeIsSubclassOf Function

Checks whether the specified type is a subclass of another type.

C++

```
NResult N_API NTypeIsSubclassOf(HNType hType, HNType hOtherType, NBool * pValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the first NType [871].
HNType hOtherType	[in] Handle to the second NType [871].
NBool * pValue	[out] NTrue [1064] if the first type is subclass of the second type; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.51 NTypeParseValue Function

Converts the specified string to a value of the specified type.

C++

```
NResult N_API NTypeParseValue(HNType hType, const NChar * szValue, const NChar * szFormat,
void * pValue, NSizeType valueSize);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
const NChar * szValue	[in] Pointer to the string to parse.
const NChar * szFormat	[in] Pointer to format string.
void * pValue	[out] Pointer to the parsed value.
NSizeType valueSize	[in] Value size.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NType Unit [871]

7.1.2.62.1.52 NTypeParseValueVN Function

Converts the specified string to a value of the specified type.

C++

```
NResult N_API NTypeParseValueVN(HNType hType, HNString hValue, const NChar * szFormat, void
* pValue, NSizeType valueSize);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
HNString hValue	[in] Handle to the string to parse.
const NChar * szFormat	[in] Pointer to format string.
void * pValue	[out] Pointer to the parsed value.
NSizeType valueSize	[in] Value size.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NType Unit [871]

7.1.2.62.1.53 NTypeReset Function

Resets the type of the specified object.

C++

```
NResult N_API NTypeReset(HNType hType, HNObject hObject);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
HNObject hObject	[in] Handle to the NObject [536].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.54 NTypeResetPropertyValue Function

Resets the type and property of the specified object.

C++

```
NResult N_API NTypeResetPropertyValue(HNType hType, HNObject hObject, const NChar * szName);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
HNObject hObject	[in] Handle to the NObject [536].
const NChar * szName	[in] Pointer to the name of the property to be reset.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.55 NTypeSetParameterEx Function

Sets the value of the specified parameter.

C++

```
NResult N_API NTypeSetParameterEx(HNType hType, HNObject hObject, NUShort partId, NUInt
parameterId, NInt typeId, const void * pValue, NSizeType valueSize);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
HNObject hObject	[in] Handle to the object.
NUShort partId	[in] Part ID.
NUInt parameterId	[in] Parameter ID.
NInt typeId	[in] Type ID.
const void * pValue	[in] Pointer to the new parameter value.
NSizeType valueSize	[in] Value size.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.56 NTypeSetPropertyValueN Function

Sets the value of the specified property.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NTypeSetPropertyValueN(HNType hType, HNObject hObject, HNString hName,
HNValue hValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type of the object.
HNObject hObject	[in] Handle to the object.
HNString hName	[in] Handle to the name of the property.
HNValue hValue	[in] Handle to the new value of the property.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.57 NTypeSetPropertyValueP Function

Sets the value of the specified property.

C++

```
NResult N_API NTypeSetPropertyValueP(HNType hType, HNObject hObject, const NChar * szName,
NTypeOfProc pValueTypeOf, NAttributes attributes, const void * arValues, NSizeType
valuesSize, NInt valuesLength, NBool hasValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type of the object.
HNObject hObject	[in] Handle to the object.
const NChar * szName	[in] Pointer to the name of the property.
NTypeOfProc pValueTypeOf	[in] Pointer to a function that retrieves the type of the property.
NAttributes attributes	[in] Property attributes.
const void * arValues	[in] Pointer to an array of property values.
NSizeType valuesSize	[in] Size of the property value.
NInt valuesLength	[in] Length of the property value array.
NBool hasValue	[in] NTrue [1064] if the value array contains a value; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.58 NTypeSetValue Function

Sets...

C++

```
NResult N_API NTypeSetValue(HNType hType, const void * pSrcValue, NSizeType srcValueSize,
void * pDstValue, NSizeType dstValueSize);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNType hType	[in] Handle to the value type.
const void * pSrcValue	[in] Pointer to the source value.
NSizeType srcValueSize	[in] Size of the source value.
void * pDstValue	[out] Pointer to the destination value.
NSizeType dstValueSize	[in] Size of the destination value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.1.59 NTypeTryParseValue Function

Converts the specified string to a value of the specified type. Returns a Boolean value indicating whether the conversion succeeded.

C++

```
NResult N_API NTypeTryParseValue(HNType hType, const NChar * szValue, const NChar *
szFormat, void * pValue, NSizeType valueSize, NBool * pResult);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
const NChar * szValue	[in] Pointer to the string to parse.
const NChar * szFormat	[in] Pointer to format string.
void * pValue	[out] Pointer to the parsed value.
NSizeType valueSize	[in] Value size.
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NType Unit [871]

7.1.2.62.1.60 NTypeTryParseValueVN Function

Converts the specified string to a value of the specified type. Returns a Boolean value indicating whether the conversion succeeded.

C++

```
NResult N_API NTypeTryParseValueVN(HNType hType, HNString hValue, const NChar * szFormat,
void * pValue, NSizeType valueSize, NBool * pResult);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNString hValue	[in] Handle to the string to parse.
const NChar * szFormat	[in] Pointer to format string.
void * pValue	[out] Pointer to the parsed value.
NSizeType valueSize	[in] Value size.
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NType Unit [871]

7.1.2.62.1.61 NTypeValueToString Function

Converts the specified value to string.

C++

```
NResult N_API NTypeValueToString(HNType hType, const void * pValue, NSizeType valueSize,
const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
const void * pValue	[in] Pointer to the value to convert.
NSizeType valueSize	[in] Value size.
const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] Pointer to a handle to the converted string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NType Unit [871]

7.1.2.62.1.62 NTypeValueToStringN Function

Converts the specified value to string.

C++

```
NResult N_API NTypeValueToStringN(HNType hType, const void * pValue, NSizeType valueSize,
HNString hFormat, HNString * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the NType [871].
const void * pValue	[in] Pointer to the value to convert.
NSizeType valueSize	[in] Value size.
HNString hFormat	[in] Handle to format string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNString * phValue	[out] Pointer to a handle to the converted string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NType Unit [871]

7.1.2.62.2 Structs, Records, Enums

7.1.2.62.2.1 NTypeCode Enumeration

Enumerates type codes.

C++

```
typedef enum NTypeCode_ {
  ntcNone = 0,
  ntcOther = 1,
  ntcByte = 2,
  ntcSByte = 3,
  ntcUInt16 = 4,
  ntcInt16 = 5,
  ntcUInt32 = 6,
  ntcInt32 = 7,
  ntcUInt64 = 8,
  ntcInt64 = 9,
  ntcSingle = 10,
  ntcDouble = 11,
  ntcBoolean = 12,
  ntcSizeType = 13,
  ntcSSizeType = 14,
  ntcPointer = 15,
  ntcResult = 16,
  ntcAChar = 17,
  ntcWChar = 18,
  ntcChar = ntcAChar,
  ntcString = 19,
  ntcObject = 20,
  ntcValue = 21,
  ntcArray = 22,
  ntcTimeSpan = 23,
  ntcDateTime = 24,
  ntcURational = 25,
  ntcRational = 26,
  ntcComplex = 27,
  ntcGuid = 28,
  ntcBuffer = 29
} NTypeCode;
```

Members

Members	Description
ntcNone = 0	Empty value.
ntcOther = 1	Other value type.
ntcByte = 2	Byte.
ntcSByte = 3	Signed byte.
ntcUInt16 = 4	Unsigned short integer value.
ntcInt16 = 5	Short integer value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

ntcUInt32 = 6	Unsigned integer value.
ntcInt32 = 7	Integer value.
ntcUInt64 = 8	Unsigned long integer value.
ntcInt64 = 9	Long integer value.
ntcSingle = 10	Single value.
ntcDouble = 11	Double value.
ntcBoolean = 12	Boolean value.
ntcSizeType = 13	NSizeType [1008] value.
ntcSSizeType = 14	NSSizeType [1009] value.
ntcPointer = 15	A pointer.
ntcResult = 16	NResult [1007] value.
ntcAChar = 17	ASCII character.
ntcWChar = 18	Unicode character.
ntcChar = ntcAChar	Character.
ntcString = 19	String value.
ntcObject = 20	An object.
ntcValue = 21	Custom type value.
ntcArray = 22	Array.
ntcTimeSpan = 23	NTimeSpan [855] value.
ntcDateTime = 24	Date-time value.
ntcURational = 25	Unsigned rational number value.
ntcRational = 26	A rational number value.
ntcComplex = 27	Complex number.
ntcGuid = 28	Global unique identifier.
ntcBuffer = 29	Buffer.

Module

NType Unit [871]

7.1.2.62.3 Macros

7.1.2.62.3.1 NTypeCopyParameters Macro

Copies parameters from one object to another object of the same type.

C++

```
#define NTypeCopyParameters(hType, hDstObject, hSrcObject) NTypeCopyPropertyValues(hType,
hDstObject, hSrcObject)
```

Parameters

Parameters	Description
hType	[in] Handle to the NType [871].
hDstObject	[in] Handle to the destination object.
hSrcObject	[in] Handle to the source object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.62.3.2 NTypeGetName Macro

Gets the name of the specified type.

C++

```
#define NTypeGetName(hType, pValue) NTypeGetNameEx(hType, pValue, N_INT_MAX)
```

Parameters

Parameters	Description
hType	[in] Handle to the NType [871].
pValue	[out] Pointer to the name of the type.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.3.3 NTypeGetNameEx Macro

Picks a Unicode (W) or ASCII (A) version of the NTypeGetNameEx function.

C++

```
#define NTypeGetNameEx N_FUNC_AW(NTypeGetNameEx)
```

Module

NType Unit [871]

7.1.2.62.3.4 NTypeGetParameter Macro

Retrieves the specified parameter value.

C++

```
#define NTypeGetParameter(hType, hObject, partId, parameterId, pValue)
NTypeGetParameterEx(hType, hObject, partId, parameterId, -1, pValue, N_SIZE_TYPE_MAX)
```

Parameters

Parameters	Description
hType	[in] Handle to the NType [871].
hObject	[in] Handle to the object.
partId	[in] Part ID.
parameterId	[in] Parameter ID.
pValue	[out] Pointer to the parameter value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

7.1.2.62.3.5 NTypeGetPropertyValueP Macro

Picks a Unicode (W) or ASCII (A) version of the NTypeGetPropertyValueP function.

C++

```
#define NTypeGetPropertyValueP N_FUNC_AW(NTypeGetPropertyValueP)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NType Unit [871]

7.1.2.62.3.6 NTypeParseValue Macro

Picks a Unicode (W) or ASCII (A) version of the NTypeParseValue function.

C++

```
#define NTypeParseValue N_FUNC_AW(NTypeParseValue)
```

Module

NType Unit [871]

7.1.2.62.3.7 NTypeParseValueVN Macro

Picks a Unicode (W) or ASCII (A) version of the NTypeParseValueVN function.

C++

```
#define NTypeParseValueVN N_FUNC_AW(NTypeParseValueVN)
```

Module

NType Unit [871]

7.1.2.62.3.8 NTypeResetPropertyValueP Macro

Picks a Unicode (W) or ASCII (A) version of the NTypeResetPropertyValue [892] function.

C++

```
#define NTypeResetPropertyValueP N_FUNC_AW(NTypeResetPropertyValue)
```

Module

NType Unit [871]

7.1.2.62.3.9 NTypeSetParameter Macro

Sets the value of the specified parameter.

C++

```
#define NTypeSetParameter(hType, hObject, partId, parameterId, pValue)
NTypeSetParameterEx(hType, hObject, partId, parameterId, -1, pValue, N_SIZE_TYPE_MAX)
```

Parameters

Parameters	Description
hType	[in] Handle to the NType [871].
hObject	[in] Handle to the object.
partId	[in] Part ID.
parameterId	[in] Parameter ID.
pValue	[in] Pointer to the new parameter value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NType Unit [871]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.62.3.10 NTypeSetPropertyValueP Macro

Picks a Unicode (W) or ASCII (A) version of the NTypeSetPropertyValueP function.

C++

```
#define NTypeSetPropertyValueP N_FUNC_AW(NTypeSetPropertyValueP)
```

Module

NType Unit [871]

7.1.2.62.3.11 NTypeTryParseValue Macro

Picks a Unicode (W) or ASCII (A) version of the NTypeTryParseValue function.

C++

```
#define NTypeTryParseValue N_FUNC_AW(NTypeTryParseValue)
```

Module

NType Unit [871]

7.1.2.62.3.12 NTypeTryParseValueVN Macro

Picks a Unicode (W) or ASCII (A) version of the NTypeTryParseValueVN function.

C++

```
#define NTypeTryParseValueVN N_FUNC_AW(NTypeTryParseValueVN)
```

Module

NType Unit [871]

7.1.2.62.3.13 NTypeValueToString Macro

Picks a Unicode (W) or ASCII (A) version of the NTypeValueToString function.

C++

```
#define NTypeValueToString N_FUNC_AW(NTypeValueToString)
```

Module

NType Unit [871]

7.1.2.62.4 Files

7.1.2.62.4.1 NType.h

Header file for the NType [871] module.

Enumerations

	Name	Description
	NTypeCode_ [896]	Enumerates type codes.
	NTypeCode [896]	Enumerates type codes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NTypeAreValuesEqual [873]	Checks whether the values of the specified type are equal.
	NTypeCompareValues [873]	Compares two values of the specified type and returns an integer that indicates their relative position in the sort order.
	NTypeCopyPropertyValues [874]	Copies properties from one object to another object of the same type.
	NTypeCopyValue [874]	Copies the specified value.
	NTypeDisposeValue [875]	Disposes the specified value.
	NTypeDisposeValues [875]	Disposes the specified value.
	NTypeFreeValues [875]	Frees the specified values.
	NTypeGetBaseType [876]	Gets the base type of the specified type.
	NTypeGetModule [876]	Gets the module of the specified type.
	NTypeGetNameEx [876]	Gets the name of the specified type.
	NTypeGetNameN [877]	Gets the name of the specified type.
	NTypeGetOwnerType [877]	Gets the owner type of the specified type.
	NTypeGetParameterEx [877]	Retrieves the specified parameter value.
	NTypeGetPropertyValueN [878]	Retrieves the specified property value.
	NTypeGetPropertyValueP [878]	Retrieves the specified property value.
	NTypeGetRootType [879]	Retrieves the root type of the specified type.
	NTypeGetRootTypeCode [879]	Retrieves the code of the root type of the specified type.
	NTypeGetTypeCode [879]	Retrieves the type code for the specified type.
	NTypeGetUseInsteadType [880]	Gets the use-instead type of the specified type.
	NTypeGetValueHashCode [880]	Gets the hash code of the specified value.
	NTypeGetValueSize [881]	Gets the size of the specified value type.
	NTypeHasOwnerType [881]	Checks whether the specified type has an owner type.
	NTypeIdentifierToString [881]	Picks a Unicode (W) or ASCII (A) version of the NTypeIdentifierToString function.
	NTypeIdentifierToStringFN [881]	Picks a Unicode (W) or ASCII (A) version of the NTypeIdentifierToStringFN function.
	NTypeIdentifierToStringVN [882]	Picks a Unicode (W) or ASCII (A) version of the NTypeIdentifierToStringVN function.
	NTypeIsAbstract [882]	Checks whether the specified type is abstract.
	NTypeIsAssignableFrom [882]	Checks whether the specified type is abstract from another type.
	NTypeIsBasic [882]	Checks whether the specified value is basic.
	NTypeIsCallback [883]	Checks whether the specified value is a callback.
	NTypeIsCloneable [883]	Checks whether the specified type is cloneable.
	NTypeIsComparable [884]	Checks whether the specified type is comparable.
	NTypeIsDeprecated [884]	Checks whether the specified type is deprecated.
	NTypeIsDisposable [884]	Checks whether the specified type is disposable.
	NTypeIsEnum [885]	Checks whether the specified type is an enumeration.
	NTypeIsEquatable [885]	Checks whether the specified type is equatable.
	NTypeIsFlagsEnum [885]	Checks whether the specified value is flag enumerated.
	NTypeIsHandle [886]	Checks whether the specified type is a handle.
	NTypeIsInstanceOfType [886]	Checks whether the specified object is an instance of the specified type.
	NTypeIsInstanceOfTypeP [886]	Checks whether the specified object is an instance of the specified type.
	NTypeIsMemorySerializable [887]	Checks whether the specified type is memory serializable.
	NTypeIsObject [887]	Checks whether the specified type is an object.
	NTypeIsParsable [887]	Checks whether the specified type is parsable.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NTypeIsPrimitive [888]	Checks whether the specified type is primitive.
	NTypeIsPublic [888]	Checks whether the specified type is public.
	NTypeIsSealed [888]	Checks whether the specified type is sealed.
	NTypeIsSerializable [889]	Checks whether the specified type is serializable.
	NTypeIsSignNeutral [889]	Checks whether the specified type is sign neutral.
	NTypeIsStatic [889]	Checks whether the specified type is static.
	NTypeIsStruct [890]	Checks whether the specified type is a structure.
	NTypeIsSubclassOf [890]	Checks whether the specified type is a subclass of another type.
	NTypeParseValue [890]	Converts the specified string to a value of the specified type.
	NTypeParseValueVN [891]	Converts the specified string to a value of the specified type.
	NTypeReset [891]	Resets the type of the specified object.
	NTypeResetPropertyValue [892]	Resets the type and property of the specified object.
	NTypeSetParameterEx [892]	Sets the value of the specified parameter.
	NTypeSetPropertyValueN [892]	Sets the value of the specified property.
	NTypeSetPropertyValueP [893]	Sets the value of the specified property.
	NTypeSetValue [893]	Sets...
	NTypeTryParseValue [894]	Converts the specified string to a value of the specified type. Returns a Boolean value indicating whether the conversion succeeded.
	NTypeTryParseValueVN [894]	Converts the specified string to a value of the specified type. Returns a Boolean value indicating whether the conversion succeeded.
	NTypeValueToString [895]	Converts the specified value to string.
	NTypeValueToStringN [895]	Converts the specified value to string.

Macros

Name	Description
NTypeCopyParameters [897]	Copies parameters from one object to another object of the same type.
NTypeGetName [898]	Gets the name of the specified type.
NTypeGetNameEx [898]	Picks a Unicode (W) or ASCII (A) version of the NTypeGetNameEx function.
NTypeGetParameter [898]	Retrieves the specified parameter value.
NTypeGetPropertyValueP [898]	Picks a Unicode (W) or ASCII (A) version of the NTypeGetPropertyValueP function.
NTypeParseValue [899]	Picks a Unicode (W) or ASCII (A) version of the NTypeParseValue function.
NTypeParseValueVN [899]	Picks a Unicode (W) or ASCII (A) version of the NTypeParseValueVN function.
NTypeResetPropertyValueP [899]	Picks a Unicode (W) or ASCII (A) version of the NTypeResetPropertyValue [892] function.
NTypeSetParameter [899]	Sets the value of the specified parameter.
NTypeSetPropertyValueP [900]	Picks a Unicode (W) or ASCII (A) version of the NTypeSetPropertyValueP function.
NTypeTryParseValue [900]	Picks a Unicode (W) or ASCII (A) version of the NTypeTryParseValue function.
NTypeTryParseValueVN [900]	Picks a Unicode (W) or ASCII (A) version of the NTypeTryParseValueVN function.
NTypeValueToString [900]	Picks a Unicode (W) or ASCII (A) version of the NTypeValueToString function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NType Unit [871]

7.1.2.63 NTypeDescriptor Unit

Provides information about the characteristics for a component, such as its attributes, properties, and events.

Files

Name	Description
NTypeDescriptor.h [905]	Header file for the NTypeDescriptor module.

Functions

	Name	Description
	NTypeDescriptorGetDefaultPropertyName [903]	Retrieves the default property name of the specified object.
	NTypeDescriptorGetDefaultPropertyNameForType [903]	Retrieves the default property name of the specified type.
	NTypeDescriptorGetProperties [904]	Retrieves the properties of the specified object.
	NTypeDescriptorGetPropertiesForType [904]	Retrieves the properties of the specified type.

7.1.2.63.1 Functions

7.1.2.63.1.1 NTypeDescriptorGetDefaultPropertyName Function

Retrieves the default property name of the specified object.

C++

```
NResult N_API NTypeDescriptorGetDefaultPropertyName(HNObject hObject, HNString * phValue);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the object.
HNString * phValue	[out] Pointer to a handle to the string with default property name.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypeDescriptor Unit [903]

7.1.2.63.1.2 NTypeDescriptorGetDefaultPropertyNameForType Function

Retrieves the default property name of the specified type.

C++

```
NResult N_API NTypeDescriptorGetDefaultPropertyNameForType(HNType hType, HNString *
phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNString * phValue	[out] Pointer to a handle to the string with default property name.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypeDescriptor Unit [903]

7.1.2.63.1.3 NTypeDescriptorGetProperties Function

Retrieves the properties of the specified object.

C++

```
NResult N_API NTypeDescriptorGetProperties(HNObject hObject, HNPropertyDescriptor * *
parhProperties, NInt * pPropertyCount);
```

Parameters

Parameters	Description
HNObject hObject	[in] Handle to the object.
HNPropertyDescriptor * * parhProperties	[out] Pointer to an array containing the handles to the properties of the object.
NInt * pPropertyCount	[out] Pointer to the number of property handles in an array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypeDescriptor Unit [903]

7.1.2.63.1.4 NTypeDescriptorGetPropertiesForType Function

Retrieves the properties of the specified type.

C++

```
NResult N_API NTypeDescriptorGetPropertiesForType(HNType hType, HNPropertyDescriptor * *
parhProperties, NInt * pPropertyCount);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type.
HNPropertyDescriptor * * parhProperties	[out] Pointer to an array containing the handles to the properties of the type.
NInt * pPropertyCount	[out] Pointer to the number of property handles in an array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypeDescriptor Unit [903]

7.1.2.63.2 Files

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.63.2.1 NTypeDescriptor.h

Header file for the NTypeDescriptor [903] module.

Functions

	Name	Description
	NTypeDescriptorGetDefaultPropertyName [903]	Retrieves the default property name of the specified object.
	NTypeDescriptorGetDefaultPropertyNameForType [903]	Retrieves the default property name of the specified type.
	NTypeDescriptorGetProperties [904]	Retrieves the properties of the specified object.
	NTypeDescriptorGetPropertiesForType [904]	Retrieves the properties of the specified type.

Module

NTypeDescriptor Unit [903]

7.1.2.64 NTypes Unit

Defines types and macros used in Neurotechnology components.

Files

Name	Description
NTypes.h [1078]	Header file for NTypes module. Defines types and macros used in Neurotechnology components.

Functions

	Name	Description
	NACharParseN [920]	Converts a NString [678] to NAChar [1004].
	NACharParseStrOrChars [921]	Converts an array of NChars to NAChar [1004].
	NACharParseVN [921]	Converts a NString [678] to NAChar [1004].
	NACharToChars [921]	Converts NAChar [1004] to an array of NAChars.
	NACharToString [922]	Converts NAChar [1004] to NString [678]
	NACharTryParseN [922]	Converts NString [678] to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.
	NACharTryParseStrOrChars [923]	Converts an array of NChars to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.
	NACharTryParseVN [923]	Converts NString [678] to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.
	NBooleanParseN [924]	Converts NString [678] to NBoolean [1004].
	NBooleanParseStrOrChars [924]	Converts an array of NChars to NBoolean [1004].
	NBooleanParseVN [924]	Converts NString [678] to NBoolean [1004].
	NBooleanToChars [925]	Converts NBoolean [1004] to an array of NChars.
	NBooleanToString [925]	Converts NBoolean [1004] to NString [678].
	NBooleanTryParseN [926]	Converts a NString [678] to NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.
	NBooleanTryParseStrOrChars [926]	Converts an array of NStrings or NChars into NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.
	NBooleanTryParseVN [927]	Converts a NString [678] to NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.
	NCharFromBinDigit [927]	Converts a binary digit to NChar [1005].
	NCharFromDigit [927]	Converts a digit to NChar [1005].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NCharFromHexDigit [928]	Converts a hexadecimal digit to NChar [1005].
	NCharFromOctDigit [928]	Converts an octal digit to NChar [1005].
	NCharIsAscii [928]	Checks whether the specified char is ASCII.
	NCharIsBinDigit [929]	Checks whether the specified character is a binary digit.
	NCharIsDigit [929]	Checks whether the specified character is a digit.
	NCharIsHexDigit [929]	Checks whether the specified character is a hexadecimal digit.
	NCharIsLetter [930]	Checks whether the specified character is a letter.
	NCharIsLower [930]	Checks whether the specified character is a lower value.
	NCharIsOctDigit [930]	Checks whether the specified character is an octal digit.
	NCharIsUpper [930]	Checks whether the specified character is an upper value.
	NCharIsWhiteSpace [931]	Checks whether the specified character is a whitespace.
	NCharParseN [931]	Converts a NString [678] to a NChar [1005].
	NCharParseStrOrChars [931]	Converts an array of NStrings or NChars into NChar [1005].
	NCharParseVN [932]	Converts a NString [678] to a NChar [1005].
	NCharToBinDigit [932]	Converts a NChar [1005] to a binary digit.
	NCharToChars [933]	Converts a NChar [1005] into an array of NChars.
	NCharToDigit [933]	Converts a NChar [1005] to a digit.
	NCharToHexDigit [933]	Converts a NChar [1005] to a hexadecimal digit.
	NCharToLower [934]	Converts a NChar [1005] to lowercase.
	NCharToOctDigit [934]	Converts a NChar [1005] to an octal digit.
	NCharToString [934]	Converts NChar [1005] to NString [678].
	NCharToUpper [935]	Converts NChar [1005] to uppercase.
	NCharTryParseN [935]	Converts a NString [678] to a NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.
	NCharTryParseStrOrChars [935]	Converts an array of NChars into NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.
	NCharTryParseVN [936]	Converts a NString [678] to a NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.
	NComplexToString [936]	Converts NComplex [1005] to NString [678].
	NDoubleGetNaN [937]	Retrieves a NDouble [1005] that is Not a Number (NaN).
	NDoubleGetNegativeInfinity [937]	Retrieves a NDouble [1005] that is NegativeInfinity.
	NDoubleGetPositiveInfinity [937]	Retrieves a NDouble [1005] that is PositiveInfinity.
	NDoubleIsInfinity [937]	Checks if NDouble [1005] evaluates to NegativeInfinity or PositiveInfinity.
	NDoubleIsNaN [938]	Checks if NDouble [1005] is Not a Number (NaN).
	NDoubleIsNegativeInfinity [938]	Checks if NDouble [1005] evaluates to NegativeInfinity.
	NDoubleIsPositiveInfinity [938]	Checks if NDouble [1005] evaluates to PositiveInfinity
	NDoubleParseN [939]	Converts a NString [678] into NDouble [1005].
	NDoubleParseStrOrChars [939]	Converted an array of NChars to NDouble [1005].
	NDoubleParseVN [939]	Converts a NString [678] into NDouble [1005].
	NDoubleToChars [940]	Converts NDouble [1005] to an array of NChars.
	NDoubleToString [940]	Converts NDouble [1005] to NString [678].
	NDoubleTryParseN [941]	Converts a NString [678] to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.
	NDoubleTryParseStrOrChars [941]	Converts an array of NChars to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.
	NDoubleTryParseVN [942]	Converts a NString [678] to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.
	NGuidParse [942]	Converts a NChar [1005] to NGuid.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NGuidParseVN [943]	Converts a NString [678] to NGuid.
	NGuidToString [943]	Converts a NGuid to NString [678].
	NGuidTryParse [943]	Converts a NChar [1005] to NGuid. A NBool [1004] value indicates whether the conversion succeeded.
	NGuidTryParseVN [944]	Converts a NString [678] to NGuid. A NBool [1004] value indicates whether the conversion succeeded.
	NIndexPairToString [944]	Convertes NIndexPair [1006] to NString [678].
	NInt16ParseN [945]	Convertes NString [678] to NInt16 [1006].
	NInt16ParseStrOrChars [945]	Converts an array of NChars to NInt16 [1006].
	NInt16ParseVN [946]	Convertes NString [678] to NInt16 [1006].
	NInt16ToChars [946]	Converts NInt16 [1006] to an array of NChars.
	NInt16ToString [946]	Converts NInt16 [1006] to NString [678].
	NInt16TryParseN [947]	Converts NString [678] to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt16TryParseStrOrChars [947]	Converts an array of NChars to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt16TryParseVN [948]	Converts NString [678] to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt32ParseN [948]	Converts NString [678] to NInt32 [1006].
	NInt32ParseStrOrChars [949]	Converts an array of NChars to NInt32 [1006].
	NInt32ParseVN [949]	Converts NString [678] to NInt32 [1006].
	NInt32ToChars [949]	Converts NInt32 [1006] to an array of NChars.
	NInt32ToString [950]	Converts NInt32 [1006] to NString [678].
	NInt32TryParseN [950]	Converts NString [678] to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt32TryParseStrOrChars [951]	Converts an array of NChars to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt32TryParseVN [951]	Converts NString [678] to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt64ParseN [952]	Converts NString [678] to NInt64 [1006].
	NInt64ParseStrOrChars [952]	Converts an array of NChars to NInt64 [1006].
	NInt64ParseVN [952]	Converts NString [678] to NInt64 [1006].
	NInt64ToChars [953]	Converts NInt64 [1006] to an array of NChars.
	NInt64ToString [953]	Converts NInt64 [1006] to NString [678].
	NInt64TryParseN [954]	Converts NString [678] to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt64TryParseStrOrChars [954]	Converts an array of NChars to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt64TryParseVN [955]	Converts NString [678] to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt8ParseN [955]	Converts NString [678] to NInt8 [1007].
	NInt8ParseStrOrChars [956]	Converts an array of NChars to NInt8 [1007].
	NInt8ParseVN [956]	Converts NString [678] to NInt8 [1007].
	NInt8ToChars [956]	Converts NInt8 [1007] to an array of NChars.
	NInt8ToString [957]	Converts NInt8 [1007] to NString [678].
	NInt8TryParseN [957]	Converts NString [678] to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt8TryParseStrOrChars [958]	Converts an array of NChars to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt8TryParseVN [958]	Converts NString [678] to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NNameStringPairCopy [959]	Creates a copy of string pair structure.
	NNameStringPairCreate [959]	Creates a new string pair structure.
	NNameStringPairDispose [959]	Disposes of the specified string pair structure.
	NNameStringPairSet [960]	Sets the value of the specified string pair structure.
	NOSFamilyGetCurrent [960]	Retrieves the current operating system family.
	NPointerParseN [960]	Converts NString [678] to NPointer.
	NPointerParseStrOrChars [961]	Converts an array of NChars to NPointer.
	NPointerParseVN [961]	Converts NString [678] to NPointer.
	NPointerToChars [961]	Converts NPointer to an array of NChars.
	NPointerToString [962]	Converts NPointer to NString [678].
	NPointerTryParseN [962]	Converts NString [678] to NPointer. A NBool [1004] value indicates whether the conversion succeeded.
	NPointerTryParseStrOrChars [963]	Converts an array of NChars to NPointer. A NBool [1004] value indicates whether the conversion succeeded.
	NPointerTryParseVN [963]	Converts NString [678] to NPointer. A NBool [1004] value indicates whether the conversion succeeded.
	NRangeToString [964]	Converts NRange to NString [678].
	NRationalToString [964]	Converts NRational [1007] to NString [678].
	NResultParseN [964]	Converts NString [678] to NResult [1007].
	NResultParseStrOrChars [965]	Converts an array of NChars to NResult [1007].
	NResultParseVN [965]	Converts NString [678] to NResult [1007].
	NResultToChars [966]	Converts NResult [1007] to an array of NChars.
	NResultToString [966]	Converts NResult [1007] to NString [678].
	NResultTryParseN [967]	Converts NString [678] to NResult [1007]. A NBool [1004] value indicates whether the conversion succeeded.
	NResultTryParseStrOrChars [967]	Converts an array of NChars to NResult [1007]. A NBool [1004] value indicates whether the conversion succeeded.
	NResultTryParseVN [967]	Converts NString [678] to NResult [1007]. A NBool [1004] value indicates whether the conversion succeeded.
	NSingleGetNaN [968]	Retrieves a NSingle [1008] that is Not a Number (NaN).
	NSingleGetNegativeInfinity [968]	Retrieves a NSingle [1008] that is NegativeInfinity.
	NSingleGetPositiveInfinity [968]	Retrieves a NSingle [1008] that is PositiveInfinity.
	NSingleIsInfinity [969]	Checks if NSingle [1008] evaluates to NegativeInfinity or PositiveInfinity.
	NSingleIsNaN [969]	Checks if NSingle [1008] Not a Number (NaN).
	NSingleIsNegativeInfinity [969]	Checks if NSingle [1008] evaluates to NegativeInfinity.
	NSingleIsPositiveInfinity [970]	Checks if NSingle [1008] evaluates to PositiveInfinity.
	NSingleParseN [970]	Converts NString [678] to NSingle [1008].
	NSingleParseStrOrChars [970]	Converts an array of NChars to NSingle [1008].
	NSingleParseVN [971]	Converts NString [678] to NSingle [1008].
	NSingleToChars [971]	Converts NSingle [1008] to an array of NChars.
	NSingleToString [972]	Converts NSingle [1008] to NString [678].
	NSingleTryParseN [972]	Converts NString [678] to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.
	NSingleTryParseStrOrChars [972]	Converts an array of NChars to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.
	NSingleTryParseVN [973]	Converts NString [678] to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.
	NSizeTypeParseN [973]	Converts NString [678] to NSizeType [1008].
	NSizeTypeParseStrOrChars [974]	Converts an array of NChars to NSizeType [1008].
	NSizeTypeParseVN [974]	Converts NString [678] to NSizeType [1008].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NSizeTypeToChars [975]	Converts NSizeType [1008] to an array of NChars.
	NSizeTypeToString [975]	Converts NSizeType [1008] to NString [678].
	NSizeTypeTryParseN [975]	Converts NString [678] to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.
	NSizeTypeTryParseStrOrChars [976]	Converts an array of NChars to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.
	NSizeTypeTryParseVN [976]	Converts NString [678] to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.
	NSSizeTypeParseN [977]	Converts NString [678] to NSSizeType [1009].
	NSSizeTypeParseStrOrChars [977]	Converts an array of NChars to NSSizeType [1009].
	NSSizeTypeParseVN [978]	Converts NString [678] to NSSizeType [1009].
	NSSizeTypeToChars [978]	Converts NSSizeType [1009] to an array of NChars.
	NSSizeTypeToString [979]	Converts NSSizeType [1009] to NString [678].
	NSSizeTypeTryParseN [979]	Converts NString [678] to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NSSizeTypeTryParseStrOrChars [979]	Converts an array of NChars to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NSSizeTypeTryParseVN [980]	Converts NString [678] to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NTypesGetPointer [980]	Retrieves a pointer to object.
	NUInt16ParseN [981]	Converts NString [678] to NUInt16 [1009].
	NUInt16ParseStrOrChars [981]	Converts an array of NChars to NUInt16 [1009].
	NUInt16ParseVN [981]	Converts NString [678] to NUInt16 [1009].
	NUInt16ToChars [982]	Converts NUInt16 [1009] to an array of NChars.
	NUInt16ToString [982]	Converts NUInt16 [1009] to NString [678].
	NUInt16TryParseN [983]	Converts NString [678] to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt16TryParseStrOrChars [983]	Converts an array of NChars to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt16TryParseVN [984]	Converts NString [678] to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt32ParseN [984]	Converts NString [678] to NUInt32 [1009].
	NUInt32ParseStrOrChars [984]	Converts an array of NChars to NUInt32 [1009].
	NUInt32ParseVN [985]	Converts NString [678] to NUInt32 [1009].
	NUInt32ToChars [985]	Converts NUInt32 [1009] to an array of NChars.
	NUInt32ToString [986]	Converts NUInt32 [1009] to NString [678].
	NUInt32TryParseN [986]	Converts NString [678] to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt32TryParseStrOrChars [987]	Converts an array of NChars to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt32TryParseVN [987]	Converts NString [678] to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt64ParseN [988]	Converts NString [678] to NUInt64 [1009].
	NUInt64ParseStrOrChars [988]	Converts an array of NChars to NUInt64 [1009].
	NUInt64ParseVN [988]	Converts NString [678] to NUInt64 [1009].
	NUInt64ToChars [989]	Converts NUInt64 [1009] to an array of NChars.
	NUInt64ToString [989]	Converts NUInt64 [1009] to NString [678].
	NUInt64TryParseN [990]	Converts NString [678] to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NUInt64TryParseStrOrChars [990]	Converts an array of NChars to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt64TryParseVN [991]	Converts NString [678] to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt8ParseN [991]	Converts NString [678] to NUInt8 [1009].
	NUInt8ParseStrOrChars [991]	Converts an array of NChars to NUInt8 [1009].
	NUInt8ParseVN [992]	Converts NString [678] to NUInt8 [1009].
	NUInt8ToChars [992]	Converts NUInt8 [1009] to an array of NChars.
	NUInt8ToString [993]	Converts NUInt8 [1009] to NString [678].
	NUInt8TryParseN [993]	Converts NString [678] to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt8TryParseStrOrChars [994]	Converts an array of NChars to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt8TryParseVN [994]	Converts NString [678] to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NURationalToString [995]	Converts NURational [1010] to NString [678].
	NVersionRangeContains [995]	Checks if NVersionRange [1011] contains a certain version.
	NVersionRangeContainsRange [995]	Checks if NVersionRange [1011] contains a certain version range.
	NVersionRangeIntersect [996]	Replaces NVersionRange [1011] with the intersection of itself and another NVersionRange [1011].
	NVersionRangeIntersectsWith [996]	Returns third NVersionRange [1011] that represents the intersection of two other NVersionRanges. If there is no intersection, an empty NVersionRange [1011] is returned.
	NVersionRangeToString [996]	Converts NVersionRange [1011] to NString [678].
	NVersionToString [997]	Converts NVersion [1010] to NString [678].
	NWCharParseN [997]	Converts NString [678] to NWChar.
	NWCharParseStrOrChars [998]	Converts an array of NChars to NWChar.
	NWCharParseVN [998]	Converts NString [678] to NWChar.
	NWCharToChars [998]	Converts NWChar to an array of NChars.
	NWCharToString [999]	Converts NWChar to NString [678].
	NWCharTryParseN [999]	Converts NString [678] to NWChar. A NBool [1004] value indicates whether the conversion succeeded.
	NWCharTryParseStrOrChars [1000]	Converts an array of NChars to NWChar. A NBool [1004] value indicates whether the conversion succeeded.
	NWCharTryParseVN [1000]	Converts NString [678] to NWChar. A NBool [1004] value indicates whether the conversion succeeded.

Macros

Name	Description
N_DECLARE_HANDLE [1011]	Declares handle with specified name.
N_AW [1011]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version in a macro (with either 'A' or 'W' suffix accordingly).
N_BOOLEAN_FALSE_STRING [1011]	ANSI or Unicode (if N_UNICODE [384] is defined) string, representing NFalse [1041] Boolean value.
N_BOOLEAN_FALSE_STRINGA [1011]	ANSI string, representing NFalse [1041] Boolean value.
N_BOOLEAN_FALSE_STRINGW [1012]	Unicode (if N_UNICODE [384] is defined) string, representing NFalse [1041] Boolean value.
N_BOOLEAN_TRUE_STRING [1012]	ANSI or Unicode (if N_UNICODE [384] is defined) string, representing NTrue [1064] Boolean value.
N_BOOLEAN_TRUE_STRINGA [1012]	ANSI string, representing NTrue [1064] Boolean value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_BOOLEAN_TRUE_STRINGW [1012]	Unicode (if N_UNICODE [384] is defined) string, representing NTrue [1064] Boolean value.
N_BYTE_MAX [1012]	Maximum value for NByte [1005].
N_BYTE_MIN [1013]	Minimum value for NByte [1005].
N_CALLBACK_AW [1013]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the callback (with either 'A' or 'W' suffix accordingly).
N_DECLARE_HANDLE_TYPE [1013]	Declares handle type with specified name.
N_DECLARE_OBJECT_TYPE [1013]	Declares object type with specified name.
N_DECLARE_STATIC_OBJECT_TYPE [1013]	Declares static object type with specified name.
N_DECLARE_TYPE [1014]	Declares type with specified name.
N_DECLATE_PRIMITIVE [1014]	Declares primitive with specified name.
N_DOUBLE_EPSILON [1014]	Epsilon value for NDouble [1005].
N_DOUBLE_MAX [1014]	Maximum value for NDouble [1005].
N_DOUBLE_MIN [1014]	Minimum value for NDouble [1005].
N_EMPTY_STRING [1014]	ANSI or Unicode (if N_UNICODE [384] is defined) string, representing empty string value.
N_EMPTY_STRINGA [1015]	ANSI string, representing empty string value.
N_EMPTY_STRINGW [1015]	Unicode (if N_UNICODE [384] is defined) string, representing empty string value.
N_FIELD_AW [1015]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the field (with either 'A' or 'W' suffix accordingly).
N_FLOAT_EPSILON [1015]	Epsilon value for NFloat [1005].
N_FLOAT_INFINITY_STRING [1015]	ANSI or Unicode (if N_UNICODE [384] is defined) string, representing "Infinity" floating-point value.
N_FLOAT_INFINITY_STRINGA [1015]	ANSI string, representing "Infinity" floating-point value.
N_FLOAT_INFINITY_STRINGW [1016]	Unicode (if N_UNICODE [384] is defined) string, representing "Infinity" floating-point value.
N_FLOAT_MAX [1016]	Maximum value for NFloat [1005].
N_FLOAT_MIN [1016]	Minimum value for NFloat [1005].
N_FLOAT_NAN_STRING [1016]	ANSI or Unicode (if N_UNICODE [384] is defined) string, representing "NaN" floating-point value.
N_FLOAT_NAN_STRINGA [1016]	ANSI string, representing "NaN" floating-point value.
N_FLOAT_NAN_STRINGW [1017]	Unicode (if N_UNICODE [384] is defined) string, representing "NaN" floating-point value.
N_FUNC_AW [1017]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the function (with either 'A' or 'W' suffix accordingly).
N_INT_MAX [1017]	Maximum value for NInt [1006].
N_INT_MIN [1017]	Minimum value for NInt [1006].
N_INT16_MAX [1017]	Maximum value for NShort [1008].
N_INT16_MIN [1017]	Minimum value for NShort [1008].
N_INT32_MAX [1018]	Maximum value for NInt [1006].
N_INT32_MIN [1018]	Minimum value for NInt [1006].
N_INT64_MAX [1018]	Maximum value for NInt64 [1006].
N_INT64_MIN [1018]	Minimum value for NInt64 [1006].
N_INT8_MAX [1018]	Maximum value for NSByte [1008].
N_INT8_MIN [1018]	Minimum value for NSByte [1008].
N_JOIN_SYMBOLS [1019]	Concatenates two symbols, yielding a single concatenated symbol.
N_JOIN_SYMBOLS_ [1019]	Concatenates two symbols, yielding a single concatenated symbol. Can serve as a macro argument.
N_JOIN_SYMBOLS2 [1019]	Concatenates two symbols, yielding a single concatenated symbol.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_JOIN_SYMBOLS2_ [1019]	Concatenates two symbols, yielding a single concatenated symbol. Can serve as a macro argument.
N_LONG_MAX [1019]	Maximum value for NLong [1007].
N_LONG_MIN [1020]	Minimum value for NLong [1007].
N_MACRO_AW [1020]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the macro (with either 'A' or 'W' suffix accordingly).
N_PTR_SIZE [1020]	Defines pointer size.
N_SBYTE_MAX [1020]	Maximum value for NSByte [1008].
N_SBYTE_MIN [1020]	Minimum value for NSByte [1008].
N_SHORT_MAX [1020]	Maximum value for NShort [1008].
N_SHORT_MIN [1021]	Minimum value for NShort [1008].
N_SINGLE_EPSILON [1021]	Epsilon value for NSingle [1008].
N_SINGLE_MAX [1021]	Maximum value for NSingle [1008].
N_SINGLE_MIN [1021]	Minimum value for NSingle [1008].
N_SIZE_TYPE_MAX [1021]	Maximum value for NSizeType [1008].
N_SIZE_TYPE_MIN [1021]	Minimum value for NSizeType [1008].
N_SSIZE_TYPE_MAX [1022]	Maximum value for NSSizeType [1009].
N_SSIZE_TYPE_MIN [1022]	Minimum value for NSSizeType [1009].
N_STRINGIZE [1022]	Defines the size of ANSI or Unicode (if N_UNICODE [384] is defined) string.
N_STRINGIZE_ [1022]	Defines the size of ANSI or Unicode (if N_UNICODE [384] is defined) string. Can serve as a macro argument.
N_STRINGIZE_A [1022]	Defines the size of ANSI string. Can serve as a macro argument.
N_STRINGIZE_W [1023]	Defines the size of Unicode (if N_UNICODE [384] is defined) string. Can serve as a macro argument.
N_STRINGIZEA [1023]	Defines the size of ANSI string.
N_STRINGIZEW [1023]	Defines the size of Unicode (if N_UNICODE [384] is defined) string.
N_STRUCT_AW [1023]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the struct (with either 'A' or 'W' suffix accordingly).
N_SYMBOL_AW [1023]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the symbol (with either 'A' or 'W' suffix accordingly).
N_T [1023]	Makes either ANSI or Unicode (if N_UNICODE [384] is defined) string or character constant.
N_T_A [1024]	Makes ANSI string or character constant. Can serve as a macro argument.
N_T_W [1024]	Makes Unicode (if N_UNICODE [384] is defined) string or character constant. Can serve as a macro argument.
N_TA [1024]	Makes ANSI string or character constant.
N_TW [1024]	Makes Unicode (if N_UNICODE [384] is defined) string or character constant.
N_TYPE_AW [1024]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the type (with either 'A' or 'W' suffix accordingly).
N_TYPE_OF [1024]	Used to obtain a pointer to the function that retrieves NType [871] object for the specified type name.",
N_UINT_MAX [1025]	Maximum value for NUInt [1009].
N_UINT_MIN [1025]	Minimum value for NUInt [1009].
N_UINT16_MAX [1025]	Maximum value for NUShort [1010].
N_UINT16_MIN [1025]	Minimum value for NUShort [1010].
N_UINT32_MAX [1025]	Maximum value for NUInt [1009].
N_UINT32_MIN [1026]	Minimum value for NUInt [1009].
N_UINT64_MAX [1026]	Maximum value for NULong [1010].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_UINT64_MIN [1026]	Minimum value for NULong [1010].
N_UINT8_MAX [1026]	Maximum value for NUInt8 [1009].
N_UINT8_MIN [1026]	Minimum value for NUInt8 [1009].
N_ULONG_MAX [1026]	Maximum value for NULong [1010].
N_ULONG_MIN [1027]	Minimum value for NULong [1010].
N_UNREFERENCED_PARAMETER [1027]	Suppresses the warning "unreferenced parameter".
N_UNUSED_VARIABLE [1027]	Suppresses the warning "unused variable".
N_USHORT_MAX [1027]	Maximum value for NUShort [1010].
N_USHORT_MIN [1027]	Minimum value for NUShort [1010].
N_VAR_AW [1027]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the variable (with either 'A' or 'W' suffix accordingly).
N_WCHAR_SIZE [1028]	Defines the size of Unicode char.
NACharParse [1028]	Converts a NString [678] to NAChar [1004].
NACharParseChars [1028]	Converts an array of NChars to NAChar [1004].
NACharParseStrOrChars [1029]	Picks a Unicode (W) or ASCII (A) version of the NACharParseStrOrChars function.
NACharParseVN [1029]	Picks a Unicode (W) or ASCII (A) version of the NACharParseVN function.
NACharToChars [1029]	Picks a Unicode (W) or ASCII (A) version of the NACharToChars function.
NACharToString [1029]	Picks a Unicode (W) or ASCII (A) version of the NACharToString function.
NACharTryParse [1029]	Converts NString [678] to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.
NACharTryParseChars [1030]	Converts an array of NChars to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.
NACharTryParseStrOrChars [1030]	Picks a Unicode (W) or ASCII (A) version of the NACharTryParseStrOrChars function.
NACharTryParseVN [1030]	Picks a Unicode (W) or ASCII (A) version of the NACharTryParseVN function.
NBooleanParse [1030]	Converts NString [678] to NBoolean [1004].
NBooleanParseChars [1031]	Converts an array of NChars to NBoolean [1004].
NBooleanParseStrOrChars [1031]	Picks a Unicode (W) or ASCII (A) version of the NBooleanParseStrOrChars function.
NBooleanParseVN [1031]	Picks a Unicode (W) or ASCII (A) version of the NBooleanParseVN function.
NBooleanToChars [1032]	Picks a Unicode (W) or ASCII (A) version of the NBooleanToChars function.
NBooleanToString [1032]	Picks a Unicode (W) or ASCII (A) version of the NBooleanToString function.
NBooleanTryParse [1032]	Converts NString [678] to NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.
NBooleanTryParseChars [1032]	Converts an array of NChars to NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.
NBooleanTryParseStrOrChars [1033]	Picks a Unicode (W) or ASCII (A) version of the NBooleanTryParseStrOrChars function.
NBooleanTryParseVN [1033]	Picks a Unicode (W) or ASCII (A) version of the NBooleanTryParseVN function.
NCharFromBinDigit [1033]	Picks a Unicode (W) or ASCII (A) version of the NCharFromBinDigit function.
NCharFromDigit [1033]	Picks a Unicode (W) or ASCII (A) version of the NCharFromDigit function.
NCharFromHexDigit [1033]	Picks a Unicode (W) or ASCII (A) version of the NCharFromHexDigit function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCharFromOctDigit [1034]	Picks a Unicode (W) or ASCII (A) version of the NCharFromOctDigit function.
NCharIsAscii [1034]	Picks a Unicode (W) or ASCII (A) version of the NCharIsAscii function.
NCharIsBinDigit [1034]	Picks a Unicode (W) or ASCII (A) version of the NCharIsBinDigit function.
NCharIsDigit [1034]	Picks a Unicode (W) or ASCII (A) version of the NCharIsDigit function.
NCharIsHexDigit [1034]	Picks a Unicode (W) or ASCII (A) version of the NCharIsHexDigit function.
NCharIsLetter [1034]	Picks a Unicode (W) or ASCII (A) version of the NCharIsLetter function.
NCharIsLower [1035]	Picks a Unicode (W) or ASCII (A) version of the NCharIsLower function.
NCharIsOctDigit [1035]	Picks a Unicode (W) or ASCII (A) version of the NCharIsOctDigit function.
NCharIsUpper [1035]	Picks a Unicode (W) or ASCII (A) version of the NCharIsUpper function.
NCharIsWhiteSpace [1035]	Picks a Unicode (W) or ASCII (A) version of the NCharIsWhiteSpace function.
NCharParse [1035]	Converts NString [678] to NChar [1005].
NCharParseChars [1036]	Converts an array of NChars to NBoolean [1004].
NCharToBinDigit [1036]	Picks a Unicode (W) or ASCII (A) version of the NCharToBinDigit function.
NCharToDigit [1036]	Picks a Unicode (W) or ASCII (A) version of the NCharToDigit function.
NCharToHexDigit [1036]	Picks a Unicode (W) or ASCII (A) version of the NCharToHexDigit function.
NCharToLower [1037]	Picks a Unicode (W) or ASCII (A) version of the NCharToLower function.
NCharToOctDigit [1037]	Picks a Unicode (W) or ASCII (A) version of the NCharToOctDigit function.
NCharToUpper [1037]	Picks a Unicode (W) or ASCII (A) version of the NCharToUpper function.
NCharTryParse [1037]	Converts NString [678] to NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.
NCharTryParseChars [1038]	Converts an array of NChars to NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.
NDoubleParse [1038]	Converts NString [678] to NDouble [1005].
NDoubleParseChars [1038]	Converts an array of NChars to NChar [1005].
NDoubleParseStrOrChars [1039]	Picks a Unicode (W) or ASCII (A) version of the NDoubleParseStrOrChars function.
NDoubleParseVN [1039]	Picks a Unicode (W) or ASCII (A) version of the NDoubleParseVN function.
NDoubleToChars [1039]	Picks a Unicode (W) or ASCII (A) version of the NDoubleToChars function.
NDoubleToString [1039]	Picks a Unicode (W) or ASCII (A) version of the NDoubleToString function.
NDoubleTryParse [1039]	Converts NString [678] to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.
NDoubleTryParseChars [1040]	Converts an array of NChars to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.
NDoubleTryParseStrOrChars [1040]	Picks a Unicode (W) or ASCII (A) version of the NDoubleTryParseStrOrChars function.
NDoubleTryParseVN [1040]	Picks a Unicode (W) or ASCII (A) version of the NDoubleTryParseVN function.
NFalse [1041]	False value for NBoolean [1004].
NGuidParse [1041]	Picks a Unicode (W) or ASCII (A) version of the NGuidParse function.
NGuidParseVN [1041]	Picks a Unicode (W) or ASCII (A) version of the NGuidParseVN function.
NGuidToString [1041]	Picks a Unicode (W) or ASCII (A) version of the NGuidToString function.
NGuidTryParse [1041]	Picks a Unicode (W) or ASCII (A) version of the NGuidTryParse function.
NGuidTryParseVN [1041]	Picks a Unicode (W) or ASCII (A) version of the NGuidTryParseVN function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NIndexPairToString [1042]	Picks a Unicode (W) or ASCII (A) version of the NIndexPairToString function.
NInt16Parse [1042]	Converts NString [678] to NInt16 [1006].
NInt16ParseChars [1042]	Converts an array of NChars to NInt16 [1006].
NInt16ParseStrOrChars [1043]	Picks a Unicode (W) or ASCII (A) version of the NInt16ParseStrOrChars function.
NInt16ParseVN [1043]	Picks a Unicode (W) or ASCII (A) version of the NInt16ParseVN function.
NInt16ToChars [1043]	Picks a Unicode (W) or ASCII (A) version of the NInt16ToChars function.
NInt16ToString [1043]	Picks a Unicode (W) or ASCII (A) version of the NInt16ToString function.
NInt16TryParse [1043]	Converts NString [678] to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
NInt16TryParseChars [1044]	Converts an array of NChars to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
NInt16TryParseStrOrChars [1044]	Picks a Unicode (W) or ASCII (A) version of the NInt16TryParseStrOrChars function.
NInt16TryParseVN [1044]	Picks a Unicode (W) or ASCII (A) version of the NInt16TryParseVN function.
NInt32Parse [1044]	Converts NString [678] to NInt32 [1006].
NInt32ParseChars [1045]	Converts an array of NChars to NInt32 [1006].
NInt32ParseStrOrChars [1045]	Picks a Unicode (W) or ASCII (A) version of the NInt32ParseStrOrChars function.
NInt32ParseVN [1045]	Picks a Unicode (W) or ASCII (A) version of the NInt32ParseVN function.
NInt32ToChars [1046]	Picks a Unicode (W) or ASCII (A) version of the NInt32ToChars function.
NInt32ToString [1046]	Picks a Unicode (W) or ASCII (A) version of the NInt32ToString function.
NInt32TryParse [1046]	Converts NString [678] to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
NInt32TryParseChars [1046]	Converts an array of NChars to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
NInt32TryParseStrOrChars [1047]	Picks a Unicode (W) or ASCII (A) version of the NInt32TryParseStrOrChars function.
NInt32TryParseVN [1047]	Picks a Unicode (W) or ASCII (A) version of the NInt32tryParseVN function.
NInt64Parse [1047]	Converts NString [678] to NInt64 [1006].
NInt64ParseChars [1047]	Converts an array of NChars to NInt64 [1006].
NInt64ParseStrOrChars [1048]	Picks a Unicode (W) or ASCII (A) version of the NInt64ParseStrOrChars function.
NInt64ParseVN [1048]	Picks a Unicode (W) or ASCII (A) version of the NInt64ParseVN function.
NInt64ToChars [1048]	Picks a Unicode (W) or ASCII (A) version of the NInt64ToChars function.
NInt64ToString [1048]	Picks a Unicode (W) or ASCII (A) version of the NInt64ToString function.
NInt64TryParse [1048]	Converts NString [678] to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
NInt64TryParseChars [1049]	Converts an array of NChars to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
NInt64TryParseStrOrChars [1049]	Picks a Unicode (W) or ASCII (A) version of the NInt64TryParseStrOrChars function.
NInt64TryParseVN [1049]	Picks a Unicode (W) or ASCII (A) version of the NInt64TryParseVN function.
NInt8Parse [1050]	Converts NString [678] to NInt8 [1007].
NInt8ParseChars [1050]	Converts an array of NChars to NInt8 [1007].
NInt8ParseStrOrChars [1050]	Picks a Unicode (W) or ASCII (A) version of the NInt8ParseStrOrChars function.
NInt8ParseVN [1051]	Picks a Unicode (W) or ASCII (A) version of the NInt8ParseVN function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt8ToChars [1051]	Picks a Unicode (W) or ASCII (A) version of the NInt8ToChars function.
NInt8ToString [1051]	Picks a Unicode (W) or ASCII (A) version of the NInt8ToString function.
NInt8TryParse [1051]	Converts NString [678] to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.
NInt8TryParseChars [1051]	Converts an array of NChars to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.
NInt8TryParseStrOrChars [1052]	Picks a Unicode (W) or ASCII (A) version of the NInt8TryParseStrOrChars function.
NInt8TryParseVN [1052]	Picks a Unicode (W) or ASCII (A) version of the NInt8TryParseVN function.
NPointerParse [1052]	Converts NString [678] to NPointer.
NPointerParseChars [1053]	Converts an array of NChars to NPointer.
NPointerParseStrOrChars [1053]	Picks a Unicode (W) or ASCII (A) version of the NPointerParseStrOrChars function.
NPointerParseVN [1053]	Picks a Unicode (W) or ASCII (A) version of the NPointerParseVN function.
NPointerToChars [1053]	Picks a Unicode (W) or ASCII (A) version of the NPointerToChars function.
NPointerToString [1054]	Picks a Unicode (W) or ASCII (A) version of the NPointerToString function.
NPointerTryParse [1054]	Converts NString [678] to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.
NPointerTryParseChars [1054]	Converts an array of NChars to NPointer. A NBool [1004] value indicates whether the conversion succeeded.
NPointerTryParseStrOrChars [1055]	Picks a Unicode (W) or ASCII (A) version of the NPointerTryParseStrOrChars function.
NPointerTryParseVN [1055]	Picks a Unicode (W) or ASCII (A) version of the NPointerTryParseVN function.
NRangeToString [1055]	Picks a Unicode (W) or ASCII (A) version of the NRangeToString function.
NRationalToString [1055]	Picks a Unicode (W) or ASCII (A) version of the NRationalToString function.
NResultParseVN [1055]	Picks a Unicode (W) or ASCII (A) version of the NResultParseVN function.
NResultToChars [1055]	Picks a Unicode (W) or ASCII (A) version of the NResultToChars function.
NResultToString [1056]	Picks a Unicode (W) or ASCII (A) version of the ResultToString function.
NResultTryParseStrOrChars [1056]	Picks a Unicode (W) or ASCII (A) version of the NResultTryParseStrOrChars function.
NResultTryParseVN [1056]	Picks a Unicode (W) or ASCII (A) version of the NResultTryParseVN function.
NSingleParse [1056]	Converts NString [678] to NSingle [1008].
NSingleParseChars [1056]	Converts an array of NChars to NSingle [1008].
NSingleParseStrOrChars [1057]	Picks a Unicode (W) or ASCII (A) version of the NSingleParseStrOrChars function.
NSingleParseVN [1057]	Picks a Unicode (W) or ASCII (A) version of the NSingleParseVN function.
NSingleToChars [1057]	Picks a Unicode (W) or ASCII (A) version of the NSingleToChars function.
NSingleToString [1057]	Picks a Unicode (W) or ASCII (A) version of the NSingleToString function.
NSingleTryParse [1058]	Converts NString [678] to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.
NSingleTryParseChars [1058]	Converts an array of NChars to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.
NSingleTryParseStrOrChars [1058]	Picks a Unicode (W) or ASCII (A) version of the NSingleTryParseStrOrChars function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSingleTryParseVN [1059]	Picks a Unicode (W) or ASCII (A) version of the NSingleTryParseVN function.
NSizeTypeParse [1059]	Converts NString [678] to NSizeType [1008].
NSizeTypeParseChars [1059]	Converts an array of NChars to NSizeType [1008].
NSizeTypeParseStrOrChars [1059]	Picks a Unicode (W) or ASCII (A) version of the NSizeTypeParseStrOrChars function.
NSizeTypeParseVN [1060]	Picks a Unicode (W) or ASCII (A) version of the NSizeTypeParseVN function.
NSizeTypeToChars [1060]	Picks a Unicode (W) or ASCII (A) version of the NSizeTypeToChars function.
NSizeTypeToString [1060]	Picks a Unicode (W) or ASCII (A) version of the NSizeTypeToString function.
NSizeTypeTryParse [1060]	Converts NString [678] to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.
NSizeTypeTryParseChars [1061]	Converts an array of NChars to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.
NSizeTypeTryParseStrOrChars [1061]	Picks a Unicode (W) or ASCII (A) version of the NSizeTypeTryParseStrOrChars function.
NSizeTypeTryParseVN [1061]	Picks a Unicode (W) or ASCII (A) version of the NSizeTypeTryParseVN function.
NSSizeTypeParse [1061]	Converts NString [678] to NSSizeType [1009].
NSSizeTypeParseChars [1062]	Converts an array of NChars to NSSizeType [1009].
NSSizeTypeParseStrOrChars [1062]	Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeParseStrOrChars function.
NSSizeTypeParseVN [1062]	Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeParseVN function.
NSSizeTypeToChars [1063]	Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeToChars function.
NSSizeTypeToString [1063]	Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeToString function.
NSSizeTypeTryParse [1063]	Converts NString [678] to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NSSizeTypeTryParseChars [1063]	Converts an array of NChars to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NSSizeTypeTryParseStrOrChars [1064]	Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeTryParseStrOrChars function.
NSSizeTypeTryParseVN [1064]	Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeTryParseVN function.
NTrue [1064]	True value for NBoolean [1004].
NUInt16Parse [1064]	Converts NString [678] to NUInt16 [1009].
NUInt16ParseChars [1065]	Converts an array of NChars to NUInt16 [1009].
NUInt16ParseStrOrChars [1065]	Picks a Unicode (W) or ASCII (A) version of the NUInt16ParseStrOrChars function.
NUInt16ParseVN [1065]	Picks a Unicode (W) or ASCII (A) version of the NUInt16ParseVN function.
NUInt16ToChars [1065]	Picks a Unicode (W) or ASCII (A) version of the NUInt16ToChars function.
NUInt16ToString [1066]	Picks a Unicode (W) or ASCII (A) version of the NUInt16ParseToString function.
NUInt16TryParse [1066]	Converts NString [678] to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt16TryParseChars [1066]	Converts an array of NChars to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt16TryParseStrOrChars [1067]	Picks a Unicode (W) or ASCII (A) version of the NUInt16TryParseStrOrChars function.
NUInt16TryParseVN [1067]	Picks a Unicode (W) or ASCII (A) version of the NUInt16TryParseVN function.
NUInt32Parse [1067]	Converts NString [678] to NUInt32 [1009].
NUInt32ParseChars [1067]	Converts an array of NChars to NUInt32 [1009].
NUInt32ParseStrOrChars [1068]	Picks a Unicode (W) or ASCII (A) version of the NUInt32ParseStrOrChars function.
NUInt32ParseVN [1068]	Picks a Unicode (W) or ASCII (A) version of the NUInt32ParseVN function.
NUInt32ToChars [1068]	Picks a Unicode (W) or ASCII (A) version of the NUInt32ToChars function.
NUInt32ToString [1068]	Picks a Unicode (W) or ASCII (A) version of the NUInt32ToString function.
NUInt32TryParse [1068]	Converts NString [678] to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt32TryParseChars [1069]	Converts an array of NChars to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt32TryParseStrOrChars [1069]	Picks a Unicode (W) or ASCII (A) version of the NUInt32TryParseStrOrChars function.
NUInt32TryParseVN [1069]	Picks a Unicode (W) or ASCII (A) version of the NUInt32TryParseVN function.
NUInt64Parse [1070]	Converts NString [678] to NUInt64 [1009].
NUInt64ParseChars [1070]	Converts an array of NChars to NUInt64 [1009].
NUInt64ParseStrOrChars [1070]	Picks a Unicode (W) or ASCII (A) version of the NUInt64ParseStrOrChars function.
NUInt64ParseVN [1071]	Picks a Unicode (W) or ASCII (A) version of the NUInt64ParseVN function.
NUInt64ToChars [1071]	Picks a Unicode (W) or ASCII (A) version of the NUInt64ToChars function.
NUInt64ToString [1071]	Picks a Unicode (W) or ASCII (A) version of the NUInt64ToString function.
NUInt64TryParse [1071]	Converts NString [678] to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt64TryParseChars [1071]	Converts an array of NChars to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt64TryParseStrOrChars [1072]	Picks a Unicode (W) or ASCII (A) version of the NUInt64TryParseStrOrChars function.
NUInt64TryParseVN [1072]	Picks a Unicode (W) or ASCII (A) version of the NUInt64TryParseVN function.
NUInt8Parse [1072]	Converts NString [678] to NUInt8 [1009].
NUInt8ParseChars [1073]	Converts an array of NChars to NUInt8 [1009].
NUInt8ParseStrOrChars [1073]	Picks a Unicode (W) or ASCII (A) version of the NUInt8ParseStrOrChars function.
NUInt8ParseVN [1073]	Picks a Unicode (W) or ASCII (A) version of the NUInt8ParseVN function.
NUInt8ToChars [1073]	Picks a Unicode (W) or ASCII (A) version of the NUInt8ToChars function.
NUInt8ToString [1074]	Picks a Unicode (W) or ASCII (A) version of the NUInt8ToString function.
NUInt8TryParse [1074]	Converts NString [678] to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt8TryParseChars [1074]	Converts an array of NChars to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt8TryParseStrOrChars [1075]	Picks a Unicode (W) or ASCII (A) version of the NUInt8TryParseStrOrChars function.
NUInt8TryParseVN [1075]	Picks a Unicode (W) or ASCII (A) version of the NUInt8TryParseVN function.
NULL [1075]	Null value for pointer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NURationalToString [1075]	Picks a Unicode (W) or ASCII (A) version of the NURationalToString function.
NVersionRangeToString [1075]	Picks a Unicode (W) or ASCII (A) version of the NVersionRangeToString function.
NVersionToString [1075]	Picks a Unicode (W) or ASCII (A) version of the NVersionToString function.
NWCharParse [1076]	Converts NString [678] to NWChar.
NWCharParseChars [1076]	Converts an array of NChars to NWChar.
NWCharParseStrOrChars [1076]	Picks a Unicode (W) or ASCII (A) version of the NWCharParseStrOrChars function.
NWCharParseVN [1077]	Picks a Unicode (W) or ASCII (A) version of the NWCharParseVN function.
NWCharToChars [1077]	Picks a Unicode (W) or ASCII (A) version of the NWCharToChars function.
NWCharToString [1077]	Picks a Unicode (W) or ASCII (A) version of the NWCharToString function.
NWCharTryParse [1077]	Converts NString [678] to NWChar. A NBool [1004] value indicates whether the conversion succeeded.
NWCharTryParseChars [1077]	Converts an array of NChars to NWChar. A NBool [1004] value indicates whether the conversion succeeded.
NWCharTryParseStrOrChars [1078]	Picks a Unicode (W) or ASCII (A) version of the NWCharTryParseStrOrChars function.
NWCharTryParseVN [1078]	Picks a Unicode (W) or ASCII (A) version of the NWCharTryParseVN function.

Structs, Records, Enums

	Name	Description
	NAttributes_ [1002]	Enumerates value attributes.
	NComplex_ [1001]	Represents a complex number value.
	NIndexPair_ [1001]	Represents a pair of indexes.
	NNameStringPair_ [1001]	Represents a pair of name strings.
	NOSFamily_ [1003]	Enumerates OS families.
	NRational_ [1002]	Represents a signed rational number.
	NURational_ [1002]	Represents an unsigned rational number.
	NAttributes [1002]	Enumerates value attributes.
	NOSFamily [1003]	Enumerates OS families.

Types

Name	Description
NAChar [1004]	ANSI character (8-bit).
NBool [1004]	Same as NBoolean [1004].
NBoolean [1004]	32-bit boolean value. See also NTrue [1064] and NFalse [1041].
NByte [1005]	Same as NUInt8 [1009].
NChar [1005]	ANSI character (8-bit). See also NAChar [1004].
NComplex [1005]	Complex number.
NDouble [1005]	Double precision floating point number.
NFloat [1005]	Same as NSingle [1008].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NHandle [1006]	Pointer to unspecified data (same as void *).
NIndexPair [1006]	Represents a pair of indexes.
NInt [1006]	Same as NInt32 [1006].
NInt16 [1006]	16-bit signed integer (short).
NInt32 [1006]	32-bit signed integer (int).
NInt64 [1006]	64-bit signed integer (long). Not available on some 32-bit platforms.
NInt8 [1007]	8-bit signed integer (signed byte).
NLong [1007]	Same as NInt64 [1006].
NNameStringPair [1007]	Two linked strings.
NRational [1007]	Represents a signed rational number.
NResult [1007]	Result of a function (same as NInt [1006]). See also NErrors module.
NSByte [1008]	Same as NInt8 [1007].
NShort [1008]	Same as NInt16 [1006].
NSingle [1008]	Single precision floating point number.
NSizeType [1008]	Platform dependent size type. Unsigned 32-bit integer on 32-bit platform, unsigned 64-bit integer on 64-bit platform.
NSSizeType [1009]	Platform dependent size type. Signed 32-bit integer on 32-bit platform, signed 64-bit integer on 64-bit platform.
NUInt [1009]	Same as NUInt32 [1009].
NUInt16 [1009]	16-bit unsigned integer (unsigned short).
NUInt32 [1009]	32-bit unsigned integer (unsigned int).
NUInt64 [1009]	64-bit unsigned integer (unsigned long). Not available on some 32-bit platforms.
NUInt8 [1009]	8-bit unsigned integer (byte).
NULong [1010]	Same as NUInt64 [1009].
NURational [1010]	Represents an unsigned rational number.
NUShort [1010]	Same as NUInt16 [1009].
NVersion [1010]	Two NInt32 [1006] values defining a version.
NVersion_ [1010]	A NUShort [1010] typedef defining a version.
NVersionRange [1011]	Four NInt32 [1006] values defining a version range.
NVersionRange_ [1011]	A NUInt [1009] typedef defining a version range.

7.1.2.64.1 Functions

7.1.2.64.1.1 NACharParseN Function

Converts a NString [678] to NAChar [1004].

C++

```
NResult N_API NACharParseN(HNString hValue, HNString hFormat, NAChar * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to format string.
NAChar * pValue	[out] A pointer to the converted NAChar [1004].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.2 NACharParseStrOrChars Function

Converts an array of NChars to NAChar [1004].

C++

```
NResult N_API NACharParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NAChar * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NAChar * pValue	[out] A pointer to the converted NAChar [1004].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.3 NACharParseVN Function

Converts a NString [678] to NAChar [1004].

C++

```
NResult N_API NACharParseVN(HNString hValue, const NChar * szFormat, NAChar * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NAChar * pValue	[out] A pointer to the converted NAChar [1004].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.4 NACharToChars Function

Converts NAChar [1004] to an array of NAChars.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NACharToChars(NAChar value, const NChar * szFormat, NChar * arValue, NInt valueLength);
```

Parameters

Parameters	Description
NAChar value	[in] The NAChar [1004] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NChar * arValue	[out] A pointer the converted array of NAChars.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.5 NACharToString Function

Converts NAChar [1004] to NString [678]

C++

```
NResult N_API NACharToString(NAChar value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NAChar value	[in] NAChar [1004] to be converted.
const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.6 NACharTryParseN Function

Converts NString [678] to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NACharTryParseN(HNString hValue, HNString hFormat, NAChar * pValue, NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to format string.
NAChar * pValue	[out] A pointer to the converted NAChar [1004].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.7 NACharTryParseStrOrChars Function

Converts an array of NChars to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NACharTryParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar
* szFormat, NAChar * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NAChar * pValue	[out] A pointer to the converted NAChar [1004].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.8 NACharTryParseVN Function

Converts NString [678] to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NACharTryParseVN(HNString hValue, const NChar * szFormat, NAChar * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NAChar * pValue	[out] A pointer to the converted NAChar [1004].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.9 NBooleanParseN Function

Converts NString [678] to NBoolean [1004].

C++

```
NResult N_API NBooleanParseN(HNString hValue, HNString hFormat, NBoolean * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted..
HNString hFormat	[in] Handle to format string.
NBoolean * pValue	[out] A pointer to the converted NBoolean [1004].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.10 NBooleanParseStrOrChars Function

Converts an array of NChars to NBoolean [1004].

C++

```
NResult N_API NBooleanParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar
* szFormat, NBoolean * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NBoolean * pValue	[out] A pointer to the converted NBoolean [1004].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.11 NBooleanParseVN Function

Converts NString [678] to NBoolean [1004].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NBooleanParseVN(HNString hValue, const NChar * szFormat, NBoolean * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NBoolean * pValue	[out] A pointer to the converted NBoolean [1004].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.12 NBooleanToChars Function

Converts NBoolean [1004] to an array of NChars.

C++

```
NResult N_API NBooleanToChars(NBoolean value, const NChar * szFormat, NChar * arValue, NInt
valueLength);
```

Parameters

Parameters	Description
NBoolean value	[in] The NBoolean [1004] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NChar * arValue	[out] A pointer to the converted array of NAChars.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.13 NBooleanToString Function

Converts NBoolean [1004] to NString [678].

C++

```
NResult N_API NBooleanToString(NBoolean value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NBoolean value	[in] NBoolean [1004] to be converted.
const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.14 NBooleanTryParseN Function

Converts a NString [678] to NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NBooleanTryParseN(HNString hValue, HNString hFormat, NBoolean * pValue, NBool
* pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to convert.
HNString hFormat	[in] Handle to format string.
NBoolean * pValue	[out] A pointer to the converted NBoolean [1004].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.15 NBooleanTryParseStrOrChars Function

Converts an array of NStrings or NChars into NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NBooleanTryParseStrOrChars(const NChar * arValue, NInt valueLength, const
NChar * szFormat, NBoolean * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to a NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NBoolean * pValue	[out] A pointer to the converted NBoolean [1004].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.16 NBooleanTryParseVN Function

Converts a NString [678] to NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NBooleanTryParseVN(HNString hValue, const NChar * szFormat, NBoolean *
pValue, NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to convert.
const NChar * szFormat	[in] Pointer to format string.
NBoolean * pValue	[out] A pointer to the converted NBoolean [1004].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.17 NCharFromBinDigit Function

Converts a binary digit to NChar [1005].

C++

```
NChar N_API NCharFromBinDigit(NInt value);
```

Parameters

Parameters	Description
NInt value	[in] NInt [1006] value to be converted.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.18 NCharFromDigit Function

Converts a digit to NChar [1005].

C++

```
NChar N_API NCharFromDigit(NInt value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NInt value	[in] NInt [1006] value to be converted.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.19 NCharFromHexDigit Function

Converts a hexadecimal digit to NChar [1005].

C++

```
NChar N_API NCharFromHexDigit(NInt value, NBool lowercase);
```

Parameters

Parameters	Description
NInt value	[in] NInt [1006] value to be converted.
NBool lowercase	[in] NTrue [1064] if lowercase; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.20 NCharFromOctDigit Function

Converts an octal digit to NChar [1005].

C++

```
NChar N_API NCharFromOctDigit(NInt value);
```

Parameters

Parameters	Description
NInt value	[in] NInt [1006] value to be converted.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.21 NCharIsAscii Function

Checks whether the specified char is ASCII.

C++

```
NBool N_API NCharIsAscii(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] Character to be checked.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.22 NCharIsBinDigit Function

Checks whether the specified character is a binary digit.

C++

```
NBool N_API NCharIsBinDigit(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] Character to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.23 NCharIsDigit Function

Checks whether the specified character is a digit.

C++

```
NBool N_API NCharIsDigit(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] Character to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.24 NCharIsHexDigit Function

Checks whether the specified character is a hexadecimal digit.

C++

```
NBool N_API NCharIsHexDigit(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] Character to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.1.25 NCharIsLetter Function

Checks whether the specified character is a letter.

C++

```
NBool N_API NCharIsLetter(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] Character to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.26 NCharIsLower Function

Checks whether the specified character is a lower value.

C++

```
NBool N_API NCharIsLower(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] Character to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.27 NCharIsOctDigit Function

Checks whether the specified character is an octal digit.

C++

```
NBool N_API NCharIsOctDigit(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] Character to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.28 NCharIsUpper Function

Checks whether the specified character is an upper value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NBool N_API NCharIsUpper(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] Character to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.29 NCharIsWhiteSpace Function

Checks whether the specified character is a whitespace.

C++

```
NBool N_API NCharIsWhiteSpace(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] Character to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.30 NCharParseN Function

Converts a NString [678] to a NChar [1005].

C++

```
NResult N_API NCharParseN(HNString hValue, HNString hFormat, NChar * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to convert.
HNString hFormat	For internal use.
NChar * pValue	[out] A pointer to converted NChar [1005].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.31 NCharParseStrOrChars Function

Converts an array of NStrings or NChars into NChar [1005].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NCharParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NChar * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to a NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NChar * pValue	[out] A pointer to converted NChar [1005].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.32 NCharParseVN Function

Converts a NString [678] to a NChar [1005].

C++

```
NResult N_API NCharParseVN(HNString hValue, const NChar * szFormat, NChar * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to convert.
const NChar * szFormat	[in] Pointer to format string.
NChar * pValue	[out] A pointer to converted NChar [1005].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.33 NCharToBinDigit Function

Converts a NChar [1005] to a binary digit.

C++

```
NInt N_API NCharToBinDigit(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] NChar [1005] to convert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.1.34 NCharToChars Function

Converts a NChar [1005] into an array of NChars.

C++

```
NResult N_API NCharToChars(NChar value, const NChar * szFormat, NChar * arValue, NInt
valueLength);
```

Parameters

Parameters	Description
NChar value	[in] A NChar [1005] to convert.
const NChar * szFormat	For internal use.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] The length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.35 NCharToDigit Function

Converts a NChar [1005] to a digit.

C++

```
NInt N_API NCharToDigit(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] NChar [1005] to convert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.36 NCharToHexDigit Function

Converts a NChar [1005] to a hexadecimal digit.

C++

```
NInt N_API NCharToHexDigit(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] NChar [1005] to convert.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.37 NCharToLower Function

Converts a NChar [1005] to lowercase.

C++

```
NChar N_API NCharToLower(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] NChar [1005] to convert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.38 NCharToOctDigit Function

Converts a NChar [1005] to an octal digit.

C++

```
NInt N_API NCharToOctDigit(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] NChar [1005] to convert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.39 NCharToString Function

Converts NChar [1005] to NString [678].

C++

```
NResult N_API NCharToString(NChar value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NChar value	[in] NChar [1005] to convert.
const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.40 NCharToUpper Function

Converts NChar [1005] to uppercase.

C++

```
NChar N_API NCharToUpper(NChar value);
```

Parameters

Parameters	Description
NChar value	[in] NChar [1005] to convert.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.41 NCharTryParseN Function

Converts a NString [678] to a NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NCharTryParseN(HNString hValue, HNString hFormat, NChar * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to convert.
HNString hFormat	For internal use.
NChar * pValue	[out] A pointer to converted NChar [1005].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.42 NCharTryParseStrOrChars Function

Converts an array of NChars into NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NCharTryParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar
* szFormat, NChar * pValue, NBool * pResult);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to a NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NChar * pValue	[out] A pointer to converted NChar [1005].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.43 NCharTryParseVN Function

Converts a NString [678] to a NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NCharTryParseVN(HNString hValue, const NChar * szFormat, NChar * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to convert.
const NChar * szFormat	[in] Pointer to format string.
NChar * pValue	[out] A pointer to converted NChar [1005].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.44 NComplexToString Function

Converts NComplex [1005] to NString [678].

C++

```
NResult N_API NComplexToString(const NComplex * pValue, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
const NComplex * pValue	[in] NComplex [1005] to be converted.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.45 NDoubleGetNaN Function

Retrieves a NDouble [1005] that is Not a Number (NaN).

C++

```
NDouble N_API NDoubleGetNaN();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.46 NDoubleGetNegativeInfinity Function

Retrieves a NDouble [1005] that is NegativeInfinity.

C++

```
NDouble N_API NDoubleGetNegativeInfinity();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.47 NDoubleGetPositiveInfinity Function

Retrieves a NDouble [1005] that is PositiveInfinity.

C++

```
NDouble N_API NDoubleGetPositiveInfinity();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.48 NDoubleIsInfinity Function

Checks if NDouble [1005] evaluates to NegativeInfinity or PositiveInfinity.

C++

```
NBool N_API NDoubleIsInfinity(NDouble value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NDouble value	[in] NDouble [1005] to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.49 NDoubleIsNaN Function

Checks if NDouble [1005] is Not a Number (NaN).

C++

```
NBool N_API NDoubleIsNaN(NDouble value);
```

Parameters

Parameters	Description
NDouble value	[in] NDouble [1005] to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.50 NDoubleIsNegativeInfinity Function

Checks if NDouble [1005] evaluates to NegativeInfinity.

C++

```
NBool N_API NDoubleIsNegativeInfinity(NDouble value);
```

Parameters

Parameters	Description
NDouble value	[in] NDouble [1005] to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.51 NDoubleIsPositiveInfinity Function

Checks if NDouble [1005] evaluates to PositiveInfinity

C++

```
NBool N_API NDoubleIsPositiveInfinity(NDouble value);
```

Parameters

Parameters	Description
NDouble value	[in] NDouble [1005] to be checked.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.52 NDoubleParseN Function

Converts a NString [678] into NDouble [1005].

C++

```
NResult N_API NDoubleParseN(HNString hValue, HNString hFormat, NDouble * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NDouble * pValue	[out] A pointer to the converted NDouble [1005].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.53 NDoubleParseStrOrChars Function

Converted an array of NChars to NDouble [1005].

C++

```
NResult N_API NDoubleParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NDouble * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NDouble * pValue	[out] A pointer to the converted NDouble [1005].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.54 NDoubleParseVN Function

Converts a NString [678] into NDouble [1005].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NDoubleParseVN(HNString hValue, const NChar * szFormat, NDouble * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NDouble * pValue	[out] A pointer to the converted NDouble [1005].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.55 NDoubleToChars Function

Converts NDouble [1005] to an array of NChars.

C++

```
NResult N_API NDoubleToChars(NDouble value, const NChar * szFormat, NChar * arValue, NInt
valueLength);
```

Parameters

Parameters	Description
NDouble value	[in] NDouble [1005] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NChar * arValue	[out] A pointer to the converted array of NChars.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.56 NDoubleToString Function

Converts NDouble [1005] to NString [678].

C++

```
NResult N_API NDoubleToString(NDouble value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NDouble value	[in] NDouble [1005] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.57 NDoubleTryParseN Function

Converts a NString [678] to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NDoubleTryParseN(HNString hValue, HNString hFormat, NDouble * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NDouble * pValue	[out] A pointer to the converted NDouble [1005].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.58 NDoubleTryParseStrOrChars Function

Converts an array of NChars to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NDoubleTryParseStrOrChars(const NChar * arValue, NInt valueLength, const
NChar * szFormat, NDouble * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NDouble * pValue	[out] A pointer to the converted NDouble [1005].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.59 NDoubleTryParseVN Function

Converts a NString [678] to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NDoubleTryParseVN(HNString hValue, const NChar * szFormat, NDouble * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NDouble * pValue	[out] A pointer to the converted NDouble [1005].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.60 NGuidParse Function

Converts a NChar [1005] to NGuid.

C++

```
NResult N_API NGuidParse(const NChar * szValue, const NChar * szFormat, NGuid * pValue);
```

Parameters

Parameters	Description
const NChar * szValue	[in] A pointer to NChar [1005] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NGuid * pValue	[out] A pointer to the converted NGuid.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.1.61 NGuidParseVN Function

Converts a NString [678] to NGuid.

C++

```
NResult N_API NGuidParseVN(HNString hValue, const NChar * szFormat, NGuid * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NGuid * pValue	[out] A pointer to the converted NGuid.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.62 NGuidToString Function

Converts a NGuid to NString [678].

C++

```
NResult N_API NGuidToString(const NGuid * pValue, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
const NGuid * pValue	[in] A pointer to NGuid to be converted.
const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.63 NGuidTryParse Function

Converts a NChar [1005] to NGuid. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NGuidTryParse(const NChar * szValue, const NChar * szFormat, NGuid * pValue,
NBool * pResult);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const NChar * szValue	[in] A pointer to NChar [1005] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NGuid * pValue	[out] A pointer to the converted NGuide.
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.64 NGuidTryParseVN Function

Converts a NString [678] to NGuid. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NGuidTryParseVN(HNString hValue, const NChar * szFormat, NGuid * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NGuid * pValue	[out] A pointer to the converted NGuide.
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.65 NIndexPairToString Function

Convertes NIndexPair [1006] to NString [678].

C++

```
NResult N_API NIndexPairToString(const NIndexPair * pValue, const NChar * szFormat,
HNString * phValue);
```

Parameters

Parameters	Description
const NIndexPair * pValue	[in] A pointer to NIndexPair [1006] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.66 NInt16ParseN Function

Convertes NString [678] to NInt16 [1006].

C++

```
NResult N_API NInt16ParseN(HNString hValue, HNString hFormat, NInt16 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NInt16 * pValue	[out] A pointer to the converted NInt16 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.67 NInt16ParseStrOrChars Function

Converts an array of NChars to NInt16 [1006].

C++

```
NResult N_API NInt16ParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NInt16 * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt16 * pValue	[out] A pointer to the converted NInt16 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.1.68 NInt16ParseVN Function

Convertes NString [678] to NInt16 [1006].

C++

```
NResult N_API NInt16ParseVN(HNString hValue, const NChar * szFormat, NInt16 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt16 * pValue	[out] A pointer to the converted NInt16 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.69 NInt16ToChars Function

Converts NInt16 [1006] to an array of NChars.

C++

```
NResult N_API NInt16ToChars(NInt16 value, const NChar * szFormat, NChar * arValue, NInt
valueLength);
```

Parameters

Parameters	Description
NInt16 value	[in] NInt16 [1006] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.70 NInt16ToString Function

Converts NInt16 [1006] to NString [678].

C++

```
NResult N_API NInt16ToString(NInt16 value, const NChar * szFormat, HNString * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NInt16 value	[in] NInt16 [1006] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.71 NInt16TryParseN Function

Converts NString [678] to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NInt16TryParseN(HNString hValue, HNString hFormat, NInt16 * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NInt16 * pValue	[out] A pointer to the converted NInt16 [1006].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.72 NInt16TryParseStrOrChars Function

Converts an array of NChars to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NInt16TryParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar
* szFormat, NInt16 * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt16 * pValue	[out] A pointer to the converted NInt16 [1006].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.73 NInt16TryParseVN Function

Converts NString [678] to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NInt16TryParseVN(HNString hValue, const NChar * szFormat, NInt16 * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt16 * pValue	[out] A pointer to the converted NInt16 [1006].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.74 NInt32ParseN Function

Converts NString [678] to NInt32 [1006].

C++

```
NResult N_API NInt32ParseN(HNString hValue, HNString hFormat, NInt32 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NInt32 * pValue	[out] A pointer to the converted NInt32 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.1.75 NInt32ParseStrOrChars Function

Converts an array of NChars to NInt32 [1006].

C++

```
NResult N_API NInt32ParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NInt32 * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt32 * pValue	[out] A pointer to the converted NInt32 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.76 NInt32ParseVN Function

Converts NString [678] to NInt32 [1006].

C++

```
NResult N_API NInt32ParseVN(HNString hValue, const NChar * szFormat, NInt32 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt32 * pValue	[out] A pointer to the converted NInt32 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.77 NInt32ToChars Function

Converts NInt32 [1006] to an array of NChars.

C++

```
NResult N_API NInt32ToChars(NInt32 value, const NChar * szFormat, NChar * arValue, NInt
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
valueLength);
```

Parameters

Parameters	Description
NInt32 value	[in] NInt32 [1006] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.78 NInt32ToString Function

Converts NInt32 [1006] to NString [678].

C++

```
NResult N_API NInt32ToString(NInt32 value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NInt32 value	[in] NInt32 [1006] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.79 NInt32TryParseN Function

Converts NString [678] to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NInt32TryParseN(HNString hValue, HNString hFormat, NInt32 * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NInt32 * pValue	[out] A pointer to the converted NInt32 [1006].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.80 NInt32TryParseStrOrChars Function

Converts an array of NChars to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NInt32TryParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar
* szFormat, NInt32 * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt32 * pValue	[out] A pointer to the converted NInt32 [1006].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.81 NInt32TryParseVN Function

Converts NString [678] to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NInt32TryParseVN(HNString hValue, const NChar * szFormat, NInt32 * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt32 * pValue	[out] A pointer to the converted NInt32 [1006].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.82 NInt64ParseN Function

Converts NString [678] to NInt64 [1006].

C++

```
NResult N_API NInt64ParseN(HNString hValue, HNString hFormat, NInt64 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NInt64 * pValue	[out] A pointer to the converted NInt64 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.83 NInt64ParseStrOrChars Function

Converts an array of NChars to NInt64 [1006].

C++

```
NResult N_API NInt64ParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NInt64 * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt64 * pValue	[out] A pointer to the converted NInt64 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.84 NInt64ParseVN Function

Converts NString [678] to NInt64 [1006].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NInt64ParseVN(HNString hValue, const NChar * szFormat, NInt64 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt64 * pValue	[out] A pointer to the converted NInt64 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.85 NInt64ToChars Function

Converts NInt64 [1006] to an array of NChars.

C++

```
NResult N_API NInt64ToChars(NInt64 value, const NChar * szFormat, NChar * arValue, NInt
valueLength);
```

Parameters

Parameters	Description
NInt64 value	[in] NInt64 [1006] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.86 NInt64ToString Function

Converts NInt64 [1006] to NString [678].

C++

```
NResult N_API NInt64ToString(NInt64 value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NInt64 value	[in] NInt64 [1006] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.87 NInt64TryParseN Function

Converts NString [678] to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NInt64TryParseN(HNString hValue, HNString hFormat, NInt64 * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NInt64 * pValue	[out] A pointer to the converted NInt64 [1006].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.88 NInt64TryParseStrOrChars Function

Converts an array of NChars to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NInt64TryParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar
* szFormat, NInt64 * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt64 * pValue	[out] A pointer to the converted NInt64 [1006].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.89 NInt64TryParseVN Function

Converts NString [678] to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NInt64TryParseVN(HNString hValue, const NChar * szFormat, NInt64 * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt64 * pValue	[out] A pointer to the converted NInt64 [1006].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.90 NInt8ParseN Function

Converts NString [678] to NInt8 [1007].

C++

```
NResult N_API NInt8ParseN(HNString hValue, HNString hFormat, NInt8 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NInt8 * pValue	[out] A pointer to the converted NInt8 [1007].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.1.91 NInt8ParseStrOrChars Function

Converts an array of NChars to NInt8 [1007].

C++

```
NResult N_API NInt8ParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NInt8 * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt8 * pValue	[out] A pointer to the converted NInt8 [1007].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.92 NInt8ParseVN Function

Converts NString [678] to NInt8 [1007].

C++

```
NResult N_API NInt8ParseVN(HNString hValue, const NChar * szFormat, NInt8 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt8 * pValue	[out] A pointer to the converted NInt8 [1007].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.93 NInt8ToChars Function

Converts NInt8 [1007] to an array of NChars.

C++

```
NResult N_API NInt8ToChars(NInt8 value, const NChar * szFormat, NChar * arValue, NInt
valueLength);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NInt8 value	[in] NInt8 [1007] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.94 NInt8ToString Function

Converts NInt8 [1007] to NString [678].

C++

```
NResult N_API NInt8ToString(NInt8 value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NInt8 value	[in] NInt8 [1007] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.95 NInt8TryParseN Function

Converts NString [678] to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NInt8TryParseN(HNString hValue, HNString hFormat, NInt8 * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NInt8 * pValue	[out] A pointer to the converted NInt8 [1007].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.96 NInt8TryParseStrOrChars Function

Converts an array of NChars to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NInt8TryParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar
* szFormat, NInt8 * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt8 * pValue	[out] A pointer to the converted NInt8 [1007].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.97 NInt8TryParseVN Function

Converts NString [678] to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NInt8TryParseVN(HNString hValue, const NChar * szFormat, NInt8 * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NInt8 * pValue	[out] A pointer to the converted NInt8 [1007].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.98 NNameStringPairCopy Function

Creates a copy of string pair structure.

C++

```
NResult N_API NNameStringPairCopy(const struct NNameStringPair_ * pSrcValue, struct
NNameStringPair_ * pDstValue);
```

Parameters

Parameters	Description
const struct NNameStringPair_ * pSrcValue	[in] Pointer to the original string pair structure.
struct NNameStringPair_ * pDstValue	[out] Pointer to the copied string pair structure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.99 NNameStringPairCreate Function

Creates a new string pair structure.

C++

```
NResult N_API NNameStringPairCreate(const NChar * szKey, const NChar * szValue,
NNameStringPair * pValue);
```

Parameters

Parameters	Description
const NChar * szKey	[in] Pointer to the key string for the pair.
const NChar * szValue	[in] Pointer to the value string for the pair.
NNameStringPair * pValue	[out] Pointer to the new string pair structure.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.100 NNameStringPairDispose Function

Disposes of the specified string pair structure.

C++

```
NResult N_API NNameStringPairDispose(struct NNameStringPair_ * pValue);
```

Parameters

Parameters	Description
struct NNameStringPair_ * pValue	[in/out] Pointer to the string pair structure to be disposed of.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.101 NNameStringPairSet Function

Sets the value of the specified string pair structure.

C++

```
NResult N_API NNameStringPairSet(const struct NNameStringPair_ * pSrcValue, struct
NNameStringPair_ * pDstValue);
```

Parameters

Parameters	Description
const struct NNameStringPair_ * pSrcValue	[in] Pointer to string pair structure containing new value.
struct NNameStringPair_ * pDstValue	[in/out] Pointer to the string pair structure to be changed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.102 NOSFamilyGetCurrent Function

Retrieves the current operating system family.

C++

```
NOSFamily N_API NOSFamilyGetCurrent();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.103 NPointerParseN Function

Converts NString [678] to NPointer.

C++

```
NResult N_API NPointerParseN(HNString hValue, HNString hFormat, void * * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to format string.
void * * pValue	[out] A pointer to the converted NPointer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.1.104 NPointerParseStrOrChars Function

Converts an array of NChars to NPointer.

C++

```
NResult N_API NPointerParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar
* szFormat, void * * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
void * * pValue	[out] A pointer to the converted NPointer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.105 NPointerParseVN Function

Converts NString [678] to NPointer.

C++

```
NResult N_API NPointerParseVN(HNString hValue, const NChar * szFormat, void * * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
void * * pValue	[out] A pointer to the converted NPointer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.106 NPointerToChars Function

Converts NPointer to an array of NChars.

C++

```
NResult N_API NPointerToChars(const void * value, const NChar * szFormat, NChar * arValue,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
NInt valueLength);
```

Parameters

Parameters	Description
const void * value	[in] NPointer to be converted.
const NChar * szFormat	[in] Pointer to format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.107 NPointerToString Function

Converts NPointer to NString [678].

C++

```
NResult N_API NPointerToString(const void * value, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
const void * value	[in] NPointer to be converted.
const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.108 NPointerTryParseN Function

Converts NString [678] to NPointer. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NPointerTryParseN(HNString hValue, HNString hFormat, void * * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to format string.
void * * pValue	[out] A pointer to the converted NPointer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.109 NPointerTryParseStrOrChars Function

Converts an array of NChars to NPointer. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NPointerTryParseStrOrChars(const NChar * arValue, NInt valueLength, const
NChar * szFormat, void * * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
void * * pValue	[out] A pointer to the converted NPointer.
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.110 NPointerTryParseVN Function

Converts NString [678] to NPointer. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NPointerTryParseVN(HNString hValue, const NChar * szFormat, void * * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
void * * pValue	[out] A pointer to the converted NPointer.
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.111 NRangeToString Function

Converts NRange to NString [678].

C++

```
NResult N_API NRangeToString(const NRange * pValue, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
const NRange * pValue	[in] A pointer to NRange to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.112 NRationalToString Function

Converts NRational [1007] to NString [678].

C++

```
NResult N_API NRationalToString(const NRational * pValue, const NChar * szFormat, HNString
* phValue);
```

Parameters

Parameters	Description
const NRational * pValue	[in] A pointer to NRational [1007] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.113 NResultParseN Function

Converts NString [678] to NResult [1007].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NResultParseN(HNString hValue, HNString hFormat, NResult * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to format string.
NResult * pValue	[out] A pointer to the converted NResult [1007].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.114 NResultParseStrOrChars Function

Converts an array of NChars to NResult [1007].

C++

```
NResult N_API NResultParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NResult * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NResult * pValue	[out] A pointer to the converted NResult [1007].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.115 NResultParseVN Function

Converts NString [678] to NResult [1007].

C++

```
NResult N_API NResultParseVN(HNString hValue, const NChar * szFormat, NResult * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NResult * pValue	[out] A pointer to the converted NResult [1007].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.116 NResultToChars Function

Converts NResult [1007] to an array of NChars.

C++

```
NResult N_API NResultToChars(NResult value, const NChar * szFormat, NChar * arValue, NInt valueLength);
```

Parameters

Parameters	Description
NResult value	[in] NResult [1007] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.117 NResultToString Function

Converts NResult [1007] to NString [678].

C++

```
NResult N_API NResultToString(NResult value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NResult value	[in] NResult [1007] to be converted.
const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.1.118 NResultTryParseN Function

Converts NString [678] to NResult [1007]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NResultTryParseN(HNString hValue, HNString hFormat, NResult * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to format string.
NResult * pValue	[out] A pointer to the converted NResult [1007].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.119 NResultTryParseStrOrChars Function

Converts an array of NChars to NResult [1007]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NResultTryParseStrOrChars(const NChar * arValue, NInt valueLength, const
NChar * szFormat, NResult * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NResult * pValue	[out] A pointer to the converted NResult [1007].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.120 NResultTryParseVN Function

Converts NString [678] to NResult [1007]. A NBool [1004] value indicates whether the conversion succeeded.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NResultTryParseVN(HNString hValue, const NChar * szFormat, NResult * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NResult * pValue	[out] A pointer to the converted NResult [1007].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.121 NSingleGetNaN Function

Retrieves a NSingle [1008] that is Not a Number (NaN).

C++

```
NSingle N_API NSingleGetNaN();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.122 NSingleGetNegativeInfinity Function

Retrieves a NSingle [1008] that is NegativeInfinity.

C++

```
NSingle N_API NSingleGetNegativeInfinity();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.123 NSingleGetPositiveInfinity Function

Retrieves a NSingle [1008] that is PositiveInfinity.

C++

```
NSingle N_API NSingleGetPositiveInfinity();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.1.124 NSingleIsInfinity Function

Checks if NSingle [1008] evaluates to NegativeInfinity or PositiveInfinity.

C++

```
NBool N_API NSingleIsInfinity(NSingle value);
```

Parameters

Parameters	Description
NSingle value	[in] NDouble [1005] to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.125 NSingleIsNaN Function

Checks if NSingle [1008] Not a Number (NaN).

C++

```
NBool N_API NSingleIsNaN(NSingle value);
```

Parameters

Parameters	Description
NSingle value	[in] NDouble [1005] to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.126 NSingleIsNegativeInfinity Function

Checks if NSingle [1008] evaluates to NegativeInfinity.

C++

```
NBool N_API NSingleIsNegativeInfinity(NSingle value);
```

Parameters

Parameters	Description
NSingle value	[in] NDouble [1005] to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.1.127 NSingleIsPositiveInfinity Function

Checks if NSingle [1008] evaluates to PositiveInfinity.

C++

```
NBool N_API NSingleIsPositiveInfinity(NSingle value);
```

Parameters

Parameters	Description
NSingle value	[in] NDouble [1005] to be checked.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.128 NSingleParseN Function

Converts NString [678] to NSingle [1008].

C++

```
NResult N_API NSingleParseN(HNString hValue, HNString hFormat, NSingle * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NSingle * pValue	[out] A pointer to the converted NSingle [1008].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.129 NSingleParseStrOrChars Function

Converts an array of NChars to NSingle [1008].

C++

```
NResult N_API NSingleParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NSingle * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NSingle * pValue	[out] A pointer to the converted NSingle [1008].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.130 NSingleParseVN Function

Converts NString [678] to NSingle [1008].

C++

```
NResult N_API NSingleParseVN(HNString hValue, const NChar * szFormat, NSingle * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NSingle * pValue	[out] A pointer to the converted NSingle [1008].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.131 NSingleToChars Function

Converts NSingle [1008] to an array of NChars.

C++

```
NResult N_API NSingleToChars(NSingle value, const NChar * szFormat, NChar * arValue, NInt
valueLength);
```

Parameters

Parameters	Description
NSingle value	[in] NSingle [1008] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.1.132 NSingleToString Function

Converts NSingle [1008] to NString [678].

C++

```
NResult N_API NSingleToString(NSingle value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NSingle value	[in] NSingle [1008] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.133 NSingleTryParseN Function

Converts NString [678] to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NSingleTryParseN(HNString hValue, HNString hFormat, NSingle * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NSingle * pValue	[out] A pointer to the converted NSingle [1008].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.134 NSingleTryParseStrOrChars Function

Converts an array of NChars to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NSingleTryParseStrOrChars(const NChar * arValue, NInt valueLength, const
NChar * szFormat, NSingle * pValue, NBool * pResult);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NSingle * pValue	[out] A pointer to the converted NSingle [1008].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.135 NSingleTryParseVN Function

Converts NString [678] to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NSingleTryParseVN(HNString hValue, const NChar * szFormat, NSingle * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NSingle * pValue	[out] A pointer to the converted NSingle [1008].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.136 NSizeTypeParseN Function

Converts NString [678] to NSizeType [1008].

C++

```
NResult N_API NSizeTypeParseN(HNString hValue, HNString hFormat, NSizeType * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to format string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSizeType * pValue	[out] A pointer to the converted NSizeType [1008].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.137 NSizeTypeParseStrOrChars Function

Converts an array of NChars to NSizeType [1008].

C++

```
NResult N_API NSizeTypeParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar
* szFormat, NSizeType * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NSizeType * pValue	[out] A pointer to the converted NSizeType [1008].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.138 NSizeTypeParseVN Function

Converts NString [678] to NSizeType [1008].

C++

```
NResult N_API NSizeTypeParseVN(HNString hValue, const NChar * szFormat, NSizeType * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NSizeType * pValue	[out] A pointer to the converted NSizeType [1008].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.1.139 NSizeTypeToChars Function

Converts NSizeType [1008] to an array of NChars.

C++

```
NResult N_API NSizeTypeToChars(NSizeType value, const NChar * szFormat, NChar * arValue,
NInt valueLength);
```

Parameters

Parameters	Description
NSizeType value	[in] NSizeType [1008] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.140 NSizeTypeToString Function

Converts NSizeType [1008] to NString [678].

C++

```
NResult N_API NSizeTypeToString(NSizeType value, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
NSizeType value	[in] NSizeType [1008] to be converted.
const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.141 NSizeTypeTryParseN Function

Converts NString [678] to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NSizeTypeTryParseN(HNString hValue, HNString hFormat, NSizeType * pValue,
NBool * pResult);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to format string.
NSizeType * pValue	[out] A pointer to the converted NSizeType [1008].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.142 NSizeTypeTryParseStrOrChars Function

Converts an array of NChars to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NSizeTypeTryParseStrOrChars(const NChar * arValue, NInt valueLength, const
NChar * szFormat, NSizeType * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NSizeType * pValue	[out] A pointer to the converted NSizeType [1008].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.143 NSizeTypeTryParseVN Function

Converts NString [678] to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NSizeTypeTryParseVN(HNString hValue, const NChar * szFormat, NSizeType *
pValue, NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const NChar * szFormat	[in] Pointer to format string.
NSizeType * pValue	[out] A pointer to the converted NSizeType [1008].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.144 NSSizeTypeParseN Function

Converts NString [678] to NSSizeType [1009].

C++

```
NResult N_API NSSizeTypeParseN(HNString hValue, HNString hFormat, NSSizeType * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to format string.
NSSizeType * pValue	[out] A pointer to the converted NSSizeType [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.145 NSSizeTypeParseStrOrChars Function

Converts an array of NChars to NSSizeType [1009].

C++

```
NResult N_API NSSizeTypeParseStrOrChars(const NChar * arValue, NInt valueLength, const
NChar * szFormat, NSSizeType * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NSSizeType * pValue	[out] A pointer to the converted NSSizeType [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.146 NSSizeTypeParseVN Function

Converts NString [678] to NSSizeType [1009].

C++

```
NResult N_API NSSizeTypeParseVN(HNString hValue, const NChar * szFormat, NSSizeType *
pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NSSizeType * pValue	[out] A pointer to the converted NSSizeType [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.147 NSSizeTypeToChars Function

Converts NSSizeType [1009] to an array of NChars.

C++

```
NResult N_API NSSizeTypeToChars(NSSizeType value, const NChar * szFormat, NChar * arValue,
NInt valueLength);
```

Parameters

Parameters	Description
NSSizeType value	[in] NSSizeType [1009] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.1.148 NSSizeTypeToString Function

Converts NSSizeType [1009] to NString [678].

C++

```
NResult N_API NSSizeTypeToString(NSSizeType value, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
NSSizeType value	[in] NSSizeType [1009] to be converted.
const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.149 NSSizeTypeTryParseN Function

Converts NString [678] to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NSSizeTypeTryParseN(HNString hValue, HNString hFormat, NSSizeType * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to format string.
NSSizeType * pValue	[out] A pointer to the converted NSSizeType [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.150 NSSizeTypeTryParseStrOrChars Function

Converts an array of NChars to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NSSizeTypeTryParseStrOrChars(const NChar * arValue, NInt valueLength, const
NChar * szFormat, NSSizeType * pValue, NBool * pResult);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NSSizeType * pValue	[out] A pointer to the converted NSSizeType [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.151 NSSizeTypeTryParseVN Function

Converts NString [678] to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NSSizeTypeTryParseVN(HNString hValue, const NChar * szFormat, NSSizeType *
pValue, NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NSSizeType * pValue	[out] A pointer to the converted NSSizeType [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.152 NTypesGetPointer Function

Retrieves a pointer to object.

C++

```
void * N_API_PTR_RET NTypesGetPointer(void * value);
```

Parameters

Parameters	Description
void * value	[out] A pointer to object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.1.153 NUInt16ParseN Function

Converts NString [678] to NUInt16 [1009].

C++

```
NResult N_API NUInt16ParseN(HNString hValue, HNString hFormat, NUInt16 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NUInt16 * pValue	[out] A pointer to the converted NUInt16 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.154 NUInt16ParseStrOrChars Function

Converts an array of NChars to NUInt16 [1009].

C++

```
NResult N_API NUInt16ParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NUInt16 * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt16 * pValue	[out] A pointer to the converted NUInt16 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.155 NUInt16ParseVN Function

Converts NString [678] to NUInt16 [1009].

C++

```
NResult N_API NUInt16ParseVN(HNString hValue, const NChar * szFormat, NUInt16 * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt16 * pValue	[out] A pointer to the converted NUInt16 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.156 NUInt16ToChars Function

Converts NUInt16 [1009] to an array of NChars.

C++

```
NResult N_API NUInt16ToChars(NUInt16 value, const NChar * szFormat, NChar * arValue, NInt valueLength);
```

Parameters

Parameters	Description
NUInt16 value	[in] NUInt16 [1009] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.157 NUInt16ToString Function

Converts NUInt16 [1009] to NString [678].

C++

```
NResult N_API NUInt16ToString(NUInt16 value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NUInt16 value	[in] NUInt16 [1009] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.158 NUInt16TryParseN Function

Converts NString [678] to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NUInt16TryParseN(HNString hValue, HNString hFormat, NUInt16 * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NUInt16 * pValue	[out] A pointer to the converted NUInt16 [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.159 NUInt16TryParseStrOrChars Function

Converts an array of NChars to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NUInt16TryParseStrOrChars(const NChar * arValue, NInt valueLength, const
NChar * szFormat, NUInt16 * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt16 * pValue	[out] A pointer to the converted NUInt16 [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.1.160 NUInt16TryParseVN Function

Converts NString [678] to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NUInt16TryParseVN(HNString hValue, const NChar * szFormat, NUInt16 * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt16 * pValue	[out] A pointer to the converted NUInt16 [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.161 NUInt32ParseN Function

Converts NString [678] to NUInt32 [1009].

C++

```
NResult N_API NUInt32ParseN(HNString hValue, HNString hFormat, NUInt32 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NUInt32 * pValue	[out] A pointer to the converted NUInt32 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.162 NUInt32ParseStrOrChars Function

Converts an array of NChars to NUInt32 [1009].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NUInt32ParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NUInt32 * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt32 * pValue	[out] A pointer to the converted NUInt32 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.163 NUInt32ParseVN Function

Converts NString [678] to NUInt32 [1009].

C++

```
NResult N_API NUInt32ParseVN(HNString hValue, const NChar * szFormat, NUInt32 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt32 * pValue	[out] A pointer to the converted NUInt32 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.164 NUInt32ToChars Function

Converts NUInt32 [1009] to an array of NChars.

C++

```
NResult N_API NUInt32ToChars(NUInt32 value, const NChar * szFormat, NChar * arValue, NInt
valueLength);
```

Parameters

Parameters	Description
NUInt32 value	[in] NUInt32 [1009] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.165 NUInt32ToString Function

Converts NUInt32 [1009] to NString [678].

C++

```
NResult N_API NUInt32ToString(NUInt32 value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NUInt32 value	[in] NUInt32 [1009] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.166 NUInt32TryParseN Function

Converts NString [678] to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NUInt32TryParseN(HNString hValue, HNString hFormat, NUInt32 * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NUInt32 * pValue	[out] A pointer to the converted NUInt32 [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.1.167 NUInt32TryParseStrOrChars Function

Converts an array of NChars to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NUInt32TryParseStrOrChars(const NChar * arValue, NInt valueLength, const
NChar * szFormat, NUInt32 * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt32 * pValue	[out] A pointer to the converted NUInt32 [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.168 NUInt32TryParseVN Function

Converts NString [678] to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NUInt32TryParseVN(HNString hValue, const NChar * szFormat, NUInt32 * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt32 * pValue	[out] A pointer to the converted NUInt32 [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.1.169 NUInt64ParseN Function

Converts NString [678] to NUInt64 [1009].

C++

```
NResult N_API NUInt64ParseN(HNString hValue, HNString hFormat, NUInt64 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NUInt64 * pValue	[out] A pointer to the converted NUInt64 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.170 NUInt64ParseStrOrChars Function

Converts an array of NChars to NUInt64 [1009].

C++

```
NResult N_API NUInt64ParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NUInt64 * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt64 * pValue	[out] A pointer to the converted NUInt64 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.171 NUInt64ParseVN Function

Converts NString [678] to NUInt64 [1009].

C++

```
NResult N_API NUInt64ParseVN(HNString hValue, const NChar * szFormat, NUInt64 * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt64 * pValue	[out] A pointer to the converted NUInt64 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.172 NUInt64ToChars Function

Converts NUInt64 [1009] to an array of NChars.

C++

```
NResult N_API NUInt64ToChars(NUInt64 value, const NChar * szFormat, NChar * arValue, NInt valueLength);
```

Parameters

Parameters	Description
NUInt64 value	[in] NUInt64 [1009] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.173 NUInt64ToString Function

Converts NUInt64 [1009] to NString [678].

C++

```
NResult N_API NUInt64ToString(NUInt64 value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NUInt64 value	[in] NUInt64 [1009] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.174 NUInt64TryParseN Function

Converts NString [678] to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NUInt64TryParseN(HNString hValue, HNString hFormat, NUInt64 * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NUInt64 * pValue	[out] A pointer to the converted NUInt64 [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.175 NUInt64TryParseStrOrChars Function

Converts an array of NChars to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NUInt64TryParseStrOrChars(const NChar * arValue, NInt valueLength, const
NChar * szFormat, NUInt64 * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt64 * pValue	[out] A pointer to the converted NUInt64 [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.1.176 NUInt64TryParseVN Function

Converts NString [678] to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NUInt64TryParseVN(HNString hValue, const NChar * szFormat, NUInt64 * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt64 * pValue	[out] A pointer to the converted NUInt64 [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.177 NUInt8ParseN Function

Converts NString [678] to NUInt8 [1009].

C++

```
NResult N_API NUInt8ParseN(HNString hValue, HNString hFormat, NUInt8 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NUInt8 * pValue	[out] A pointer to the converted NUInt8 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.178 NUInt8ParseStrOrChars Function

Converts an array of NChars to NUInt8 [1009].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NUInt8ParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NUInt8 * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt8 * pValue	[out] A pointer to the converted NUInt8 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.179 NUInt8ParseVN Function

Converts NString [678] to NUInt8 [1009].

C++

```
NResult N_API NUInt8ParseVN(HNString hValue, const NChar * szFormat, NUInt8 * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt8 * pValue	[out] A pointer to the converted NUInt8 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.180 NUInt8ToChars Function

Converts NUInt8 [1009] to an array of NChars.

C++

```
NResult N_API NUInt8ToChars(NUInt8 value, const NChar * szFormat, NChar * arValue, NInt
valueLength);
```

Parameters

Parameters	Description
NUInt8 value	[in] NUInt8 [1009] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.181 NUInt8ToString Function

Converts NUInt8 [1009] to NString [678].

C++

```
NResult N_API NUInt8ToString(NUInt8 value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NUInt8 value	[in] NUInt8 [1009] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.182 NUInt8TryParseN Function

Converts NString [678] to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NUInt8TryParseN(HNString hValue, HNString hFormat, NUInt8 * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to numeric format string.
NUInt8 * pValue	[out] A pointer to the converted NUInt8 [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.1.183 NUInt8TryParseStrOrChars Function

Converts an array of NChars to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NUInt8TryParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar
* szFormat, NUInt8 * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt8 * pValue	[out] A pointer to the converted NUInt8 [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.184 NUInt8TryParseVN Function

Converts NString [678] to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NUInt8TryParseVN(HNString hValue, const NChar * szFormat, NUInt8 * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
NUInt8 * pValue	[out] A pointer to the converted NUInt8 [1009].
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.1.185 NURationalToString Function

Converts NURational [1010] to NString [678].

C++

```
NResult N_API NURationalToString(const NURational * pValue, const NChar * szFormat,
HNString * phValue);
```

Parameters

Parameters	Description
const NURational * pValue	[in] A pointer to NURational [1010] to be converted.
const NChar * szFormat	[in] Pointer to numeric format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.186 NVersionRangeContains Function

Checks if NVersionRange [1011] contains a certain version.

C++

```
NBool N_API NVersionRangeContains(NVersionRange_ value, NVersion_ version);
```

Parameters

Parameters	Description
NVersionRange_ value	[in] NVersionRange [1011] to be checked.
NVersion_ version	[in] NVersion [1010] that may be contained in NVersionRange [1011].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.187 NVersionRangeContainsRange Function

Checks if NVersionRange [1011] contains a certain version range.

C++

```
NBool N_API NVersionRangeContainsRange(NVersionRange_ value, NVersionRange_ versionRange);
```

Parameters

Parameters	Description
NVersionRange_ value	[in] NVersionRange [1011] to be checked.
NVersionRange_ versionRange	[in] The version range that may be contained in NVersionRange [1011].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.188 NVersionRangeIntersect Function

Replaces NVersionRange [1011] with the intersection of itself and another NVersionRange [1011].

C++

```
NVersionRange_ N_API NVersionRangeIntersect(NVersionRange_ value1, NVersionRange_ value2);
```

Parameters

Parameters	Description
NVersionRange_ value1	[in] Basic NVersionRange [1011].
NVersionRange_ value2	[in] Intersecting NVersionRange [1011].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.189 NVersionRangeIntersectsWith Function

Returns third NVersionRange [1011] that represents the intersection of two other NVersionRanges. If there is no intersection, an empty NVersionRange [1011] is returned.

C++

```
NBool N_API NVersionRangeIntersectsWith(NVersionRange_ value, NVersionRange_ otherValue);
```

Parameters

Parameters	Description
NVersionRange_ value	[in] First NVersionRange [1011] to intersect.
NVersionRange_ otherValue	[in] Second NVersionRange [1011] to intersect.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.1.190 NVersionRangeToString Function

Converts NVersionRange [1011] to NString [678].

C++

```
NResult N_API NVersionRangeToString(NVersionRange value, const NChar * szFormat, HNString *
phValue);
```

Parameters

Parameters	Description
NVersionRange value	[in] NVersionRange [1011] to be converted.
const NChar * szFormat	[in] Pointer to format string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.191 NVersionToString Function

Converts NVersion [1010] to NString [678].

C++

```
NResult N_API NVersionToString(NVersion value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NVersion value	[in] NVersion [1010] to be converted.
const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.192 NWCharParseN Function

Converts NString [678] to NWChar.

C++

```
NResult N_API NWCharParseN(HNString hValue, HNString hFormat, NWChar * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to format string.
NWChar * pValue	[out] A pointer to the converted NWChar.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.1.193 NWCharParseStrOrChars Function

Converts an array of NChars to NWChar.

C++

```
NResult N_API NWCharParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar *
szFormat, NWChar * pValue);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NWChar * pValue	[out] A pointer to the converted NWChar.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.194 NWCharParseVN Function

Converts NString [678] to NWChar.

C++

```
NResult N_API NWCharParseVN(HNString hValue, const NChar * szFormat, NWChar * pValue);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NWChar * pValue	[out] A pointer to the converted NWChar.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.195 NWCharToChars Function

Converts NWChar to an array of NChars.

C++

```
NResult N_API NWCharToChars(NWChar value, const NChar * szFormat, NChar * arValue, NInt
valueLength);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NWChar value	[in] NWChar to be converted.
const NChar * szFormat	[in] Pointer to format string.
NChar * arValue	[out] A pointer to the converted NChar [1005] array.
NInt valueLength	[in] Length of the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.196 NWCharToString Function

Converts NWChar to NString [678].

C++

```
NResult N_API NWCharToString(NWChar value, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
NWChar value	[in] NWChar to be converted.
const NChar * szFormat	[in] Pointer to format string.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.197 NWCharTryParseN Function

Converts NString [678] to NWChar. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NWCharTryParseN(HNString hValue, HNString hFormat, NWChar * pValue, NBool *
pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
HNString hFormat	[in] Handle to format string.
NWChar * pValue	[out] A pointer to the converted NWChar.
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.198 NWCharTryParseStrOrChars Function

Converts an array of NChars to NWChar. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NWCharTryParseStrOrChars(const NChar * arValue, NInt valueLength, const NChar
* szFormat, NWChar * pValue, NBool * pResult);
```

Parameters

Parameters	Description
const NChar * arValue	[in] A pointer to NChar [1005] array.
NInt valueLength	[in] Length of the array.
const NChar * szFormat	[in] Pointer to format string.
NWChar * pValue	[out] A pointer to the converted NWChar.
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.1.199 NWCharTryParseVN Function

Converts NString [678] to NWChar. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
NResult N_API NWCharTryParseVN(HNString hValue, const NChar * szFormat, NWChar * pValue,
NBool * pResult);
```

Parameters

Parameters	Description
HNString hValue	[in] A handle to NString [678] to be converted.
const NChar * szFormat	[in] Pointer to format string.
NWChar * pValue	[out] A pointer to the converted NWChar.
NBool * pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

For format specifications see Format Specifications [685].

Module

NTypes Unit [905]

7.1.2.64.2 Structs, Records, Enums

7.1.2.64.2.1 NComplex_ Structure

Represents a complex number value.

C++

```
struct NComplex_ {
  NDouble Real;
  NDouble Imaginary;
};
```

Members

Members	Description
NDouble Real;	Real part of the complex number.
NDouble Imaginary;	Imaginary part of the complex number.

Module

NTypes Unit [905]

7.1.2.64.2.2 NIndexPair_ Structure

Represents a pair of indexes.

C++

```
struct NIndexPair_ {
  NInt Index1;
  NInt Index2;
};
```

Members

Members	Description
NInt Index1;	First index of this NIndexPair [1006].
NInt Index2;	Second index of this NIndexPair [1006].

Module

NTypes Unit [905]

7.1.2.64.2.3 NNameStringPair_ Structure

Represents a pair of name strings.

C++

```
struct NNameStringPair_ {
  HNString hKey;
  HNString hValue;
};
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Members

Members	Description
HNString hKey;	Key string of the name string pair.
HNString hValue;	Value string of the name string pair.

Module

NTypes Unit [905]

7.1.2.64.2.4 NRational_ Structure

Represents a signed rational number.

C++

```
struct NRational_ {
  NInt Numerator;
  NInt Denominator;
};
```

Members

Members	Description
NInt Numerator;	Numerator of this NRational [1007].
NInt Denominator;	Denominator of this NRational [1007].

Module

NTypes Unit [905]

7.1.2.64.2.5 NURational_ Structure

Represents an unsigned rational number.

C++

```
struct NURational_ {
  NUInt Numerator;
  NUInt Denominator;
};
```

Members

Members	Description
NUInt Numerator;	Numerator of this NURational [1010].
NUInt Denominator;	Denominator of this NURational [1010].

Module

NTypes Unit [905]

7.1.2.64.2.6 NAttributes Enumeration

Enumerates value attributes.

C++

```
typedef enum NAttributes_ {
  naNone = 0,
  naSignNeutral = 0x00000001,
  naSingleValue = 0x00000002,
  naArray = 0x00000004,
  naNullable = 0x00000008,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
    naLocal = 0x00000010,
    naSet = 0x00000040,
    naCache = 0x00000080,
    naDeprecated = 0x00000100,
    naStatic = 0x00002000,
    naPublic = 0x00004000,
    naMinValue = 0x00100000,
    naMaxValue = 0x00200000,
    naStdValues = 0x00400000,
    naStdValuesExclusive = 0x00800000,
    naNoRead = 0x01000000,
    naNoWrite = 0x02000000,
    naByRef = 0x04000000,
    naRetValue = 0x08000000,
    naDefaultValue = 0x10000000,
    naOptional = 0x20000000,
    naPromoteValue = 0x80000000
  } NAttributes;
```

Members

Members	Description
naNone = 0	Indicates no attributes.
naSignNeutral = 0x00000001	Indicates that unsigned integer value can be interpreted as signed and vice versa
naSingleValue = 0x00000002	Indicates that the value is a single value.
naArray = 0x00000004	Indicates that the value is an array.
naNullable = 0x00000008	Indicates that the value is nullable.
naLocal = 0x00000010	Indicates that the value is local.
naSet = 0x00000040	Indicates that several values (from standard ones) can be combined.
naCache = 0x00000080	Indicates that the object wrapper has to be cached.
naDeprecated = 0x00000100	Indicates that the value is deprecated.
naStatic = 0x00002000	Indicates that the value is static.
naPublic = 0x00004000	Indicates that the value is public.
naMinValue = 0x00100000	Indicates that the value is the minimum value.
naMaxValue = 0x00200000	Indicates that the value is the maximum value.
naStdValues = 0x00400000	Indicates that standard values are used.
naStdValuesExclusive = 0x00800000	Indicates that standard values are exclusive.
naNoRead = 0x01000000	Indicates that the value can't be read.
naNoWrite = 0x02000000	Indicates that the value can't be written.
naByRef = 0x04000000	Indicates that parameters are passed by reference.
naRetValue = 0x08000000	Indicates a return value.
naDefaultValue = 0x10000000	Indicates that the value is the default value.
naOptional = 0x20000000	Indicates that the value is optional.
naPromoteValue = 0x80000000	For NValueCreate [1099]/NArrayCreate [270] specifies that the created object takes ownership of the provided pointer.

Module

NTypes Unit [905]

7.1.2.64.2.7 NOSFamily Enumeration

Enumerates OS families.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
typedef enum NOSFamily_ {
  nosfNone = 0,
  nosfWindows = 1,
  nosfWindowsCE = 2,
  nosfWindowsPhone = 4,
  nosfMacOSX = 8,
  nosfIOS = 16,
  nosfLinux = 32,
  nosfEmbeddedLinux = 64,
  nosfAndroid = 128,
  nosfUnix = 256
} NOSFamily;
```

Members

Members	Description
nosfNone = 0	Indicates no OS.
nosfWindows = 1	Indicates Windows OS.
nosfWindowsCE = 2	Indicates Windows CE OS.
nosfWindowsPhone = 4	Indicates Windows Phone OS.
nosfMacOSX = 8	Indicates Mac OS X.
nosfIOS = 16	Indicates iOS.
nosfLinux = 32	Indicates Linux OS.
nosfEmbeddedLinux = 64	Indicates embedded Linux OS.
nosfAndroid = 128	Indicates Android OS.
nosfUnix = 256	Indicates Unix OS.

Module

NTypes Unit [905]

7.1.2.64.3 Types

7.1.2.64.3.1 NAChar Type

ANSI character (8-bit).

C++

```
typedef char NAChar;
```

Module

NTypes Unit [905]

7.1.2.64.3.2 NBool Type

Same as NBoolean [1004].

C++

```
typedef NBoolean NBool;
```

Module

NTypes Unit [905]

7.1.2.64.3.3 NBoolean Type

32-bit boolean value. See also NTrue [1064] and NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
typedef NInt NBoolean;
```

Module

NTypes Unit [905]

7.1.2.64.3.4 NByte Type

Same as NUInt8 [1009].

C++

```
typedef NUInt8 NByte;
```

Module

NTypes Unit [905]

7.1.2.64.3.5 NChar Type

ANSI character (8-bit). See also NAChar [1004].

C++

```
typedef NAChar NChar;
```

Module

NTypes Unit [905]

7.1.2.64.3.6 NComplex Type

Complex number.

C++

```
typedef struct NComplex_ NComplex;
```

Module

NTypes Unit [905]

7.1.2.64.3.7 NDouble Type

Double precision floating point number.

C++

```
typedef double NDouble;
```

Module

NTypes Unit [905]

7.1.2.64.3.8 NFloat Type

Same as NSingle [1008].

C++

```
typedef NSingle NFloat;
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.3.9 NHandle Type

Pointer to unspecified data (same as void *).

C++

```
typedef void * NHandle;
```

Module

NTypes Unit [905]

7.1.2.64.3.10 NIndexPair Type

Represents a pair of indexes.

C++

```
typedef struct NIndexPair_ NIndexPair;
```

Module

NTypes Unit [905]

7.1.2.64.3.11 NInt Type

Same as NInt32 [1006].

C++

```
typedef NInt32 NInt;
```

Module

NTypes Unit [905]

7.1.2.64.3.12 NInt16 Type

16-bit signed integer (short).

C++

```
typedef signed short NInt16;
```

Module

NTypes Unit [905]

7.1.2.64.3.13 NInt32 Type

32-bit signed integer (int).

C++

```
typedef signed int NInt32;
```

Module

NTypes Unit [905]

7.1.2.64.3.14 NInt64 Type

64-bit signed integer (long). Not available on some 32-bit platforms.

C++

```
typedef signed long long NInt64;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.3.15 NInt8 Type

8-bit signed integer (signed byte).

C++

```
typedef signed char NInt8;
```

Module

NTypes Unit [905]

7.1.2.64.3.16 NLong Type

Same as NInt64 [1006].

C++

```
typedef NInt64 NLong;
```

Module

NTypes Unit [905]

7.1.2.64.3.17 NNameStringPair Type

Two linked strings.

C++

```
typedef struct NNameStringPair_ NNameStringPair;
```

Module

NTypes Unit [905]

7.1.2.64.3.18 NRational Type

Represents a signed rational number.

C++

```
typedef struct NRational_ NRational;
```

Module

NTypes Unit [905]

7.1.2.64.3.19 NResult Type

Result of a function (same as NInt [1006]). See also NErrors module.

C++

```
typedef signed int NResult;
```

Remarks

Function result can be checked using NFailed [434] and NSucceeded [434] macros. When an error occurs NFailed [434](function_result) returns value which is less than zero. When function succeeds NSucceeded [434](function_result) returns value which is greater or equal to zero. So function result (NResult) can be checked using the following code:

```
/* ... */
if (NSucceeded(function_result))
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
{
   /* code which should be executed when function succeeds */
}
if (NFailed(function_result))
{
   /* code which should be executed when function fails */
}
/* ... */
```

When an error occurs function returns an error code. All error codes have negative values and can be checked in the error codes list [17].

Some functions that create memory buffer when succeed can return the size of created memory buffer (for example the number of NChar [1005] values in memory buffer). You should note that all NChar [1005] memory buffers are zero terminated so the length of these buffers are length of a string plus 1 (`result = GetValue(buffer, length + 1)`). The pattern for functions with NChar [1005] parameters is as the following:

```
NInt length = GetValue(NULL, 0);
if (NFailed(length)) { /* handle error */ }
malloc((length + 1) * sizeof(NChar))
result = GetValue(buffer, length + 1);
```

Module

NTypes Unit [905]

7.1.2.64.3.20 NSByte Type

Same as NInt8 [1007].

C++

```
typedef NInt8 NSByte;
```

Module

NTypes Unit [905]

7.1.2.64.3.21 NShort Type

Same as NInt16 [1006].

C++

```
typedef NInt16 NShort;
```

Module

NTypes Unit [905]

7.1.2.64.3.22 NSingle Type

Single precision floating point number.

C++

```
typedef float NSingle;
```

Module

NTypes Unit [905]

7.1.2.64.3.23 NSizeType Type

Platform dependent size type. Unsigned 32-bit integer on 32-bit platform, unsigned 64-bit integer on 64-bit platform.

C++

```
typedef NUInt32 NSizeType;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.3.24 NSSizeType Type

Platform dependent size type. Signed 32-bit integer on 32-bit platform, signed 64-bit integer on 64-bit platform.

C++

```
typedef NInt32 NSSizeType;
```

Module

NTypes Unit [905]

7.1.2.64.3.25 NUInt Type

Same as NUInt32 [1009].

C++

```
typedef NUInt32 NUInt;
```

Module

NTypes Unit [905]

7.1.2.64.3.26 NUInt16 Type

16-bit unsigned integer (unsigned short).

C++

```
typedef unsigned short NUInt16;
```

Module

NTypes Unit [905]

7.1.2.64.3.27 NUInt32 Type

32-bit unsigned integer (unsigned int).

C++

```
typedef unsigned int NUInt32;
```

Module

NTypes Unit [905]

7.1.2.64.3.28 NUInt64 Type

64-bit unsigned integer (unsigned long). Not available on some 32-bit platforms.

C++

```
typedef unsigned long long NUInt64;
```

Module

NTypes Unit [905]

7.1.2.64.3.29 NUInt8 Type

8-bit unsigned integer (byte).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
typedef unsigned char NUInt8;
```

Module

NTypes Unit [905]

7.1.2.64.3.30 NULong Type

Same as NUInt64 [1009].

C++

```
typedef NUInt64 NULong;
```

Module

NTypes Unit [905]

7.1.2.64.3.31 NURational Type

Represents an unsigned rational number.

C++

```
typedef struct NURational_ NURational;
```

Module

NTypes Unit [905]

7.1.2.64.3.32 NUShort Type

Same as NUInt16 [1009].

C++

```
typedef NUInt16 NUShort;
```

Module

NTypes Unit [905]

7.1.2.64.3.33 NVersion Type

Two NInt32 [1006] values defining a version.

C++

```
typedef NVersion_ NVersion;
```

Module

NTypes Unit [905]

7.1.2.64.3.34 NVersion_ Type

A NUShort [1010] typedef defining a version.

C++

```
typedef NUShort NVersion_;
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.3.35 NVersionRange Type

Four NInt32 [1006] values defining a version range.

C++

```
typedef NVersionRange_ NVersionRange;
```

Module

NTypes Unit [905]

7.1.2.64.3.36 NVersionRange_ Type

A NUInt [1009] typedef defining a version range.

C++

```
typedef NUInt NVersionRange_;
```

Module

NTypes Unit [905]

7.1.2.64.4 Macros

7.1.2.64.4.1 N_DECLARE_HANDLE Macro

Declares handle with specified name.

C++

```
#define N_DECLARE_HANDLE(name) typedef struct name##_ { int unused; } * name;
```

Module

NTypes Unit [905]

7.1.2.64.4.2 N_AW Macro

Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version in a macro (with either 'A' or 'W' suffix accordingly).

C++

```
#define N_AW A
```

Module

NTypes Unit [905]

7.1.2.64.4.3 N_BOOLEAN_FALSE_STRING Macro

ANSI or Unicode (if N_UNICODE [384] is defined) string, representing NFalse [1041] Boolean value.

C++

```
#define N_BOOLEAN_FALSE_STRING N_MACRO_AW(N_BOOLEAN_FALSE_STRING)
```

Module

NTypes Unit [905]

7.1.2.64.4.4 N_BOOLEAN_FALSE_STRINGA Macro

ANSI string, representing NFalse [1041] Boolean value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_BOOLEAN_FALSE_STRINGA "False"
```

Module

NTypes Unit [905]

7.1.2.64.4.5 N_BOOLEAN_FALSE_STRINGW Macro

Unicode (if N_UNICODE [384] is defined) string, representing NFalse [1041] Boolean value.

C++

```
#define N_BOOLEAN_FALSE_STRINGW L"False"
```

Module

NTypes Unit [905]

7.1.2.64.4.6 N_BOOLEAN_TRUE_STRING Macro

ANSI or Unicode (if N_UNICODE [384] is defined) string, representing NTrue [1064] Boolean value.

C++

```
#define N_BOOLEAN_TRUE_STRING N_MACRO_AW(N_BOOLEAN_TRUE_STRING)
```

Module

NTypes Unit [905]

7.1.2.64.4.7 N_BOOLEAN_TRUE_STRINGA Macro

ANSI string, representing NTrue [1064] Boolean value.

C++

```
#define N_BOOLEAN_TRUE_STRINGA "True"
```

Module

NTypes Unit [905]

7.1.2.64.4.8 N_BOOLEAN_TRUE_STRINGW Macro

Unicode (if N_UNICODE [384] is defined) string, representing NTrue [1064] Boolean value.

C++

```
#define N_BOOLEAN_TRUE_STRINGW L"True"
```

Module

NTypes Unit [905]

7.1.2.64.4.9 N_BYTE_MAX Macro

Maximum value for NByte [1005].

C++

```
#define N_BYTE_MAX N_UINT8_MAX
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.10 N_BYTE_MIN Macro

Minimum value for NByte [1005].

C++

```
#define N_BYTE_MIN N_UINT8_MIN
```

Module

NTypes Unit [905]

7.1.2.64.4.11 N_CALLBACK_AW Macro

Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the callback (with either 'A' or 'W' suffix accordingly).

C++

```
#define N_CALLBACK_AW(name) N_SYMBOL_AW(name)
```

Module

NTypes Unit [905]

7.1.2.64.4.12 N_DECLARE_HANDLE_TYPE Macro

Declares handle type with specified name.

C++

```
#define N_DECLARE_HANDLE_TYPE(name) \
    N_DECLARE_HANDLE(H##name)\
    N_DECLARE_TYPE(name)
```

Module

NTypes Unit [905]

7.1.2.64.4.13 N_DECLARE_OBJECT_TYPE Macro

Declares object type with specified name.

C++

```
#define N_DECLARE_OBJECT_TYPE(name, baseName) \
        typedef H##baseName H##name;\
        N_DECLARE_STATIC_OBJECT_TYPE(name)
```

Module

NTypes Unit [905]

7.1.2.64.4.14 N_DECLARE_STATIC_OBJECT_TYPE Macro

Declares static object type with specified name.

C++

```
#define N_DECLARE_STATIC_OBJECT_TYPE(name) \
    N_DECLARE_TYPE(name)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.15 N_DECLARE_TYPE Macro

Declares type with specified name.

C++

```
#define N_DECLARE_TYPE(name) \
    NResult N_API name##TypeOf(HNType * phValue);
```

Module

NTypes Unit [905]

7.1.2.64.4.16 N_DECLATE_PRIMITIVE Macro

Declares primitive with specified name.

C++

```
#define N_DECLATE_PRIMITIVE(name, size) typedef struct { union { void * ptr; /* for correct
alignment */ NByte data[size]; } unused; } name;
```

Module

NTypes Unit [905]

7.1.2.64.4.17 N_DOUBLE_EPSILON Macro

Epsilon value for NDouble [1005].

C++

```
#define N_DOUBLE_EPSILON 2.2204460492503131e-016
```

Module

NTypes Unit [905]

7.1.2.64.4.18 N_DOUBLE_MAX Macro

Maximum value for NDouble [1005].

C++

```
#define N_DOUBLE_MAX 1.7976931348623158e+308
```

Module

NTypes Unit [905]

7.1.2.64.4.19 N_DOUBLE_MIN Macro

Minimum value for NDouble [1005].

C++

```
#define N_DOUBLE_MIN -1.7976931348623158e+308
```

Module

NTypes Unit [905]

7.1.2.64.4.20 N_EMPTY_STRING Macro

ANSI or Unicode (if N_UNICODE [384] is defined) string, representing empty string value.

C++

```
#define N_EMPTY_STRING N_EMPTY_STRINGA
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.21 N_EMPTY_STRINGA Macro

ANSI string, representing empty string value.

C++

```
#define N_EMPTY_STRINGA ""
```

Module

NTypes Unit [905]

7.1.2.64.4.22 N_EMPTY_STRINGW Macro

Unicode (if N_UNICODE [384] is defined) string, representing empty string value.

C++

```
#define N_EMPTY_STRINGW L""
```

Module

NTypes Unit [905]

7.1.2.64.4.23 N_FIELD_AW Macro

Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the field (with either 'A' or 'W' suffix accordingly).

C++

```
#define N_FIELD_AW(name) N_SYMBOL_AW(name)
```

Module

NTypes Unit [905]

7.1.2.64.4.24 N_FLOAT_EPSILON Macro

Epsilon value for NFloat [1005].

C++

```
#define N_FLOAT_EPSILON N_SINGLE_EPSILON
```

Module

NTypes Unit [905]

7.1.2.64.4.25 N_FLOAT_INFINITY_STRING Macro

ANSI or Unicode (if N_UNICODE [384] is defined) string, representing "Infinity" floating-point value.

C++

```
#define N_FLOAT_INFINITY_STRING N_MACRO_AW(N_FLOAT_INFINITY_STRING)
```

Module

NTypes Unit [905]

7.1.2.64.4.26 N_FLOAT_INFINITY_STRINGA Macro

ANSI string, representing "Infinity" floating-point value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_FLOAT_INFINITY_STRINGA "Infinity"
```

Module

NTypes Unit [905]

7.1.2.64.4.27 N_FLOAT_INFINITY_STRINGW Macro

Unicode (if N_UNICODE [384] is defined) string, representing "Infinity" floating-point value.

C++

```
#define N_FLOAT_INFINITY_STRINGW L"Infinity"
```

Module

NTypes Unit [905]

7.1.2.64.4.28 N_FLOAT_MAX Macro

Maximum value for NFloat [1005].

C++

```
#define N_FLOAT_MAX N_SINGLE_MAX
```

Module

NTypes Unit [905]

7.1.2.64.4.29 N_FLOAT_MIN Macro

Minimum value for NFloat [1005].

C++

```
#define N_FLOAT_MIN N_SINGLE_MIN
```

Module

NTypes Unit [905]

7.1.2.64.4.30 N_FLOAT_NAN_STRING Macro

ANSI or Unicode (if N_UNICODE [384] is defined) string, representing "NaN" floating-point value.

C++

```
#define N_FLOAT_NAN_STRING N_MACRO_AW(N_FLOAT_NAN_STRING)
```

Module

NTypes Unit [905]

7.1.2.64.4.31 N_FLOAT_NAN_STRINGA Macro

ANSI string, representing "NaN" floating-point value.

C++

```
#define N_FLOAT_NAN_STRINGA "NaN"
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.32 N_FLOAT_NAN_STRINGW Macro

Unicode (if N_UNICODE [384] is defined) string, representing "NaN" floating-point value.

C++

```
#define N_FLOAT_NAN_STRINGW L"NaN"
```

Module

NTypes Unit [905]

7.1.2.64.4.33 N_FUNC_AW Macro

Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the function (with either 'A' or 'W' suffix accordingly).

C++

```
#define N_FUNC_AW(name) N_SYMBOL_AW(name)
```

Module

NTypes Unit [905]

7.1.2.64.4.34 N_INT_MAX Macro

Maximum value for NInt [1006].

C++

```
#define N_INT_MAX N_INT32_MAX
```

Module

NTypes Unit [905]

7.1.2.64.4.35 N_INT_MIN Macro

Minimum value for NInt [1006].

C++

```
#define N_INT_MIN N_INT32_MIN
```

Module

NTypes Unit [905]

7.1.2.64.4.36 N_INT16_MAX Macro

Maximum value for NShort [1008].

C++

```
#define N_INT16_MAX ((NInt16)0x7FFF)
```

Module

NTypes Unit [905]

7.1.2.64.4.37 N_INT16_MIN Macro

Minimum value for NShort [1008].

C++

```
#define N_INT16_MIN ((NInt16)0x8000)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.38 N_INT32_MAX Macro

Maximum value for NInt [1006].

C++

```
#define N_INT32_MAX 0x7FFFFFFF
```

Module

NTypes Unit [905]

7.1.2.64.4.39 N_INT32_MIN Macro

Minimum value for NInt [1006].

C++

```
#define N_INT32_MIN ((NInt32)0x80000000)
```

Module

NTypes Unit [905]

7.1.2.64.4.40 N_INT64_MAX Macro

Maximum value for NInt64 [1006].

C++

```
#define N_INT64_MAX 0x7FFFFFFFFFFFFFFFll
```

Module

NTypes Unit [905]

7.1.2.64.4.41 N_INT64_MIN Macro

Minimum value for NInt64 [1006].

C++

```
#define N_INT64_MIN ((NInt64)0x8000000000000000ull)
```

Module

NTypes Unit [905]

7.1.2.64.4.42 N_INT8_MAX Macro

Maximum value for NSByte [1008].

C++

```
#define N_INT8_MAX ((NInt8)0x7F)
```

Module

NTypes Unit [905]

7.1.2.64.4.43 N_INT8_MIN Macro

Minimum value for NSByte [1008].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_INT8_MIN ((NInt8)0x80)
```

Module

NTypes Unit [905]

7.1.2.64.4.44 N_JOIN_SYMBOLS Macro

Concatenates two symbols, yielding a single concatenated symbol.

C++

```
#define N_JOIN_SYMBOLS(name1, name2) N_JOIN_SYMBOLS_(name1, name2)
```

Module

NTypes Unit [905]

7.1.2.64.4.45 N_JOIN_SYMBOLS_ Macro

Concatenates two symbols, yielding a single concatenated symbol. Can serve as a macro argument.

C++

```
#define N_JOIN_SYMBOLS_(name1, name2) name1##name2
```

Module

NTypes Unit [905]

7.1.2.64.4.46 N_JOIN_SYMBOLS2 Macro

Concatenates two symbols, yielding a single concatenated symbol.

C++

```
#define N_JOIN_SYMBOLS2(name1, name2) N_JOIN_SYMBOLS2_(name1, name2)
```

Module

NTypes Unit [905]

7.1.2.64.4.47 N_JOIN_SYMBOLS2_ Macro

Concatenates two symbols, yielding a single concatenated symbol. Can serve as a macro argument.

C++

```
#define N_JOIN_SYMBOLS2_(name1, name2) name1##name2
```

Module

NTypes Unit [905]

7.1.2.64.4.48 N_LONG_MAX Macro

Maximum value for NLong [1007].

C++

```
#define N_LONG_MAX N_INT64_MAX
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.49 N_LONG_MIN Macro

Minimum value for NLong [1007].

C++

```
#define N_LONG_MIN N_INT64_MIN
```

Module

NTypes Unit [905]

7.1.2.64.4.50 N_MACRO_AW Macro

Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the macro (with either 'A' or 'W' suffix accordingly).

C++

```
#define N_MACRO_AW(name) N_SYMBOL_AW(name)
```

Module

NTypes Unit [905]

7.1.2.64.4.51 N_PTR_SIZE Macro

Defines pointer size.

C++

```
#define N_PTR_SIZE sizeof(void *)
```

Module

NTypes Unit [905]

7.1.2.64.4.52 N_SBYTE_MAX Macro

Maximum value for NSByte [1008].

C++

```
#define N_SBYTE_MAX N_INT8_MAX
```

Module

NTypes Unit [905]

7.1.2.64.4.53 N_SBYTE_MIN Macro

Minimum value for NSByte [1008].

C++

```
#define N_SBYTE_MIN N_INT8_MIN
```

Module

NTypes Unit [905]

7.1.2.64.4.54 N_SHORT_MAX Macro

Maximum value for NShort [1008].

C++

```
#define N_SHORT_MAX N_INT16_MAX
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.55 N_SHORT_MIN Macro

Minimum value for NShort [1008].

C++

```
#define N_SHORT_MIN N_INT16_MIN
```

Module

NTypes Unit [905]

7.1.2.64.4.56 N_SINGLE_EPSILON Macro

Epsilon value for NSingle [1008].

C++

```
#define N_SINGLE_EPSILON 1.192092896e-07F
```

Module

NTypes Unit [905]

7.1.2.64.4.57 N_SINGLE_MAX Macro

Maximum value for NSingle [1008].

C++

```
#define N_SINGLE_MAX 3.402823466e+38F
```

Module

NTypes Unit [905]

7.1.2.64.4.58 N_SINGLE_MIN Macro

Minimum value for NSingle [1008].

C++

```
#define N_SINGLE_MIN -3.402823466e+38F
```

Module

NTypes Unit [905]

7.1.2.64.4.59 N_SIZE_TYPE_MAX Macro

Maximum value for NSizeType [1008].

C++

```
#define N_SIZE_TYPE_MAX N_UINT32_MAX
```

Module

NTypes Unit [905]

7.1.2.64.4.60 N_SIZE_TYPE_MIN Macro

Minimum value for NSizeType [1008].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_SIZE_TYPE_MIN N_UINT32_MIN
```

Module

NTypes Unit [905]

7.1.2.64.4.61 N_SSIZE_TYPE_MAX Macro

Maximum value for NSSizeType [1009].

C++

```
#define N_SSIZE_TYPE_MAX N_INT32_MAX
```

Module

NTypes Unit [905]

7.1.2.64.4.62 N_SSIZE_TYPE_MIN Macro

Minimum value for NSSizeType [1009].

C++

```
#define N_SSIZE_TYPE_MIN N_INT32_MIN
```

Module

NTypes Unit [905]

7.1.2.64.4.63 N_STRINGIZE Macro

Defines the size of ANSI or Unicode (if N_UNICODE [384] is defined) string.

C++

```
#define N_STRINGIZE N_STRINGIZEA
```

Module

NTypes Unit [905]

7.1.2.64.4.64 N_STRINGIZE_ Macro

Defines the size of ANSI or Unicode (if N_UNICODE [384] is defined) string. Can serve as a macro argument.

C++

```
#define N_STRINGIZE_ N_STRINGIZE_A
```

Module

NTypes Unit [905]

7.1.2.64.4.65 N_STRINGIZE_A Macro

Defines the size of ANSI string. Can serve as a macro argument.

C++

```
#define N_STRINGIZE_A(value) N_TA(#value)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.66 N_STRINGIZE_W Macro

Defines the size of Unicode (if N_UNICODE [384] is defined) string. Can serve as a macro argument.

C++

```
#define N_STRINGIZE_W(value) N_TW(#value)
```

Module

NTypes Unit [905]

7.1.2.64.4.67 N_STRINGIZEA Macro

Defines the size of ANSI string.

C++

```
#define N_STRINGIZEA(value) N_STRINGIZE_A(value)
```

Module

NTypes Unit [905]

7.1.2.64.4.68 N_STRINGIZEW Macro

Defines the size of Unicode (if N_UNICODE [384] is defined) string.

C++

```
#define N_STRINGIZEW(value) N_STRINGIZE_W(value)
```

Module

NTypes Unit [905]

7.1.2.64.4.69 N_STRUCT_AW Macro

Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the struct (with either 'A' or 'W' suffix accordingly).

C++

```
#define N_STRUCT_AW(name) N_SYMBOL_AW(name)
```

Module

NTypes Unit [905]

7.1.2.64.4.70 N_SYMBOL_AW Macro

Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the symbol (with either 'A' or 'W' suffix accordingly).

C++

```
#define N_SYMBOL_AW(name) N_JOIN_SYMBOLS2(name, N_AW)
```

Module

NTypes Unit [905]

7.1.2.64.4.71 N_T Macro

Makes either ANSI or Unicode (if N_UNICODE [384] is defined) string or character constant.

C++

```
#define N_T N_TA
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.72 N_T_A Macro

Makes ANSI string or character constant. Can serve as a macro argument.

C++

```
#define N_T_A(text) text
```

Module

NTypes Unit [905]

7.1.2.64.4.73 N_T_W Macro

Makes Unicode (if N_UNICODE [384] is defined) string or character constant. Can serve as a macro argument.

C++

```
#define N_T_W(text) L##text
```

Module

NTypes Unit [905]

7.1.2.64.4.74 N_TA Macro

Makes ANSI string or character constant.

C++

```
#define N_TA(text) N_T_A(text)
```

Module

NTypes Unit [905]

7.1.2.64.4.75 N_TW Macro

Makes Unicode (if N_UNICODE [384] is defined) string or character constant.

C++

```
#define N_TW(text) N_T_W(text)
```

Module

NTypes Unit [905]

7.1.2.64.4.76 N_TYPE_AW Macro

Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the type (with either 'A' or 'W' suffix accordingly).

C++

```
#define N_TYPE_AW(name) N_SYMBOL_AW(name)
```

Module

NTypes Unit [905]

7.1.2.64.4.77 N_TYPE_OF Macro

Used to obtain a pointer to the function that retrieves NType [871] object for the specified type name.",

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_TYPE_OF(name) N_JOIN_SYMBOLS(name, TypeOf)
```

Module

NTypes Unit [905]

7.1.2.64.4.78 N_UINT_MAX Macro

Maximum value for NUInt [1009].

C++

```
#define N_UINT_MAX N_UINT32_MAX
```

Module

NTypes Unit [905]

7.1.2.64.4.79 N_UINT_MIN Macro

Minimum value for NUInt [1009].

C++

```
#define N_UINT_MIN N_UINT32_MIN
```

Module

NTypes Unit [905]

7.1.2.64.4.80 N_UINT16_MAX Macro

Maximum value for NUShort [1010].

C++

```
#define N_UINT16_MAX ((NUInt16)0xFFFFu)
```

Module

NTypes Unit [905]

7.1.2.64.4.81 N_UINT16_MIN Macro

Minimum value for NUShort [1010].

C++

```
#define N_UINT16_MIN ((NUInt16)0x0000u)
```

Module

NTypes Unit [905]

7.1.2.64.4.82 N_UINT32_MAX Macro

Maximum value for NUInt [1009].

C++

```
#define N_UINT32_MAX 0xFFFFFFFFu
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.83 N_UINT32_MIN Macro

Minimum value for NUInt [1009].

C++

```
#define N_UINT32_MIN 0x00000000u
```

Module

NTypes Unit [905]

7.1.2.64.4.84 N_UINT64_MAX Macro

Maximum value for NULong [1010].

C++

```
#define N_UINT64_MAX 0xFFFFFFFFFFFFFFFFull
```

Module

NTypes Unit [905]

7.1.2.64.4.85 N_UINT64_MIN Macro

Minimum value for NULong [1010].

C++

```
#define N_UINT64_MIN 0x0000000000000000ull
```

Module

NTypes Unit [905]

7.1.2.64.4.86 N_UINT8_MAX Macro

Maximum value for NUInt8 [1009].

C++

```
#define N_UINT8_MAX ((NUInt8)0xFFu)
```

Module

NTypes Unit [905]

7.1.2.64.4.87 N_UINT8_MIN Macro

Minimum value for NUInt8 [1009].

C++

```
#define N_UINT8_MIN ((NUInt8)0x00u)
```

Module

NTypes Unit [905]

7.1.2.64.4.88 N_ULONG_MAX Macro

Maximum value for NULong [1010].

C++

```
#define N_ULONG_MAX N_UINT64_MAX
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.89 N_ULONG_MIN Macro

Minimum value for NULong [1010].

C++

```
#define N_ULONG_MIN N_UINT64_MIN
```

Module

NTypes Unit [905]

7.1.2.64.4.90 N_UNREFERENCED_PARAMETER Macro

Suppresses the warning "unreferenced parameter".

C++

```
#define N_UNREFERENCED_PARAMETER(parameter) (void)(parameter)
```

Module

NTypes Unit [905]

7.1.2.64.4.91 N_UNUSED_VARIABLE Macro

Suppresses the warning "unused variable".

C++

```
#define N_UNUSED_VARIABLE(variable) (void)(variable)
```

Module

NTypes Unit [905]

7.1.2.64.4.92 N_USHORT_MAX Macro

Maximum value for NUShort [1010].

C++

```
#define N_USHORT_MAX N_UINT16_MAX
```

Module

NTypes Unit [905]

7.1.2.64.4.93 N_USHORT_MIN Macro

Minimum value for NUShort [1010].

C++

```
#define N_USHORT_MIN N_UINT16_MIN
```

Module

NTypes Unit [905]

7.1.2.64.4.94 N_VAR_AW Macro

Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the variable (with either 'A' or 'W' suffix accordingly).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_VAR_AW(name) N_SYMBOL_AW(name)
```

Module

NTypes Unit [905]

7.1.2.64.4.95 N_WCHAR_SIZE Macro

Defines the size of Unicode char.

C++

```
#define N_WCHAR_SIZE 4
```

Module

NTypes Unit [905]

7.1.2.64.4.96 NACharParse Macro

Converts a NString [678] to NAChar [1004].

C++

```
#define NACharParse(szValue, szFormat, pValue) NACharParseStrOrChars(szValue, -1, szFormat,
pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NAChar [1004].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.97 NACharParseChars Macro

Converts an array of NChars to NAChar [1004].

C++

```
#define NACharParseChars(arValue, valueLength, szFormat, pValue)
NACharParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NAChar [1004].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.98 NACharParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NACharParseStrOrChars function.

C++

```
#define NACharParseStrOrChars N_FUNC_AW(NACharParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.99 NACharParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NACharParseVN function.

C++

```
#define NACharParseVN N_FUNC_AW(NACharParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.100 NACharToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NACharToChars function.

C++

```
#define NACharToChars N_FUNC_AW(NACharToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.101 NACharToString Macro

Picks a Unicode (W) or ASCII (A) version of the NACharToString function.

C++

```
#define NACharToString N_FUNC_AW(NACharToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.102 NACharTryParse Macro

Converts NString [678] to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NACharTryParse(szValue, szFormat, pValue, pResult)
NACharTryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
hValue	[in] A handle to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NAChar [1004].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.103 NACharTryParseChars Macro

Converts an array of NChars to NChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NACharTryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NACharTryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NChar [1004].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.104 NACharTryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NACharTryParseStrOrChars function.

C++

```
#define NACharTryParseStrOrChars N_FUNC_AW(NACharTryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.105 NACharTryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NACharTryParseVN function.

C++

```
#define NACharTryParseVN N_FUNC_AW(NACharTryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.106 NBooleanParse Macro

Converts NString [678] to NBoolean [1004].

C++

```
#define NBooleanParse(szValue, szFormat, pValue) NBooleanParseStrOrChars(szValue, -1,
szFormat, pValue)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NBoolean [1004].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.107 NBooleanParseChars Macro

Converts an array of NChars to NBoolean [1004].

C++

```
#define NBooleanParseChars(arValue, szFormat, szFalseString, pValue)
NBooleanParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
szFormat	[in] Pointer to numeric format string.
szFalseString	[in] String for NFalse [1041].
pValue	[out] A pointer to the converted NBoolean [1004].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.108 NBooleanParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NBooleanParseStrOrChars function.

C++

```
#define NBooleanParseStrOrChars N_FUNC_AW(NBooleanParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.109 NBooleanParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NBooleanParseVN function.

C++

```
#define NBooleanParseVN N_FUNC_AW(NBooleanParseVN)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.110 NBooleanToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NBooleanToChars function.

C++

```
#define NBooleanToChars N_FUNC_AW(NBooleanToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.111 NBooleanToString Macro

Picks a Unicode (W) or ASCII (A) version of the NBooleanToString function.

C++

```
#define NBooleanToString N_FUNC_AW(NBooleanToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.112 NBooleanTryParse Macro

Converts NString [678] to NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NBooleanTryParse(szValue, szFormat, pValue, pResult)
NBooleanTryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
hValue	[in] A handle to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NBoolean [1004].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.113 NBooleanTryParseChars Macro

Converts an array of NChars to NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NBooleanTryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NBooleanTryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NAChar [1004].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.114 NBooleanTryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NBooleanTryParseStrOrChars function.

C++

```
#define NBooleanTryParseStrOrChars N_FUNC_AW(NBooleanTryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.115 NBooleanTryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NBooleanTryParseVN function.

C++

```
#define NBooleanTryParseVN N_FUNC_AW(NBooleanTryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.116 NCharFromBinDigit Macro

Picks a Unicode (W) or ASCII (A) version of the NCharFromBinDigit function.

C++

```
#define NCharFromBinDigit N_FUNC_AW(NCharFromBinDigit)
```

Module

NTypes Unit [905]

7.1.2.64.4.117 NCharFromDigit Macro

Picks a Unicode (W) or ASCII (A) version of the NCharFromDigit function.

C++

```
#define NCharFromDigit N_FUNC_AW(NCharFromDigit)
```

Module

NTypes Unit [905]

7.1.2.64.4.118 NCharFromHexDigit Macro

Picks a Unicode (W) or ASCII (A) version of the NCharFromHexDigit function.

C++

```
#define NCharFromHexDigit N_FUNC_AW(NCharFromHexDigit)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.119 NCharFromOctDigit Macro

Picks a Unicode (W) or ASCII (A) version of the NCharFromOctDigit function.

C++

```
#define NCharFromOctDigit N_FUNC_AW(NCharFromOctDigit)
```

Module

NTypes Unit [905]

7.1.2.64.4.120 NCharIsAscii Macro

Picks a Unicode (W) or ASCII (A) version of the NCharIsAscii function.

C++

```
#define NCharIsAscii N_FUNC_AW(NCharIsAscii)
```

Module

NTypes Unit [905]

7.1.2.64.4.121 NCharIsBinDigit Macro

Picks a Unicode (W) or ASCII (A) version of the NCharIsBinDigit function.

C++

```
#define NCharIsBinDigit N_FUNC_AW(NCharIsBinDigit)
```

Module

NTypes Unit [905]

7.1.2.64.4.122 NCharIsDigit Macro

Picks a Unicode (W) or ASCII (A) version of the NCharIsDigit function.

C++

```
#define NCharIsDigit N_FUNC_AW(NCharIsDigit)
```

Module

NTypes Unit [905]

7.1.2.64.4.123 NCharIsHexDigit Macro

Picks a Unicode (W) or ASCII (A) version of the NCharIsHexDigit function.

C++

```
#define NCharIsHexDigit N_FUNC_AW(NCharIsHexDigit)
```

Module

NTypes Unit [905]

7.1.2.64.4.124 NCharIsLetter Macro

Picks a Unicode (W) or ASCII (A) version of the NCharIsLetter function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NCharIsLetter N_FUNC_AW(NCharIsLetter)
```

Module

NTypes Unit [905]

7.1.2.64.4.125 NCharIsLower Macro

Picks a Unicode (W) or ASCII (A) version of the NCharIsLower function.

C++

```
#define NCharIsLower N_FUNC_AW(NCharIsLower)
```

Module

NTypes Unit [905]

7.1.2.64.4.126 NCharIsOctDigit Macro

Picks a Unicode (W) or ASCII (A) version of the NCharIsOctDigit function.

C++

```
#define NCharIsOctDigit N_FUNC_AW(NCharIsOctDigit)
```

Module

NTypes Unit [905]

7.1.2.64.4.127 NCharIsUpper Macro

Picks a Unicode (W) or ASCII (A) version of the NCharIsUpper function.

C++

```
#define NCharIsUpper N_FUNC_AW(NCharIsUpper)
```

Module

NTypes Unit [905]

7.1.2.64.4.128 NCharIsWhiteSpace Macro

Picks a Unicode (W) or ASCII (A) version of the NCharIsWhiteSpace function.

C++

```
#define NCharIsWhiteSpace N_FUNC_AW(NCharIsWhiteSpace)
```

Module

NTypes Unit [905]

7.1.2.64.4.129 NCharParse Macro

Converts NString [678] to NChar [1005].

C++

```
#define NCharParse(szValue, szFormat, pValue) NCharParseStrOrChars(szValue, -1, szFormat,
pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NChar [1005].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.130 NCharParseChars Macro

Converts an array of NChars to NBoolean [1004].

C++

```
#define NCharParseChars(arValue, valueLength, szFormat, pValue)
NCharParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NChar [1005].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.131 NCharToBinDigit Macro

Picks a Unicode (W) or ASCII (A) version of the NCharToBinDigit function.

C++

```
#define NCharToBinDigit N_FUNC_AW(NCharToBinDigit)
```

Module

NTypes Unit [905]

7.1.2.64.4.132 NCharToDigit Macro

Picks a Unicode (W) or ASCII (A) version of the NCharToDigit function.

C++

```
#define NCharToDigit N_FUNC_AW(NCharToDigit)
```

Module

NTypes Unit [905]

7.1.2.64.4.133 NCharToHexDigit Macro

Picks a Unicode (W) or ASCII (A) version of the NCharToHexDigit function.

C++

```
#define NCharToHexDigit N_FUNC_AW(NCharToHexDigit)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.134 NCharToLower Macro

Picks a Unicode (W) or ASCII (A) version of the NCharToLower function.

C++

```
#define NCharToLower N_FUNC_AW(NCharToLower)
```

Module

NTypes Unit [905]

7.1.2.64.4.135 NCharToOctDigit Macro

Picks a Unicode (W) or ASCII (A) version of the NCharToOctDigit function.

C++

```
#define NCharToOctDigit N_FUNC_AW(NCharToOctDigit)
```

Module

NTypes Unit [905]

7.1.2.64.4.136 NCharToUpper Macro

Picks a Unicode (W) or ASCII (A) version of the NCharToUpper function.

C++

```
#define NCharToUpper N_FUNC_AW(NCharToUpper)
```

Module

NTypes Unit [905]

7.1.2.64.4.137 NCharTryParse Macro

Converts NString [678] to NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NCharTryParse(szValue, szFormat, pValue, pResult) NCharTryParseStrOrChars(szValue,
-1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NChar [1005].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.138 NCharTryParseChars Macro

Converts an array of NChars to NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NCharTryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NCharTryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NChar [1005].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.139 NDoubleParse Macro

Converts NString [678] to NDouble [1005].

C++

```
#define NDoubleParse(szValue, szFormat, pValue) NDoubleParseStrOrChars(szValue, -1,
szFormat, pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NDouble [1005].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.140 NDoubleParseChars Macro

Converts an array of NChars to NChar [1005].

C++

```
#define NDoubleParseChars(arValue, valueLength, szFormat, pValue)
NDoubleParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NDouble [1005].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.141 NDoubleParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NDoubleParseStrOrChars function.

C++

```
#define NDoubleParseStrOrChars N_FUNC_AW(NDoubleParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.142 NDoubleParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NDoubleParseVN function.

C++

```
#define NDoubleParseVN N_FUNC_AW(NDoubleParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.143 NDoubleToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NDoubleToChars function.

C++

```
#define NDoubleToChars N_FUNC_AW(NDoubleToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.144 NDoubleToString Macro

Picks a Unicode (W) or ASCII (A) version of the NDoubleToString function.

C++

```
#define NDoubleToString N_FUNC_AW(NDoubleToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.145 NDoubleTryParse Macro

Converts NString [678] to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NDoubleTryParse(szValue, szFormat, pValue, pResult)
NDoubleTryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NDouble [1005].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.146 NDoubleTryParseChars Macro

Converts an array of NChars to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NDoubleTryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NDoubleTryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NDouble [1005].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.147 NDoubleTryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NDoubleTryParseStrOrChars function.

C++

```
#define NDoubleTryParseStrOrChars N_FUNC_AW(NDoubleTryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.148 NDoubleTryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NDoubleTryParseVN function.

C++

```
#define NDoubleTryParseVN N_FUNC_AW(NDoubleTryParseVN)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.149 NFalse Macro

False value for NBoolean [1004].

C++

```
#define NFalse 0
```

Module

NTypes Unit [905]

7.1.2.64.4.150 NGuidParse Macro

Picks a Unicode (W) or ASCII (A) version of the NGuidParse function.

C++

```
#define NGuidParse N_FUNC_AW(NGuidParse)
```

Module

NTypes Unit [905]

7.1.2.64.4.151 NGuidParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NGuidParseVN function.

C++

```
#define NGuidParseVN N_FUNC_AW(NGuidParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.152 NGuidToString Macro

Picks a Unicode (W) or ASCII (A) version of the NGuidToString function.

C++

```
#define NGuidToString N_FUNC_AW(NGuidToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.153 NGuidTryParse Macro

Picks a Unicode (W) or ASCII (A) version of the NGuidTryParse function.

C++

```
#define NGuidTryParse N_FUNC_AW(NGuidTryParse)
```

Module

NTypes Unit [905]

7.1.2.64.4.154 NGuidTryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NGuidTryParseVN function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NGuidTryParseVN N_FUNC_AW(NGuidTryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.155 NIndexPairToString Macro

Picks a Unicode (W) or ASCII (A) version of the NIndexPairToString function.

C++

```
#define NIndexPairToString N_FUNC_AW(NIndexPairToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.156 NInt16Parse Macro

Converts NString [678] to NInt16 [1006].

C++

```
#define NInt16Parse(szValue, szFormat, pValue) NInt16ParseStrOrChars(szValue, -1, szFormat,
pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt16 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.157 NInt16ParseChars Macro

Converts an array of NChars to NInt16 [1006].

C++

```
#define NInt16ParseChars(arValue, valueLength, szFormat, pValue)
NInt16ParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt16 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.158 NInt16ParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NInt16ParseStrOrChars function.

C++

```
#define NInt16ParseStrOrChars N_FUNC_AW(NInt16ParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.159 NInt16ParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NInt16ParseVN function.

C++

```
#define NInt16ParseVN N_FUNC_AW(NInt16ParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.160 NInt16ToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NInt16ToChars function.

C++

```
#define NInt16ToChars N_FUNC_AW(NInt16ToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.161 NInt16ToString Macro

Picks a Unicode (W) or ASCII (A) version of the NInt16ToString function.

C++

```
#define NInt16ToString N_FUNC_AW(NInt16ToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.162 NInt16TryParse Macro

Converts NString [678] to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NInt16TryParse(szValue, szFormat, pValue, pResult)
NInt16TryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt16 [1006].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.163 NInt16TryParseChars Macro

Converts an array of NChars to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NInt16TryParseChars(arValue, valueLength, style, pValue, pResult)
NInt16TryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt16 [1006].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.164 NInt16TryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NInt16TryParseStrOrChars function.

C++

```
#define NInt16TryParseStrOrChars N_FUNC_AW(NInt16TryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.165 NInt16TryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NInt16TryParseVN function.

C++

```
#define NInt16TryParseVN N_FUNC_AW(NInt16TryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.166 NInt32Parse Macro

Converts NString [678] to NInt32 [1006].

C++

```
#define NInt32Parse(szValue, szFormat, pValue) NInt32ParseStrOrChars(szValue, -1, szFormat,
pValue)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt32 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.167 NInt32ParseChars Macro

Converts an array of NChars to NInt32 [1006].

C++

```
#define NInt32ParseChars(arValue, valueLength, szFormat, pValue)
NInt32ParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt32 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.168 NInt32ParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NInt32ParseStrOrChars function.

C++

```
#define NInt32ParseStrOrChars N_FUNC_AW(NInt32ParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.169 NInt32ParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NInt32ParseVN function.

C++

```
#define NInt32ParseVN N_FUNC_AW(NInt32ParseVN)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.170 NInt32ToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NInt32ToChars function.

C++

```
#define NInt32ToChars N_FUNC_AW(NInt32ToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.171 NInt32ToString Macro

Picks a Unicode (W) or ASCII (A) version of the NInt32ToString function.

C++

```
#define NInt32ToString N_FUNC_AW(NInt32ToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.172 NInt32TryParse Macro

Converts NString [678] to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NInt32TryParse(szValue, szFormat, pValue, pResult)
NInt32TryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt32 [1006].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.173 NInt32TryParseChars Macro

Converts an array of NChars to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NInt32TryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NInt32TryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt32 [1006].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.174 NInt32TryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NInt32TryParseStrOrChars function.

C++

```
#define NInt32TryParseStrOrChars N_FUNC_AW(NInt32TryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.175 NInt32TryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NInt32tryParseVN function.

C++

```
#define NInt32TryParseVN N_FUNC_AW(NInt32TryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.176 NInt64Parse Macro

Converts NString [678] to NInt64 [1006].

C++

```
#define NInt64Parse(szValue, szFormat, pValue) NInt64ParseStrOrChars(szValue, -1, szFormat,
pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt64 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.177 NInt64ParseChars Macro

Converts an array of NChars to NInt64 [1006].

C++

```
#define NInt64ParseChars(arValue, valueLength, szFormat, pValue)
NInt64ParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt64 [1006].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.178 NInt64ParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NInt64ParseStrOrChars function.

C++

```
#define NInt64ParseStrOrChars N_FUNC_AW(NInt64ParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.179 NInt64ParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NInt64ParseVN function.

C++

```
#define NInt64ParseVN N_FUNC_AW(NInt64ParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.180 NInt64ToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NInt64ToChars function.

C++

```
#define NInt64ToChars N_FUNC_AW(NInt64ToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.181 NInt64ToString Macro

Picks a Unicode (W) or ASCII (A) version of the NInt64ToString function.

C++

```
#define NInt64ToString N_FUNC_AW(NInt64ToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.182 NInt64TryParse Macro

Converts NString [678] to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NInt64TryParse(szValue, szFormat, pValue, pResult)
NInt64TryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt64 [1006].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.183 NInt64TryParseChars Macro

Converts an array of NChars to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NInt64TryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NInt64TryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric string format.
pValue	[out] A pointer to the converted NInt64 [1006].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.184 NInt64TryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NInt64TryParseStrOrChars function.

C++

```
#define NInt64TryParseStrOrChars N_FUNC_AW(NInt64TryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.185 NInt64TryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NInt64TryParseVN function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NInt64TryParseVN N_FUNC_AW(NInt64TryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.186 NInt8Parse Macro

Converts NString [678] to NInt8 [1007].

C++

```
#define NInt8Parse(szValue, szFormat, pValue) NInt8ParseStrOrChars(szValue, -1, szFormat,
pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt8 [1007].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.187 NInt8ParseChars Macro

Converts an array of NChars to NInt8 [1007].

C++

```
#define NInt8ParseChars(arValue, valueLength, szFormat, pValue)
NInt8ParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt8 [1007].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.188 NInt8ParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NInt8ParseStrOrChars function.

C++

```
#define NInt8ParseStrOrChars N_FUNC_AW(NInt8ParseStrOrChars)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.189 NInt8ParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NInt8ParseVN function.

C++

```
#define NInt8ParseVN N_FUNC_AW(NInt8ParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.190 NInt8ToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NInt8ToChars function.

C++

```
#define NInt8ToChars N_FUNC_AW(NInt8ToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.191 NInt8ToString Macro

Picks a Unicode (W) or ASCII (A) version of the NInt8ToString function.

C++

```
#define NInt8ToString N_FUNC_AW(NInt8ToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.192 NInt8TryParse Macro

Converts NString [678] to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NInt8TryParse(szValue, szFormat, pValue, pResult) NInt8TryParseStrOrChars(szValue,
-1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt8 [1007].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.193 NInt8TryParseChars Macro

Converts an array of NChars to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NInt8TryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NInt8TryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NInt8 [1007].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.194 NInt8TryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NInt8TryParseStrOrChars function.

C++

```
#define NInt8TryParseStrOrChars N_FUNC_AW(NInt8TryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.195 NInt8TryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NInt8TryParseVN function.

C++

```
#define NInt8TryParseVN N_FUNC_AW(NInt8TryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.196 NPointerParse Macro

Converts NString [678] to NPointer.

C++

```
#define NPointerParse(szValue, szFormat, pValue) NPointerParseStrOrChars(szValue, -1,
szFormat, pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NPointer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.197 NPointerParseChars Macro

Converts an array of NChars to NPointer.

C++

```
#define NPointerParseChars(arValue, szFormat, szFalseString, pValue)
NPointerParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NPointer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.198 NPointerParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NPointerParseStrOrChars function.

C++

```
#define NPointerParseStrOrChars N_FUNC_AW(NPointerParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.199 NPointerParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NPointerParseVN function.

C++

```
#define NPointerParseVN N_FUNC_AW(NPointerParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.200 NPointerToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NPointerToChars function.

C++

```
#define NPointerToChars N_FUNC_AW(NPointerToChars)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.201 NPointerToString Macro

Picks a Unicode (W) or ASCII (A) version of the NPointerToString function.

C++

```
#define NPointerToString N_FUNC_AW(NPointerToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.202 NPointerTryParse Macro

Converts NString [678] to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NPointerTryParse(szValue, szFormat, pValue, pResult)
NPointerTryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NPointer.
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.203 NPointerTryParseChars Macro

Converts an array of NChars to NPointer. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NPointerTryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NPointerTryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NPointer.
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.204 NPointerTryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NPointerTryParseStrOrChars function.

C++

```
#define NPointerTryParseStrOrChars N_FUNC_AW(NPointerTryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.205 NPointerTryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NPointerTryParseVN function.

C++

```
#define NPointerTryParseVN N_FUNC_AW(NPointerTryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.206 NRangeToString Macro

Picks a Unicode (W) or ASCII (A) version of the NRangeToString function.

C++

```
#define NRangeToString N_FUNC_AW(NRangeToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.207 NRationalToString Macro

Picks a Unicode (W) or ASCII (A) version of the NRationalToString function.

C++

```
#define NRationalToString N_FUNC_AW(NRationalToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.208 NResultParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NResultParseVN function.

C++

```
#define NResultParseVN N_FUNC_AW(NResultParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.209 NResultToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NResultToChars function.

C++

```
#define NResultToChars N_FUNC_AW(NResultToChars)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.210 NResultToString Macro

Picks a Unicode (W) or ASCII (A) version of the ResultToString function.

C++

```
#define NResultToString N_FUNC_AW(NResultToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.211 NResultTryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NResultTryParseStrOrChars function.

C++

```
#define NResultTryParseStrOrChars N_FUNC_AW(NResultTryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.212 NResultTryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NResultTryParseVN function.

C++

```
#define NResultTryParseVN N_FUNC_AW(NResultTryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.213 NSingleParse Macro

Converts NString [678] to NSingle [1008].

C++

```
#define NSingleParse(szValue, szFormat, pValue) NSingleParseStrOrChars(szValue, -1,
szFormat, pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NSingle [1008].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.214 NSingleParseChars Macro

Converts an array of NChars to NSingle [1008].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NSingleParseChars(arValue, valueLength, szFormat, pValue)
NSingleParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NSingle [1008].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.215 NSingleParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NSingleParseStrOrChars function.

C++

```
#define NSingleParseStrOrChars N_FUNC_AW(NSingleParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.216 NSingleParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NSingleParseVN function.

C++

```
#define NSingleParseVN N_FUNC_AW(NSingleParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.217 NSingleToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NSingleToChars function.

C++

```
#define NSingleToChars N_FUNC_AW(NSingleToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.218 NSingleToString Macro

Picks a Unicode (W) or ASCII (A) version of the NSingleToString function.

C++

```
#define NSingleToString N_FUNC_AW(NSingleToString)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.219 NSingleTryParse Macro

Converts NString [678] to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NSingleTryParse(szValue, szFormat, pValue, pResult)
NSingleTryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NSingle [1008].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.220 NSingleTryParseChars Macro

Converts an array of NChars to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NSingleTryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NSingleTryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NSingle [1008].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.221 NSingleTryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NSingleTryParseStrOrChars function.

C++

```
#define NSingleTryParseStrOrChars N_FUNC_AW(NSingleTryParseStrOrChars)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.222 NSingleTryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NSingleTryParseVN function.

C++

```
#define NSingleTryParseVN N_FUNC_AW(NSingleTryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.223 NSizeTypeParse Macro

Converts NString [678] to NSizeType [1008].

C++

```
#define NSizeTypeParse(szValue, szFormat, pValue) NSizeTypeParseStrOrChars(szValue, -1,
szFormat, pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NSizeType [1008].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.224 NSizeTypeParseChars Macro

Converts an array of NChars to NSizeType [1008].

C++

```
#define NSizeTypeParseChars(arValue, valueLength, szFormat, pValue)
NSizeTypeParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NSizeType [1008].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.225 NSizeTypeParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NSizeTypeParseStrOrChars function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NSizeTypeParseStrOrChars N_FUNC_AW(NSizeTypeParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.226 NSizeTypeParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NSizeTypeParseVN function.

C++

```
#define NSizeTypeParseVN N_FUNC_AW(NSizeTypeParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.227 NSizeTypeToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NSizeTypeToChars function.

C++

```
#define NSizeTypeToChars N_FUNC_AW(NSizeTypeToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.228 NSizeTypeToString Macro

Picks a Unicode (W) or ASCII (A) version of the NSizeTypeToString function.

C++

```
#define NSizeTypeToString N_FUNC_AW(NSizeTypeToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.229 NSizeTypeTryParse Macro

Converts NString [678] to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NSizeTypeTryParse(szValue, szFormat, pValue, pResult)
NSizeTypeTryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NSizeType [1008].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.230 NSizeTypeTryParseChars Macro

Converts an array of NChars to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NSizeTypeTryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NSizeTypeTryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NSizeType [1008].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.231 NSizeTypeTryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NSizeTypeTryParseStrOrChars function.

C++

```
#define NSizeTypeTryParseStrOrChars N_FUNC_AW(NSizeTypeTryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.232 NSizeTypeTryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NSizeTypeTryParseVN function.

C++

```
#define NSizeTypeTryParseVN N_FUNC_AW(NSizeTypeTryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.233 NSSizeTypeParse Macro

Converts NString [678] to NSSizeType [1009].

C++

```
#define NSSizeTypeParse(szValue, szFormat, pValue) NSSizeTypeParseStrOrChars(szValue, -1,
style, pValue)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NSSizeType [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.234 NSSizeTypeParseChars Macro

Converts an array of NChars to NSSizeType [1009].

C++

```
#define NSSizeTypeParseChars(arValue, valueLength, szFormat, pValue)
NSSizeTypeParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NSSizeType [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.235 NSSizeTypeParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeParseStrOrChars function.

C++

```
#define NSSizeTypeParseStrOrChars N_FUNC_AW(NSSizeTypeParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.236 NSSizeTypeParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeParseVN function.

C++

```
#define NSSizeTypeParseVN N_FUNC_AW(NSSizeTypeParseVN)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.237 NSSizeTypeToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeToChars function.

C++

```
#define NSSizeTypeToChars N_FUNC_AW(NSSizeTypeToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.238 NSSizeTypeToString Macro

Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeToString function.

C++

```
#define NSSizeTypeToString N_FUNC_AW(NSSizeTypeToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.239 NSSizeTypeTryParse Macro

Converts NString [678] to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NSSizeTypeTryParse(szValue, szFormat, pValue, pResult)
NSSizeTypeTryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NSSizeType [1009].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.240 NSSizeTypeTryParseChars Macro

Converts an array of NChars to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NSSizeTypeTryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NSSizeTypeTryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

pValue	[out] A pointer to the converted NSSizeType [1009].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.241 NSSizeTypeTryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeTryParseStrOrChars function.

C++

```
#define NSSizeTypeTryParseStrOrChars N_FUNC_AW(NSSizeTypeTryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.242 NSSizeTypeTryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeTryParseVN function.

C++

```
#define NSSizeTypeTryParseVN N_FUNC_AW(NSSizeTypeTryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.243 NTrue Macro

True value for NBoolean [1004].

C++

```
#define NTrue 1
```

Module

NTypes Unit [905]

7.1.2.64.4.244 NUInt16Parse Macro

Converts NString [678] to NUInt16 [1009].

C++

```
#define NUInt16Parse(szValue, szFormat, pValue) NUInt16ParseStrOrChars(szValue, -1,
szFormat, pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] For internal use.
pValue	[out] A pointer to the converted NUInt16 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.245 NUInt16ParseChars Macro

Converts an array of NChars to NUInt16 [1009].

C++

```
#define NUInt16ParseChars(arValue, valueLength, szFormat, pValue)
NUInt16ParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt16 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.246 NUInt16ParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt16ParseStrOrChars function.

C++

```
#define NUInt16ParseStrOrChars N_FUNC_AW(NUInt16ParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.247 NUInt16ParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt16ParseVN function.

C++

```
#define NUInt16ParseVN N_FUNC_AW(NUInt16ParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.248 NUInt16ToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt16ToChars function.

C++

```
#define NUInt16ToChars N_FUNC_AW(NUInt16ToChars)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.249 NUInt16ToString Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt16ParseToString function.

C++

```
#define NUInt16ToString N_FUNC_AW(NUInt16ToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.250 NUInt16TryParse Macro

Converts NString [678] to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NUInt16TryParse(szValue, szFormat, pValue, pResult)
NUInt16TryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt16 [1009].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.251 NUInt16TryParseChars Macro

Converts an array of NChars to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NUInt16TryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NUInt16TryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt16 [1009].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.252 NUInt16TryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt16TryParseStrOrChars function.

C++

```
#define NUInt16TryParseStrOrChars N_FUNC_AW(NUInt16TryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.253 NUInt16TryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt16TryParseVN function.

C++

```
#define NUInt16TryParseVN N_FUNC_AW(NUInt16TryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.254 NUInt32Parse Macro

Converts NString [678] to NUInt32 [1009].

C++

```
#define NUInt32Parse(szValue, szFormat, pValue) NUInt32ParseStrOrChars(szValue, -1, style,
pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt32 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.255 NUInt32ParseChars Macro

Converts an array of NChars to NUInt32 [1009].

C++

```
#define NUInt32ParseChars(arValue, valueLength, szFormat, pValue)
NUInt32ParseStrOrChars(arValue, valueLength, style, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt32 [1009].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.256 NUInt32ParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt32ParseStrOrChars function.

C++

```
#define NUInt32ParseStrOrChars N_FUNC_AW(NUInt32ParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.257 NUInt32ParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt32ParseVN function.

C++

```
#define NUInt32ParseVN N_FUNC_AW(NUInt32ParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.258 NUInt32ToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt32ToChars function.

C++

```
#define NUInt32ToChars N_FUNC_AW(NUInt32ToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.259 NUInt32ToString Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt32ToString function.

C++

```
#define NUInt32ToString N_FUNC_AW(NUInt32ToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.260 NUInt32TryParse Macro

Converts NString [678] to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NUInt32TryParse(szValue, szFormat, pValue, pResult)
NUInt32TryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt32 [1009].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.261 NUInt32TryParseChars Macro

Converts an array of NChars to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NUInt32TryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NUInt32TryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt32 [1009].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.262 NUInt32TryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt32TryParseStrOrChars function.

C++

```
#define NUInt32TryParseStrOrChars N_FUNC_AW(NUInt32TryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.263 NUInt32TryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt32TryParseVN function.

C++

```
#define NUInt32TryParseVN N_FUNC_AW(NUInt32TryParseVN)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.264 NUInt64Parse Macro

Converts NString [678] to NUInt64 [1009].

C++

```
#define NUInt64Parse(szValue, szFormat, pValue) NUInt64ParseStrOrChars(szValue, -1,
szFormat, pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt64 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.265 NUInt64ParseChars Macro

Converts an array of NChars to NUInt64 [1009].

C++

```
#define NUInt64ParseChars(arValue, valueLength, szFormat, pValue)
NUInt64ParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string..
pValue	[out] A pointer to the converted NUInt64 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.266 NUInt64ParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt64ParseStrOrChars function.

C++

```
#define NUInt64ParseStrOrChars N_FUNC_AW(NUInt64ParseStrOrChars)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.267 NUInt64ParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt64ParseVN function.

C++

```
#define NUInt64ParseVN N_FUNC_AW(NUInt64ParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.268 NUInt64ToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt64ToChars function.

C++

```
#define NUInt64ToChars N_FUNC_AW(NUInt64ToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.269 NUInt64ToString Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt64ToString function.

C++

```
#define NUInt64ToString N_FUNC_AW(NUInt64ToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.270 NUInt64TryParse Macro

Converts NString [678] to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NUInt64TryParse(szValue, szFormat, pValue, pResult)
NUInt64TryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt64 [1009].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.271 NUInt64TryParseChars Macro

Converts an array of NChars to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NUInt64TryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NUInt64TryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt64 [1009].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.272 NUInt64TryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt64TryParseStrOrChars function.

C++

```
#define NUInt64TryParseStrOrChars N_FUNC_AW(NUInt64TryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.273 NUInt64TryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt64TryParseVN function.

C++

```
#define NUInt64TryParseVN N_FUNC_AW(NUInt64TryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.274 NUInt8Parse Macro

Converts NString [678] to NUInt8 [1009].

C++

```
#define NUInt8Parse(szValue, szFormat, pValue) NUInt8ParseStrOrChars(szValue, -1, szFormat,
pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt8 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.275 NUInt8ParseChars Macro

Converts an array of NChars to NUInt8 [1009].

C++

```
#define NUInt8ParseChars(arValue, valueLength, szFormat, pValue)
NUInt8ParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt8 [1009].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.276 NUInt8ParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt8ParseStrOrChars function.

C++

```
#define NUInt8ParseStrOrChars N_FUNC_AW(NUInt8ParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.277 NUInt8ParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt8ParseVN function.

C++

```
#define NUInt8ParseVN N_FUNC_AW(NUInt8ParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.278 NUInt8ToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt8ToChars function.

C++

```
#define NUInt8ToChars N_FUNC_AW(NUInt8ToChars)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.279 NUInt8ToString Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt8ToString function.

C++

```
#define NUInt8ToString N_FUNC_AW(NUInt8ToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.280 NUInt8TryParse Macro

Converts NString [678] to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NUInt8TryParse(szValue, szFormat, pValue, pResult)
NUInt8TryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt8 [1009].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.281 NUInt8TryParseChars Macro

Converts an array of NChars to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NUInt8TryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NUInt8TryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to numeric format string.
pValue	[out] A pointer to the converted NUInt8 [1009].
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.282 NUInt8TryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt8TryParseStrOrChars function.

C++

```
#define NUInt8TryParseStrOrChars N_FUNC_AW(NUInt8TryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.283 NUInt8TryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NUInt8TryParseVN function.

C++

```
#define NUInt8TryParseVN N_FUNC_AW(NUInt8TryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.284 NULL Macro

Null value for pointer.

C++

```
#define NULL 0
```

Module

NTypes Unit [905]

7.1.2.64.4.285 NURationalToString Macro

Picks a Unicode (W) or ASCII (A) version of the NURationalToString function.

C++

```
#define NURationalToString N_FUNC_AW(NURationalToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.286 NVersionRangeToString Macro

Picks a Unicode (W) or ASCII (A) version of the NVersionRangeToString function.

C++

```
#define NVersionRangeToString N_FUNC_AW(NVersionRangeToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.287 NVersionToString Macro

Picks a Unicode (W) or ASCII (A) version of the NVersionToString function.

C++

```
#define NVersionToString N_FUNC_AW(NVersionToString)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NTypes Unit [905]

7.1.2.64.4.288 NWCharParse Macro

Converts NString [678] to NWChar.

C++

```
#define NWCharParse(szValue, szFormat, pValue) NWCharParseStrOrChars(szValue, -1, szFormat, pValue)
```

Parameters

Parameters	Description
szValue	[in] A pointer to NString [678] to be converted.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NWChar.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.289 NWCharParseChars Macro

Converts an array of NChars to NWChar.

C++

```
#define NWCharParseChars(arValue, valueLength, szFormat, pValue)
NWCharParseStrOrChars(arValue, valueLength, szFormat, pValue)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NWChar.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.290 NWCharParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NWCharParseStrOrChars function.

C++

```
#define NWCharParseStrOrChars N_FUNC_AW(NWCharParseStrOrChars)
```

Module

NTypes Unit [905]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.64.4.291 NWCharParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NWCharParseVN function.

C++

```
#define NWCharParseVN N_FUNC_AW(NWCharParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.4.292 NWCharToChars Macro

Picks a Unicode (W) or ASCII (A) version of the NWCharToChars function.

C++

```
#define NWCharToChars N_FUNC_AW(NWCharToChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.293 NWCharToString Macro

Picks a Unicode (W) or ASCII (A) version of the NWCharToString function.

C++

```
#define NWCharToString N_FUNC_AW(NWCharToString)
```

Module

NTypes Unit [905]

7.1.2.64.4.294 NWCharTryParse Macro

Converts NString [678] to NWChar. A NBool [1004] value indicates whether the conversion succeeded.

C++

```
#define NWCharTryParse(szValue, szFormat, pValue, pResult)
NWCharTryParseStrOrChars(szValue, -1, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NWChar.
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.295 NWCharTryParseChars Macro

Converts an array of NChars to NWChar. A NBool [1004] value indicates whether the conversion succeeded.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NWCharTryParseChars(arValue, valueLength, szFormat, pValue, pResult)
NWCharTryParseStrOrChars(arValue, valueLength, szFormat, pValue, pResult)
```

Parameters

Parameters	Description
arValue	[in] A pointer to NChar [1005] array.
valueLength	[in] Length of the array.
szFormat	[in] Pointer to format string.
pValue	[out] A pointer to the converted NWChar.
pResult	[out] NTrue [1064] if the conversion succeeded; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NTypes Unit [905]

7.1.2.64.4.296 NWCharTryParseStrOrChars Macro

Picks a Unicode (W) or ASCII (A) version of the NWCharTryParseStrOrChars function.

C++

```
#define NWCharTryParseStrOrChars N_FUNC_AW(NWCharTryParseStrOrChars)
```

Module

NTypes Unit [905]

7.1.2.64.4.297 NWCharTryParseVN Macro

Picks a Unicode (W) or ASCII (A) version of the NWCharTryParseVN function.

C++

```
#define NWCharTryParseVN N_FUNC_AW(NWCharTryParseVN)
```

Module

NTypes Unit [905]

7.1.2.64.5 Files

7.1.2.64.5.1 NTypes.h

Header file for NTypes [905] module. Defines types and macros used in Neurotechnology components.

Enumerations

	Name	Description
	NAttributes_ [1002]	Enumerates value attributes.
	NOSFamily_ [1003]	Enumerates OS families.
	NAttributes [1002]	Enumerates value attributes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NOSFamily [1003]	Enumerates OS families.

Functions

	Name	Description
	NACharParseN [920]	Converts a NString [678] to NAChar [1004].
	NACharParseStrOrChars [921]	Converts an array of NChars to NAChar [1004].
	NACharParseVN [921]	Converts a NString [678] to NAChar [1004].
	NACharToChars [921]	Converts NAChar [1004] to an array of NAChars.
	NACharToString [922]	Converts NAChar [1004] to NString [678]
	NACharTryParseN [922]	Converts NString [678] to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.
	NACharTryParseStrOrChars [923]	Converts an array of NChars to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.
	NACharTryParseVN [923]	Converts NString [678] to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.
	NBooleanParseN [924]	Converts NString [678] to NBoolean [1004].
	NBooleanParseStrOrChars [924]	Converts an array of NChars to NBoolean [1004].
	NBooleanParseVN [924]	Converts NString [678] to NBoolean [1004].
	NBooleanToChars [925]	Converts NBoolean [1004] to an array of NChars.
	NBooleanToString [925]	Converts NBoolean [1004] to NString [678].
	NBooleanTryParseN [926]	Converts a NString [678] to NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.
	NBooleanTryParseStrOrChars [926]	Converts an array of NStrings or NChars into NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.
	NBooleanTryParseVN [927]	Converts a NString [678] to NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.
	NCharFromBinDigit [927]	Converts a binary digit to NChar [1005].
	NCharFromDigit [927]	Converts a digit to NChar [1005].
	NCharFromHexDigit [928]	Converts a hexadecimal digit to NChar [1005].
	NCharFromOctDigit [928]	Converts an octal digit to NChar [1005].
	NCharIsAscii [928]	Checks whether the specified char is ASCII.
	NCharIsBinDigit [929]	Checks whether the specified character is a binary digit.
	NCharIsDigit [929]	Checks whether the specified character is a digit.
	NCharIsHexDigit [929]	Checks whether the specified character is a hexadecimal digit.
	NCharIsLetter [930]	Checks whether the specified character is a letter.
	NCharIsLower [930]	Checks whether the specified character is a lower value.
	NCharIsOctDigit [930]	Checks whether the specified character is an octal digit.
	NCharIsUpper [930]	Checks whether the specified character is an upper value.
	NCharIsWhiteSpace [931]	Checks whether the specified character is a whitespace.
	NCharParseN [931]	Converts a NString [678] to a NChar [1005].
	NCharParseStrOrChars [931]	Converts an array of NStrings or NChars into NChar [1005].
	NCharParseVN [932]	Converts a NString [678] to a NChar [1005].
	NCharToBinDigit [932]	Converts a NChar [1005] to a binary digit.
	NCharToChars [933]	Converts a NChar [1005] into an array of NChars.
	NCharToDigit [933]	Converts a NChar [1005] to a digit.
	NCharToHexDigit [933]	Converts a NChar [1005] to a hexadecimal digit.
	NCharToLower [934]	Converts a NChar [1005] to lowercase.
	NCharToOctDigit [934]	Converts a NChar [1005] to an octal digit.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NCharToString [934]	Converts NChar [1005] to NString [678].
	NCharToUpper [935]	Converts NChar [1005] to uppercase.
	NCharTryParseN [935]	Converts a NString [678] to a NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.
	NCharTryParseStrOrChars [935]	Converts an array of NChars into NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.
	NCharTryParseVN [936]	Converts a NString [678] to a NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.
	NComplexToString [936]	Converts NComplex [1005] to NString [678].
	NDoubleGetNaN [937]	Retrieves a NDouble [1005] that is Not a Number (NaN).
	NDoubleGetNegativeInfinity [937]	Retrieves a NDouble [1005] that is NegativeInfinity.
	NDoubleGetPositiveInfinity [937]	Retrieves a NDouble [1005] that is PositiveInfinity.
	NDoubleIsInfinity [937]	Checks if NDouble [1005] evaluates to NegativeInfinity or PositiveInfinity.
	NDoubleIsNaN [938]	Checks if NDouble [1005] is Not a Number (NaN).
	NDoubleIsNegativeInfinity [938]	Checks if NDouble [1005] evaluates to NegativeInfinity.
	NDoubleIsPositiveInfinity [938]	Checks if NDouble [1005] evaluates to PositiveInfinity
	NDoubleParseN [939]	Converts a NString [678] into NDouble [1005].
	NDoubleParseStrOrChars [939]	Converted an array of NChars to NDouble [1005].
	NDoubleParseVN [939]	Converts a NString [678] into NDouble [1005].
	NDoubleToChars [940]	Converts NDouble [1005] to an array of NChars.
	NDoubleToString [940]	Converts NDouble [1005] to NString [678].
	NDoubleTryParseN [941]	Converts a NString [678] to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.
	NDoubleTryParseStrOrChars [941]	Converts an array of NChars to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.
	NDoubleTryParseVN [942]	Converts a NString [678] to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.
	NGuidParse [942]	Converts a NChar [1005] to NGuid.
	NGuidParseVN [943]	Converts a NString [678] to NGuid.
	NGuidToString [943]	Converts a NGuid to NString [678].
	NGuidTryParse [943]	Converts a NChar [1005] to NGuid. A NBool [1004] value indicates whether the conversion succeeded.
	NGuidTryParseVN [944]	Converts a NString [678] to NGuid. A NBool [1004] value indicates whether the conversion succeeded.
	NIndexPairToString [944]	Convertes NIndexPair [1006] to NString [678].
	NInt16ParseN [945]	Convertes NString [678] to NInt16 [1006].
	NInt16ParseStrOrChars [945]	Converts an array of NChars to NInt16 [1006].
	NInt16ParseVN [946]	Convertes NString [678] to NInt16 [1006].
	NInt16ToChars [946]	Converts NInt16 [1006] to an array of NChars.
	NInt16ToString [946]	Converts NInt16 [1006] to NString [678].
	NInt16TryParseN [947]	Converts NString [678] to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt16TryParseStrOrChars [947]	Converts an array of NChars to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt16TryParseVN [948]	Converts NString [678] to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt32ParseN [948]	Converts NString [678] to NInt32 [1006].
	NInt32ParseStrOrChars [949]	Converts an array of NChars to NInt32 [1006].
	NInt32ParseVN [949]	Converts NString [678] to NInt32 [1006].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NInt32ToChars [949]	Converts NInt32 [1006] to an array of NChars.
	NInt32ToString [950]	Converts NInt32 [1006] to NString [678].
	NInt32TryParseN [950]	Converts NString [678] to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt32TryParseStrOrChars [951]	Converts an array of NChars to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt32TryParseVN [951]	Converts NString [678] to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt64ParseN [952]	Converts NString [678] to NInt64 [1006].
	NInt64ParseStrOrChars [952]	Converts an array of NChars to NInt64 [1006].
	NInt64ParseVN [952]	Converts NString [678] to NInt64 [1006].
	NInt64ToChars [953]	Converts NInt64 [1006] to an array of NChars.
	NInt64ToString [953]	Converts NInt64 [1006] to NString [678].
	NInt64TryParseN [954]	Converts NString [678] to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt64TryParseStrOrChars [954]	Converts an array of NChars to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt64TryParseVN [955]	Converts NString [678] to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt8ParseN [955]	Converts NString [678] to NInt8 [1007].
	NInt8ParseStrOrChars [956]	Converts an array of NChars to NInt8 [1007].
	NInt8ParseVN [956]	Converts NString [678] to NInt8 [1007].
	NInt8ToChars [956]	Converts NInt8 [1007] to an array of NChars.
	NInt8ToString [957]	Converts NInt8 [1007] to NString [678].
	NInt8TryParseN [957]	Converts NString [678] to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt8TryParseStrOrChars [958]	Converts an array of NChars to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.
	NInt8TryParseVN [958]	Converts NString [678] to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.
	NNameStringPairCopy [959]	Creates a copy of string pair structure.
	NNameStringPairCreate [959]	Creates a new string pair structure.
	NNameStringPairDispose [959]	Disposes of the specified string pair structure.
	NNameStringPairSet [960]	Sets the value of the specified string pair structure.
	NOSFamilyGetCurrent [960]	Retrieves the current operating system family.
	NPointerParseN [960]	Converts NString [678] to NPointer.
	NPointerParseStrOrChars [961]	Converts an array of NChars to NPointer.
	NPointerParseVN [961]	Converts NString [678] to NPointer.
	NPointerToChars [961]	Converts NPointer to an array of NChars.
	NPointerToString [962]	Converts NPointer to NString [678].
	NPointerTryParseN [962]	Converts NString [678] to NPointer. A NBool [1004] value indicates whether the conversion succeeded.
	NPointerTryParseStrOrChars [963]	Converts an array of NChars to NPointer. A NBool [1004] value indicates whether the conversion succeeded.
	NPointerTryParseVN [963]	Converts NString [678] to NPointer. A NBool [1004] value indicates whether the conversion succeeded.
	NRangeToString [964]	Converts NRange to NString [678].
	NRationalToString [964]	Converts NRational [1007] to NString [678].
	NResultParseN [964]	Converts NString [678] to NResult [1007].
	NResultParseStrOrChars [965]	Converts an array of NChars to NResult [1007].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NResultParseVN [965]	Converts NString [678] to NResult [1007].
	NResultToChars [966]	Converts NResult [1007] to an array of NChars.
	NResultToString [966]	Converts NResult [1007] to NString [678].
	NResultTryParseN [967]	Converts NString [678] to NResult [1007]. A NBool [1004] value indicates whether the conversion succeeded.
	NResultTryParseStrOrChars [967]	Converts an array of NChars to NResult [1007]. A NBool [1004] value indicates whether the conversion succeeded.
	NResultTryParseVN [967]	Converts NString [678] to NResult [1007]. A NBool [1004] value indicates whether the conversion succeeded.
	NSingleGetNaN [968]	Retrieves a NSingle [1008] that is Not a Number (NaN).
	NSingleGetNegativeInfinity [968]	Retrieves a NSingle [1008] that is NegativeInfinity.
	NSingleGetPositiveInfinity [968]	Retrieves a NSingle [1008] that is PositiveInfinity.
	NSingleIsInfinity [969]	Checks if NSingle [1008] evaluates to NegativeInfinity or PositiveInfinity.
	NSingleIsNaN [969]	Checks if NSingle [1008] Not a Number (NaN).
	NSingleIsNegativeInfinity [969]	Checks if NSingle [1008] evaluates to NegativeInfinity.
	NSingleIsPositiveInfinity [970]	Checks if NSingle [1008] evaluates to PositiveInfinity.
	NSingleParseN [970]	Converts NString [678] to NSingle [1008].
	NSingleParseStrOrChars [970]	Converts an array of NChars to NSingle [1008].
	NSingleParseVN [971]	Converts NString [678] to NSingle [1008].
	NSingleToChars [971]	Converts NSingle [1008] to an array of NChars.
	NSingleToString [972]	Converts NSingle [1008] to NString [678].
	NSingleTryParseN [972]	Converts NString [678] to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.
	NSingleTryParseStrOrChars [972]	Converts an array of NChars to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.
	NSingleTryParseVN [973]	Converts NString [678] to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.
	NSizeTypeParseN [973]	Converts NString [678] to NSizeType [1008].
	NSizeTypeParseStrOrChars [974]	Converts an array of NChars to NSizeType [1008].
	NSizeTypeParseVN [974]	Converts NString [678] to NSizeType [1008].
	NSizeTypeToChars [975]	Converts NSizeType [1008] to an array of NChars.
	NSizeTypeToString [975]	Converts NSizeType [1008] to NString [678].
	NSizeTypeTryParseN [975]	Converts NString [678] to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.
	NSizeTypeTryParseStrOrChars [976]	Converts an array of NChars to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.
	NSizeTypeTryParseVN [976]	Converts NString [678] to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.
	NSSizeTypeParseN [977]	Converts NString [678] to NSSizeType [1009].
	NSSizeTypeParseStrOrChars [977]	Converts an array of NChars to NSSizeType [1009].
	NSSizeTypeParseVN [978]	Converts NString [678] to NSSizeType [1009].
	NSSizeTypeToChars [978]	Converts NSSizeType [1009] to an array of NChars.
	NSSizeTypeToString [979]	Converts NSSizeType [1009] to NString [678].
	NSSizeTypeTryParseN [979]	Converts NString [678] to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NSSizeTypeTryParseStrOrChars [979]	Converts an array of NChars to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NSSizeTypeTryParseVN [980]	Converts NString [678] to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NTypesGetPointer [980]	Retrieves a pointer to object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NUInt16ParseN [981]	Converts NString [678] to NUInt16 [1009].
	NUInt16ParseStrOrChars [981]	Converts an array of NChars to NUInt16 [1009].
	NUInt16ParseVN [981]	Converts NString [678] to NUInt16 [1009].
	NUInt16ToChars [982]	Converts NUInt16 [1009] to an array of NChars.
	NUInt16ToString [982]	Converts NUInt16 [1009] to NString [678].
	NUInt16TryParseN [983]	Converts NString [678] to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt16TryParseStrOrChars [983]	Converts an array of NChars to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt16TryParseVN [984]	Converts NString [678] to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt32ParseN [984]	Converts NString [678] to NUInt32 [1009].
	NUInt32ParseStrOrChars [984]	Converts an array of NChars to NUInt32 [1009].
	NUInt32ParseVN [985]	Converts NString [678] to NUInt32 [1009].
	NUInt32ToChars [985]	Converts NUInt32 [1009] to an array of NChars.
	NUInt32ToString [986]	Converts NUInt32 [1009] to NString [678].
	NUInt32TryParseN [986]	Converts NString [678] to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt32TryParseStrOrChars [987]	Converts an array of NChars to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt32TryParseVN [987]	Converts NString [678] to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt64ParseN [988]	Converts NString [678] to NUInt64 [1009].
	NUInt64ParseStrOrChars [988]	Converts an array of NChars to NUInt64 [1009].
	NUInt64ParseVN [988]	Converts NString [678] to NUInt64 [1009].
	NUInt64ToChars [989]	Converts NUInt64 [1009] to an array of NChars.
	NUInt64ToString [989]	Converts NUInt64 [1009] to NString [678].
	NUInt64TryParseN [990]	Converts NString [678] to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt64TryParseStrOrChars [990]	Converts an array of NChars to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt64TryParseVN [991]	Converts NString [678] to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt8ParseN [991]	Converts NString [678] to NUInt8 [1009].
	NUInt8ParseStrOrChars [991]	Converts an array of NChars to NUInt8 [1009].
	NUInt8ParseVN [992]	Converts NString [678] to NUInt8 [1009].
	NUInt8ToChars [992]	Converts NUInt8 [1009] to an array of NChars.
	NUInt8ToString [993]	Converts NUInt8 [1009] to NString [678].
	NUInt8TryParseN [993]	Converts NString [678] to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt8TryParseStrOrChars [994]	Converts an array of NChars to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NUInt8TryParseVN [994]	Converts NString [678] to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
	NURationalToString [995]	Converts NURational [1010] to NString [678].
	NVersionRangeContains [995]	Checks if NVersionRange [1011] contains a certain version.
	NVersionRangeContainsRange [995]	Checks if NVersionRange [1011] contains a certain version range.
	NVersionRangeIntersect [996]	Replaces NVersionRange [1011] with the intersection of itself and another NVersionRange [1011].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NVersionRangeIntersectsWith [996]	Returns third NVersionRange [1011] that represents the intersection of two other NVersionRanges. If there is no intersection, an empty NVersionRange [1011] is returned.
	NVersionRangeToString [996]	Converts NVersionRange [1011] to NString [678].
	NVersionToString [997]	Converts NVersion [1010] to NString [678].
	NWCharParseN [997]	Converts NString [678] to NWChar.
	NWCharParseStrOrChars [998]	Converts an array of NChars to NWChar.
	NWCharParseVN [998]	Converts NString [678] to NWChar.
	NWCharToChars [998]	Converts NWChar to an array of NChars.
	NWCharToString [999]	Converts NWChar to NString [678].
	NWCharTryParseN [999]	Converts NString [678] to NWChar. A NBool [1004] value indicates whether the conversion succeeded.
	NWCharTryParseStrOrChars [1000]	Converts an array of NChars to NWChar. A NBool [1004] value indicates whether the conversion succeeded.
	NWCharTryParseVN [1000]	Converts NString [678] to NWChar. A NBool [1004] value indicates whether the conversion succeeded.

Macros

Name	Description
N_AW [1011]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version in a macro (with either 'A' or 'W' suffix accordingly).
N_BOOLEAN_FALSE_STRING [1011]	ANSI or Unicode (if N_UNICODE [384] is defined) string, representing NFalse [1041] Boolean value.
N_BOOLEAN_FALSE_STRINGA [1011]	ANSI string, representing NFalse [1041] Boolean value.
N_BOOLEAN_FALSE_STRINGW [1012]	Unicode (if N_UNICODE [384] is defined) string, representing NFalse [1041] Boolean value.
N_BOOLEAN_TRUE_STRING [1012]	ANSI or Unicode (if N_UNICODE [384] is defined) string, representing NTrue [1064] Boolean value.
N_BOOLEAN_TRUE_STRINGA [1012]	ANSI string, representing NTrue [1064] Boolean value.
N_BOOLEAN_TRUE_STRINGW [1012]	Unicode (if N_UNICODE [384] is defined) string, representing NTrue [1064] Boolean value.
N_BYTE_MAX [1012]	Maximum value for NByte [1005].
N_BYTE_MIN [1013]	Minimum value for NByte [1005].
N_CALLBACK_AW [1013]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the callback (with either 'A' or 'W' suffix accordingly).
N_DECLARE_HANDLE [1011]	Declares handle with specified name.
N_DECLARE_HANDLE_TYPE [1013]	Declares handle type with specified name.
N_DECLARE_OBJECT_TYPE [1013]	Declares object type with specified name.
N_DECLARE_STATIC_OBJECT_TYPE [1013]	Declares static object type with specified name.
N_DECLARE_TYPE [1014]	Declares type with specified name.
N_DECLATE_PRIMITIVE [1014]	Declares primitive with specified name.
N_DOUBLE_EPSILON [1014]	Epsilon value for NDouble [1005].
N_DOUBLE_MAX [1014]	Maximum value for NDouble [1005].
N_DOUBLE_MIN [1014]	Minimum value for NDouble [1005].
N_EMPTY_STRING [1014]	ANSI or Unicode (if N_UNICODE [384] is defined) string, representing empty string value.
N_EMPTY_STRINGA [1015]	ANSI string, representing empty string value.
N_EMPTY_STRINGW [1015]	Unicode (if N_UNICODE [384] is defined) string, representing empty string value.
N_FIELD_AW [1015]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the field (with either 'A' or 'W' suffix accordingly).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_FLOAT_EPSILON [1015]	Epsilon value for NFloat [1005].
N_FLOAT_INFINITY_STRING [1015]	ANSI or Unicode (if N_UNICODE [384] is defined) string, representing "Infinity" floating-point value.
N_FLOAT_INFINITY_STRINGA [1015]	ANSI string, representing "Infinity" floating-point value.
N_FLOAT_INFINITY_STRINGW [1016]	Unicode (if N_UNICODE [384] is defined) string, representing "Infinity" floating-point value.
N_FLOAT_MAX [1016]	Maximum value for NFloat [1005].
N_FLOAT_MIN [1016]	Minimum value for NFloat [1005].
N_FLOAT_NAN_STRING [1016]	ANSI or Unicode (if N_UNICODE [384] is defined) string, representing "NaN" floating-point value.
N_FLOAT_NAN_STRINGA [1016]	ANSI string, representing "NaN" floating-point value.
N_FLOAT_NAN_STRINGW [1017]	Unicode (if N_UNICODE [384] is defined) string, representing "NaN" floating-point value.
N_FUNC_AW [1017]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the function (with either 'A' or 'W' suffix accordingly).
N_INT_MAX [1017]	Maximum value for NInt [1006].
N_INT_MIN [1017]	Minimum value for NInt [1006].
N_INT16_MAX [1017]	Maximum value for NShort [1008].
N_INT16_MIN [1017]	Minimum value for NShort [1008].
N_INT32_MAX [1018]	Maximum value for NInt [1006].
N_INT32_MIN [1018]	Minimum value for NInt [1006].
N_INT64_MAX [1018]	Maximum value for NInt64 [1006].
N_INT64_MIN [1018]	Minimum value for NInt64 [1006].
N_INT8_MAX [1018]	Maximum value for NSByte [1008].
N_INT8_MIN [1018]	Minimum value for NSByte [1008].
N_JOIN_SYMBOLS [1019]	Concatenates two symbols, yielding a single concatenated symbol.
N_JOIN_SYMBOLS_ [1019]	Concatenates two symbols, yielding a single concatenated symbol. Can serve as a macro argument.
N_JOIN_SYMBOLS2 [1019]	Concatenates two symbols, yielding a single concatenated symbol.
N_JOIN_SYMBOLS2_ [1019]	Concatenates two symbols, yielding a single concatenated symbol. Can serve as a macro argument.
N_LONG_MAX [1019]	Maximum value for NLong [1007].
N_LONG_MIN [1020]	Minimum value for NLong [1007].
N_MACRO_AW [1020]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the macro (with either 'A' or 'W' suffix accordingly).
N_PTR_SIZE [1020]	Defines pointer size.
N_SBYTE_MAX [1020]	Maximum value for NSByte [1008].
N_SBYTE_MIN [1020]	Minimum value for NSByte [1008].
N_SHORT_MAX [1020]	Maximum value for NShort [1008].
N_SHORT_MIN [1021]	Minimum value for NShort [1008].
N_SINGLE_EPSILON [1021]	Epsilon value for NSingle [1008].
N_SINGLE_MAX [1021]	Maximum value for NSingle [1008].
N_SINGLE_MIN [1021]	Minimum value for NSingle [1008].
N_SIZE_TYPE_MAX [1021]	Maximum value for NSizeType [1008].
N_SIZE_TYPE_MIN [1021]	Minimum value for NSizeType [1008].
N_SSIZE_TYPE_MAX [1022]	Maximum value for NSSizeType [1009].
N_SSIZE_TYPE_MIN [1022]	Minimum value for NSSizeType [1009].
N_STRINGIZE [1022]	Defines the size of ANSI or Unicode (if N_UNICODE [384] is defined) string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_STRINGIZE_ [1022]	Defines the size of ANSI or Unicode (if N_UNICODE [384] is defined) string. Can serve as a macro argument.
N_STRINGIZE_A [1022]	Defines the size of ANSI string. Can serve as a macro argument.
N_STRINGIZE_W [1023]	Defines the size of Unicode (if N_UNICODE [384] is defined) string. Can serve as a macro argument.
N_STRINGIZEA [1023]	Defines the size of ANSI string.
N_STRINGIZEW [1023]	Defines the size of Unicode (if N_UNICODE [384] is defined) string.
N_STRUCT_AW [1023]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the struct (with either 'A' or 'W' suffix accordingly).
N_SYMBOL_AW [1023]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the symbol (with either 'A' or 'W' suffix accordingly).
N_T [1023]	Makes either ANSI or Unicode (if N_UNICODE [384] is defined) string or character constant.
N_T_A [1024]	Makes ANSI string or character constant. Can serve as a macro argument.
N_T_W [1024]	Makes Unicode (if N_UNICODE [384] is defined) string or character constant. Can serve as a macro argument.
N_TA [1024]	Makes ANSI string or character constant.
N_TW [1024]	Makes Unicode (if N_UNICODE [384] is defined) string or character constant.
N_TYPE_AW [1024]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the type (with either 'A' or 'W' suffix accordingly).
N_TYPE_OF [1024]	Used to obtain a pointer to the function that retrieves NType [871] object for the specified type name.",
N_UINT_MAX [1025]	Maximum value for NUInt [1009].
N_UINT_MIN [1025]	Minimum value for NUInt [1009].
N_UINT16_MAX [1025]	Maximum value for NUShort [1010].
N_UINT16_MIN [1025]	Minimum value for NUShort [1010].
N_UINT32_MAX [1025]	Maximum value for NUInt [1009].
N_UINT32_MIN [1026]	Minimum value for NUInt [1009].
N_UINT64_MAX [1026]	Maximum value for NULong [1010].
N_UINT64_MIN [1026]	Minimum value for NULong [1010].
N_UINT8_MAX [1026]	Maximum value for NUInt8 [1009].
N_UINT8_MIN [1026]	Minimum value for NUInt8 [1009].
N_ULONG_MAX [1026]	Maximum value for NULong [1010].
N_ULONG_MIN [1027]	Minimum value for NULong [1010].
N_UNREFERENCED_PARAMETER [1027]	Suppresses the warning "unreferenced parameter".
N_UNUSED_VARIABLE [1027]	Suppresses the warning "unused variable".
N_USHORT_MAX [1027]	Maximum value for NUShort [1010].
N_USHORT_MIN [1027]	Minimum value for NUShort [1010].
N_VAR_AW [1027]	Picks either ANSI or Unicode (if N_UNICODE [384] is defined) version of the variable (with either 'A' or 'W' suffix accordingly).
N_WCHAR_SIZE [1028]	Defines the size of Unicode char.
NACharParse [1028]	Converts a NString [678] to NAChar [1004].
NACharParseChars [1028]	Converts an array of NChars to NAChar [1004].
NACharParseStrOrChars [1029]	Picks a Unicode (W) or ASCII (A) version of the NACharParseStrOrChars function.
NACharParseVN [1029]	Picks a Unicode (W) or ASCII (A) version of the NACharParseVN function.
NACharToChars [1029]	Picks a Unicode (W) or ASCII (A) version of the NACharToChars function.
NACharToString [1029]	Picks a Unicode (W) or ASCII (A) version of the NACharToString function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NACharTryParse [1029]	Converts NString [678] to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.
NACharTryParseChars [1030]	Converts an array of NChars to NAChar [1004]. A NBool [1004] value indicates whether the conversion succeeded.
NACharTryParseStrOrChars [1030]	Picks a Unicode (W) or ASCII (A) version of the NACharTryParseStrOrChars function.
NACharTryParseVN [1030]	Picks a Unicode (W) or ASCII (A) version of the NACharTryParseVN function.
NBooleanParse [1030]	Converts NString [678] to NBoolean [1004].
NBooleanParseChars [1031]	Converts an array of NChars to NBoolean [1004].
NBooleanParseStrOrChars [1031]	Picks a Unicode (W) or ASCII (A) version of the NBooleanParseStrOrChars function.
NBooleanParseVN [1031]	Picks a Unicode (W) or ASCII (A) version of the NBooleanParseVN function.
NBooleanToChars [1032]	Picks a Unicode (W) or ASCII (A) version of the NBooleanToChars function.
NBooleanToString [1032]	Picks a Unicode (W) or ASCII (A) version of the NBooleanToString function.
NBooleanTryParse [1032]	Converts NString [678] to NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.
NBooleanTryParseChars [1032]	Converts an array of NChars to NBoolean [1004]. A NBool [1004] value indicates whether the conversion succeeded.
NBooleanTryParseStrOrChars [1033]	Picks a Unicode (W) or ASCII (A) version of the NBooleanTryParseStrOrChars function.
NBooleanTryParseVN [1033]	Picks a Unicode (W) or ASCII (A) version of the NBooleanTryParseVN function.
NCharFromBinDigit [1033]	Picks a Unicode (W) or ASCII (A) version of the NCharFromBinDigit function.
NCharFromDigit [1033]	Picks a Unicode (W) or ASCII (A) version of the NCharFromDigit function.
NCharFromHexDigit [1033]	Picks a Unicode (W) or ASCII (A) version of the NCharFromHexDigit function.
NCharFromOctDigit [1034]	Picks a Unicode (W) or ASCII (A) version of the NCharFromOctDigit function.
NCharIsAscii [1034]	Picks a Unicode (W) or ASCII (A) version of the NCharIsAscii function.
NCharIsBinDigit [1034]	Picks a Unicode (W) or ASCII (A) version of the NCharIsBinDigit function.
NCharIsDigit [1034]	Picks a Unicode (W) or ASCII (A) version of the NCharIsDigit function.
NCharIsHexDigit [1034]	Picks a Unicode (W) or ASCII (A) version of the NCharIsHexDigit function.
NCharIsLetter [1034]	Picks a Unicode (W) or ASCII (A) version of the NCharIsLetter function.
NCharIsLower [1035]	Picks a Unicode (W) or ASCII (A) version of the NCharIsLower function.
NCharIsOctDigit [1035]	Picks a Unicode (W) or ASCII (A) version of the NCharIsOctDigit function.
NCharIsUpper [1035]	Picks a Unicode (W) or ASCII (A) version of the NCharIsUpper function.
NCharIsWhiteSpace [1035]	Picks a Unicode (W) or ASCII (A) version of the NCharIsWhiteSpace function.
NCharParse [1035]	Converts NString [678] to NChar [1005].
NCharParseChars [1036]	Converts an array of NChars to NBoolean [1004].
NCharToBinDigit [1036]	Picks a Unicode (W) or ASCII (A) version of the NCharToBinDigit function.
NCharToDigit [1036]	Picks a Unicode (W) or ASCII (A) version of the NCharToDigit function.
NCharToHexDigit [1036]	Picks a Unicode (W) or ASCII (A) version of the NCharToHexDigit function.
NCharToLower [1037]	Picks a Unicode (W) or ASCII (A) version of the NCharToLower function.
NCharToOctDigit [1037]	Picks a Unicode (W) or ASCII (A) version of the NCharToOctDigit function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCharToUpper [1037]	Picks a Unicode (W) or ASCII (A) version of the NCharToUpper function.
NCharTryParse [1037]	Converts NString [678] to NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.
NCharTryParseChars [1038]	Converts an array of NChars to NChar [1005]. A NBool [1004] value indicates whether the conversion succeeded.
NDoubleParse [1038]	Converts NString [678] to NDouble [1005].
NDoubleParseChars [1038]	Converts an array of NChars to NChar [1005].
NDoubleParseStrOrChars [1039]	Picks a Unicode (W) or ASCII (A) version of the NDoubleParseStrOrChars function.
NDoubleParseVN [1039]	Picks a Unicode (W) or ASCII (A) version of the NDoubleParseVN function.
NDoubleToChars [1039]	Picks a Unicode (W) or ASCII (A) version of the NDoubleToChars function.
NDoubleToString [1039]	Picks a Unicode (W) or ASCII (A) version of the NDoubleToString function.
NDoubleTryParse [1039]	Converts NString [678] to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.
NDoubleTryParseChars [1040]	Converts an array of NChars to NDouble [1005]. A NBool [1004] value indicates whether the conversion succeeded.
NDoubleTryParseStrOrChars [1040]	Picks a Unicode (W) or ASCII (A) version of the NDoubleTryParseStrOrChars function.
NDoubleTryParseVN [1040]	Picks a Unicode (W) or ASCII (A) version of the NDoubleTryParseVN function.
NFalse [1041]	False value for NBoolean [1004].
NGuidParse [1041]	Picks a Unicode (W) or ASCII (A) version of the NGuidParse function.
NGuidParseVN [1041]	Picks a Unicode (W) or ASCII (A) version of the NGuidParseVN function.
NGuidToString [1041]	Picks a Unicode (W) or ASCII (A) version of the NGuidToString function.
NGuidTryParse [1041]	Picks a Unicode (W) or ASCII (A) version of the NGuidTryParse function.
NGuidTryParseVN [1041]	Picks a Unicode (W) or ASCII (A) version of the NGuidTryParseVN function.
NIndexPairToString [1042]	Picks a Unicode (W) or ASCII (A) version of the NIndexPairToString function.
NInt16Parse [1042]	Converts NString [678] to NInt16 [1006].
NInt16ParseChars [1042]	Converts an array of NChars to NInt16 [1006].
NInt16ParseStrOrChars [1043]	Picks a Unicode (W) or ASCII (A) version of the NInt16ParseStrOrChars function.
NInt16ParseVN [1043]	Picks a Unicode (W) or ASCII (A) version of the NInt16ParseVN function.
NInt16ToChars [1043]	Picks a Unicode (W) or ASCII (A) version of the NInt16ToChars function.
NInt16ToString [1043]	Picks a Unicode (W) or ASCII (A) version of the NInt16ToString function.
NInt16TryParse [1043]	Converts NString [678] to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
NInt16TryParseChars [1044]	Converts an array of NChars to NInt16 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
NInt16TryParseStrOrChars [1044]	Picks a Unicode (W) or ASCII (A) version of the NInt16TryParseStrOrChars function.
NInt16TryParseVN [1044]	Picks a Unicode (W) or ASCII (A) version of the NInt16TryParseVN function.
NInt32Parse [1044]	Converts NString [678] to NInt32 [1006].
NInt32ParseChars [1045]	Converts an array of NChars to NInt32 [1006].
NInt32ParseStrOrChars [1045]	Picks a Unicode (W) or ASCII (A) version of the NInt32ParseStrOrChars function.
NInt32ParseVN [1045]	Picks a Unicode (W) or ASCII (A) version of the NInt32ParseVN function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt32ToChars [1046]	Picks a Unicode (W) or ASCII (A) version of the NInt32ToChars function.
NInt32ToString [1046]	Picks a Unicode (W) or ASCII (A) version of the NInt32ToString function.
NInt32TryParse [1046]	Converts NString [678] to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
NInt32TryParseChars [1046]	Converts an array of NChars to NInt32 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
NInt32TryParseStrOrChars [1047]	Picks a Unicode (W) or ASCII (A) version of the NInt32TryParseStrOrChars function.
NInt32TryParseVN [1047]	Picks a Unicode (W) or ASCII (A) version of the NInt32tryParseVN function.
NInt64Parse [1047]	Converts NString [678] to NInt64 [1006].
NInt64ParseChars [1047]	Converts an array of NChars to NInt64 [1006].
NInt64ParseStrOrChars [1048]	Picks a Unicode (W) or ASCII (A) version of the NInt64ParseStrOrChars function.
NInt64ParseVN [1048]	Picks a Unicode (W) or ASCII (A) version of the NInt64ParseVN function.
NInt64ToChars [1048]	Picks a Unicode (W) or ASCII (A) version of the NInt64ToChars function.
NInt64ToString [1048]	Picks a Unicode (W) or ASCII (A) version of the NInt64ToString function.
NInt64TryParse [1048]	Converts NString [678] to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
NInt64TryParseChars [1049]	Converts an array of NChars to NInt64 [1006]. A NBool [1004] value indicates whether the conversion succeeded.
NInt64TryParseStrOrChars [1049]	Picks a Unicode (W) or ASCII (A) version of the NInt64TryParseStrOrChars function.
NInt64TryParseVN [1049]	Picks a Unicode (W) or ASCII (A) version of the NInt64TryParseVN function.
NInt8Parse [1050]	Converts NString [678] to NInt8 [1007].
NInt8ParseChars [1050]	Converts an array of NChars to NInt8 [1007].
NInt8ParseStrOrChars [1050]	Picks a Unicode (W) or ASCII (A) version of the NInt8ParseStrOrChars function.
NInt8ParseVN [1051]	Picks a Unicode (W) or ASCII (A) version of the NInt8ParseVN function.
NInt8ToChars [1051]	Picks a Unicode (W) or ASCII (A) version of the NInt8ToChars function.
NInt8ToString [1051]	Picks a Unicode (W) or ASCII (A) version of the NInt8ToString function.
NInt8TryParse [1051]	Converts NString [678] to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.
NInt8TryParseChars [1051]	Converts an array of NChars to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.
NInt8TryParseStrOrChars [1052]	Picks a Unicode (W) or ASCII (A) version of the NInt8TryParseStrOrChars function.
NInt8TryParseVN [1052]	Picks a Unicode (W) or ASCII (A) version of the NInt8TryParseVN function.
NPointerParse [1052]	Converts NString [678] to NPointer.
NPointerParseChars [1053]	Converts an array of NChars to NPointer.
NPointerParseStrOrChars [1053]	Picks a Unicode (W) or ASCII (A) version of the NPointerParseStrOrChars function.
NPointerParseVN [1053]	Picks a Unicode (W) or ASCII (A) version of the NPointerParseVN function.
NPointerToChars [1053]	Picks a Unicode (W) or ASCII (A) version of the NPointerToChars function.
NPointerToString [1054]	Picks a Unicode (W) or ASCII (A) version of the NPointerToString function.
NPointerTryParse [1054]	Converts NString [678] to NInt8 [1007]. A NBool [1004] value indicates whether the conversion succeeded.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPointerTryParseChars [1054]	Converts an array of NChars to NPointer. A NBool [1004] value indicates whether the conversion succeeded.
NPointerTryParseStrOrChars [1055]	Picks a Unicode (W) or ASCII (A) version of the NPointerTryParseStrOrChars function.
NPointerTryParseVN [1055]	Picks a Unicode (W) or ASCII (A) version of the NPointerTryParseVN function.
NRangeToString [1055]	Picks a Unicode (W) or ASCII (A) version of the NRangeToString function.
NRationalToString [1055]	Picks a Unicode (W) or ASCII (A) version of the NRationalToString function.
NResultParseVN [1055]	Picks a Unicode (W) or ASCII (A) version of the NResultParseVN function.
NResultToChars [1055]	Picks a Unicode (W) or ASCII (A) version of the NResultToChars function.
NResultToString [1056]	Picks a Unicode (W) or ASCII (A) version of the ResultToString function.
NResultTryParseStrOrChars [1056]	Picks a Unicode (W) or ASCII (A) version of the NResultTryParseStrOrChars function.
NResultTryParseVN [1056]	Picks a Unicode (W) or ASCII (A) version of the NResultTryParseVN function.
NSingleParse [1056]	Converts NString [678] to NSingle [1008].
NSingleParseChars [1056]	Converts an array of NChars to NSingle [1008].
NSingleParseStrOrChars [1057]	Picks a Unicode (W) or ASCII (A) version of the NSingleParseStrOrChars function.
NSingleParseVN [1057]	Picks a Unicode (W) or ASCII (A) version of the NSingleParseVN function.
NSingleToChars [1057]	Picks a Unicode (W) or ASCII (A) version of the NSingleToChars function.
NSingleToString [1057]	Picks a Unicode (W) or ASCII (A) version of the NSingleToString function.
NSingleTryParse [1058]	Converts NString [678] to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.
NSingleTryParseChars [1058]	Converts an array of NChars to NSingle [1008]. A NBool [1004] value indicates whether the conversion succeeded.
NSingleTryParseStrOrChars [1058]	Picks a Unicode (W) or ASCII (A) version of the NSingleTryParseStrOrChars function.
NSingleTryParseVN [1059]	Picks a Unicode (W) or ASCII (A) version of the NSingleTryParseVN function.
NSizeTypeParse [1059]	Converts NString [678] to NSizeType [1008].
NSizeTypeParseChars [1059]	Converts an array of NChars to NSizeType [1008].
NSizeTypeParseStrOrChars [1059]	Picks a Unicode (W) or ASCII (A) version of the NSizeTypeParseStrOrChars function.
NSizeTypeParseVN [1060]	Picks a Unicode (W) or ASCII (A) version of the NSizeTypeParseVN function.
NSizeTypeToChars [1060]	Picks a Unicode (W) or ASCII (A) version of the NSizeTypeToChars function.
NSizeTypeToString [1060]	Picks a Unicode (W) or ASCII (A) version of the NSizeTypeToString function.
NSizeTypeTryParse [1060]	Converts NString [678] to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.
NSizeTypeTryParseChars [1061]	Converts an array of NChars to NSizeType [1008]. A NBool [1004] value indicates whether the conversion succeeded.
NSizeTypeTryParseStrOrChars [1061]	Picks a Unicode (W) or ASCII (A) version of the NSizeTypeTryParseStrOrChars function.
NSizeTypeTryParseVN [1061]	Picks a Unicode (W) or ASCII (A) version of the NSizeTypeTryParseVN function.
NSSizeTypeParse [1061]	Converts NString [678] to NSSizeType [1009].
NSSizeTypeParseChars [1062]	Converts an array of NChars to NSSizeType [1009].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSSizeTypeParseStrOrChars [1062]	Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeParseStrOrChars function.
NSSizeTypeParseVN [1062]	Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeParseVN function.
NSSizeTypeToChars [1063]	Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeToChars function.
NSSizeTypeToString [1063]	Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeToString function.
NSSizeTypeTryParse [1063]	Converts NString [678] to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NSSizeTypeTryParseChars [1063]	Converts an array of NChars to NSSizeType [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NSSizeTypeTryParseStrOrChars [1064]	Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeTryParseStrOrChars function.
NSSizeTypeTryParseVN [1064]	Picks a Unicode (W) or ASCII (A) version of the NSSizeTypeTryParseVN function.
NTrue [1064]	True value for NBoolean [1004].
NUInt16Parse [1064]	Converts NString [678] to NUInt16 [1009].
NUInt16ParseChars [1065]	Converts an array of NChars to NUInt16 [1009].
NUInt16ParseStrOrChars [1065]	Picks a Unicode (W) or ASCII (A) version of the NUInt16ParseStrOrChars function.
NUInt16ParseVN [1065]	Picks a Unicode (W) or ASCII (A) version of the NUInt16ParseVN function.
NUInt16ToChars [1065]	Picks a Unicode (W) or ASCII (A) version of the NUInt16ToChars function.
NUInt16ToString [1066]	Picks a Unicode (W) or ASCII (A) version of the NUInt16ParseToString function.
NUInt16TryParse [1066]	Converts NString [678] to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt16TryParseChars [1066]	Converts an array of NChars to NUInt16 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt16TryParseStrOrChars [1067]	Picks a Unicode (W) or ASCII (A) version of the NUInt16TryParseStrOrChars function.
NUInt16TryParseVN [1067]	Picks a Unicode (W) or ASCII (A) version of the NUInt16TryParseVN function.
NUInt32Parse [1067]	Converts NString [678] to NUInt32 [1009].
NUInt32ParseChars [1067]	Converts an array of NChars to NUInt32 [1009].
NUInt32ParseStrOrChars [1068]	Picks a Unicode (W) or ASCII (A) version of the NUInt32ParseStrOrChars function.
NUInt32ParseVN [1068]	Picks a Unicode (W) or ASCII (A) version of the NUInt32ParseVN function.
NUInt32ToChars [1068]	Picks a Unicode (W) or ASCII (A) version of the NUInt32ToChars function.
NUInt32ToString [1068]	Picks a Unicode (W) or ASCII (A) version of the NUInt32ToString function.
NUInt32TryParse [1068]	Converts NString [678] to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt32TryParseChars [1069]	Converts an array of NChars to NUInt32 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt32TryParseStrOrChars [1069]	Picks a Unicode (W) or ASCII (A) version of the NUInt32TryParseStrOrChars function.
NUInt32TryParseVN [1069]	Picks a Unicode (W) or ASCII (A) version of the NUInt32TryParseVN function.
NUInt64Parse [1070]	Converts NString [678] to NUInt64 [1009].
NUInt64ParseChars [1070]	Converts an array of NChars to NUInt64 [1009].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt64ParseStrOrChars [1070]	Picks a Unicode (W) or ASCII (A) version of the NUInt64ParseStrOrChars function.
NUInt64ParseVN [1071]	Picks a Unicode (W) or ASCII (A) version of the NUInt64ParseVN function.
NUInt64ToChars [1071]	Picks a Unicode (W) or ASCII (A) version of the NUInt64ToChars function.
NUInt64ToString [1071]	Picks a Unicode (W) or ASCII (A) version of the NUInt64ToString function.
NUInt64TryParse [1071]	Converts NString [678] to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt64TryParseChars [1071]	Converts an array of NChars to NUInt64 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt64TryParseStrOrChars [1072]	Picks a Unicode (W) or ASCII (A) version of the NUInt64TryParseStrOrChars function.
NUInt64TryParseVN [1072]	Picks a Unicode (W) or ASCII (A) version of the NUInt64TryParseVN function.
NUInt8Parse [1072]	Converts NString [678] to NUInt8 [1009].
NUInt8ParseChars [1073]	Converts an array of NChars to NUInt8 [1009].
NUInt8ParseStrOrChars [1073]	Picks a Unicode (W) or ASCII (A) version of the NUInt8ParseStrOrChars function.
NUInt8ParseVN [1073]	Picks a Unicode (W) or ASCII (A) version of the NUInt8ParseVN function.
NUInt8ToChars [1073]	Picks a Unicode (W) or ASCII (A) version of the NUInt8ToChars function.
NUInt8ToString [1074]	Picks a Unicode (W) or ASCII (A) version of the NUInt8ToString function.
NUInt8TryParse [1074]	Converts NString [678] to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt8TryParseChars [1074]	Converts an array of NChars to NUInt8 [1009]. A NBool [1004] value indicates whether the conversion succeeded.
NUInt8TryParseStrOrChars [1075]	Picks a Unicode (W) or ASCII (A) version of the NUInt8TryParseStrOrChars function.
NUInt8TryParseVN [1075]	Picks a Unicode (W) or ASCII (A) version of the NUInt8TryParseVN function.
NULL [1075]	Null value for pointer.
NURationalToString [1075]	Picks a Unicode (W) or ASCII (A) version of the NURationalToString function.
NVersionRangeToString [1075]	Picks a Unicode (W) or ASCII (A) version of the NVersionRangeToString function.
NVersionToString [1075]	Picks a Unicode (W) or ASCII (A) version of the NVersionToString function.
NWCharParse [1076]	Converts NString [678] to NWChar.
NWCharParseChars [1076]	Converts an array of NChars to NWChar.
NWCharParseStrOrChars [1076]	Picks a Unicode (W) or ASCII (A) version of the NWCharParseStrOrChars function.
NWCharParseVN [1077]	Picks a Unicode (W) or ASCII (A) version of the NWCharParseVN function.
NWCharToChars [1077]	Picks a Unicode (W) or ASCII (A) version of the NWCharToChars function.
NWCharToString [1077]	Picks a Unicode (W) or ASCII (A) version of the NWCharToString function.
NWCharTryParse [1077]	Converts NString [678] to NWChar. A NBool [1004] value indicates whether the conversion succeeded.
NWCharTryParseChars [1077]	Converts an array of NChars to NWChar. A NBool [1004] value indicates whether the conversion succeeded.
NWCharTryParseStrOrChars [1078]	Picks a Unicode (W) or ASCII (A) version of the NWCharTryParseStrOrChars function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NWCharTryParseVN [1078]	Picks a Unicode (W) or ASCII (A) version of the NWCharTryParseVN function.

Module

NTypes Unit [905]

Structures

	Name	Description
	NComplex_ [1001]	Represents a complex number value.
	NIndexPair_ [1001]	Represents a pair of indexes.
	NNameStringPair_ [1001]	Represents a pair of name strings.
	NRational_ [1002]	Represents a signed rational number.
	NURational_ [1002]	Represents an unsigned rational number.

Types

Name	Description
NAChar [1004]	ANSI character (8-bit).
NBool [1004]	Same as NBoolean [1004].
NBoolean [1004]	32-bit boolean value. See also NTrue [1064] and NFalse [1041].
NByte [1005]	Same as NUInt8 [1009].
NChar [1005]	ANSI character (8-bit). See also NAChar [1004].
NComplex [1005]	Complex number.
NDouble [1005]	Double precision floating point number.
NFloat [1005]	Same as NSingle [1008].
NHandle [1006]	Pointer to unspecified data (same as void *).
NIndexPair [1006]	Represents a pair of indexes.
NInt [1006]	Same as NInt32 [1006].
NInt16 [1006]	16-bit signed integer (short).
NInt32 [1006]	32-bit signed integer (int).
NInt64 [1006]	64-bit signed integer (long). Not available on some 32-bit platforms.
NInt8 [1007]	8-bit signed integer (signed byte).
NLong [1007]	Same as NInt64 [1006].
NNameStringPair [1007]	Two linked strings.
NRational [1007]	Represents a signed rational number.
NResult [1007]	Result of a function (same as NInt [1006]). See also NErrors module.
NSByte [1008]	Same as NInt8 [1007].
NShort [1008]	Same as NInt16 [1006].
NSingle [1008]	Single precision floating point number.
NSizeType [1008]	Platform dependent size type. Unsigned 32-bit integer on 32-bit platform, unsigned 64-bit integer on 64-bit platform.
NSSizeType [1009]	Platform dependent size type. Signed 32-bit integer on 32-bit platform, signed 64-bit integer on 64-bit platform.
NUInt [1009]	Same as NUInt32 [1009].
NUInt16 [1009]	16-bit unsigned integer (unsigned short).
NUInt32 [1009]	32-bit unsigned integer (unsigned int).
NUInt64 [1009]	64-bit unsigned integer (unsigned long). Not available on some 32-bit platforms.
NUInt8 [1009]	8-bit unsigned integer (byte).
NULong [1010]	Same as NUInt64 [1009].
NURational [1010]	Represents an unsigned rational number.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUShort [1010]	Same as NUInt16 [1009].
NVersion [1010]	Two NInt32 [1006] values defining a version.
NVersion_ [1010]	A NUShort [1010] typedef defining a version.
NVersionRange [1011]	Four NInt32 [1006] values defining a version range.
NVersionRange_ [1011]	A NUInt [1009] typedef defining a version range.

7.1.2.65 NValue Unit

Provides functionality for management and conversion of values of different types.

Files

Name	Description
NValue.h [1121]	Header file for the NValue unit.

Functions

	Name	Description
	NKeyValuePairCopy [1096]	Creates a copy of the specified key value pair structure.
	NKeyValuePairCreate [1096]	Creates a new key value pair structure.
	NKeyValuePairDispose [1096]	Disposes of the specified key value pair structure.
	NKeyValuePairSet [1097]	Sets the value of the specified key value pair structure.
	NNameValuePairCopy [1097]	Creates a copy of the specified name value pair structure.
	NNameValuePairCreate [1097]	Creates a new name value pair structure.
	NNameValuePairDispose [1098]	Disposes of the specified name value pair structure.
	NNameValuePairSet [1098]	Sets the value of the specified name value pair structure.
	NValueChangeType [1098]	Changes the type of the specified value to the new type.
	NValueChangeTypeP [1099]	Changes the type of the specified value to the new type.
	NValueCreate [1099]	Creates a new value of the specified type.
	NValueCreateCustom [1100]	Creates a new value with specified callbacks.
	NValueCreateCustomN [1100]	Creates a new value with specified callbacks.
	NValueCreateFromBoolean [1101]	Creates a new value from the specified Boolean value.
	NValueCreateFromByte [1101]	Creates a new value from the specified byte.
	NValueCreateFromChar [1101]	Creates a new value from the specified char.
	NValueCreateFromDouble [1102]	Creates a new value from the specified double value.
	NValueCreateFromInt16 [1102]	Creates a new value from the specified short integer.
	NValueCreateFromInt32 [1102]	Creates a new value from the specified integer.
	NValueCreateFromInt64 [1103]	Creates a new value from the specified long integer.
	NValueCreateFromObject [1103]	Creates a new value from the specified object.
	NValueCreateFromObjectP [1104]	Creates a new value from the specified object.
	NValueCreateFromPointer [1104]	Creates a new value from the specified pointer.
	NValueCreateFromResult [1104]	Creates a new value from the specified NResult [1007] value.
	NValueCreateFromSByte [1105]	Creates a new value from the specified signed byte.
	NValueCreateFromSingle [1105]	Creates a new value from the specified floating point value.
	NValueCreateFromSizeType [1105]	Creates a new value from the specified NSizeType [1008] value.
	NValueCreateFromSSizeType [1106]	Creates a new value from the specified NSSizeType [1009] value.
	NValueCreateFromString [1106]	Creates a new value from the specified string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NValueCreateFromUInt16 [1106]	Creates a new value from the specified unsigned short integer.
	NValueCreateFromUInt32 [1107]	Creates a new value from the specified unsigned integer.
	NValueCreateFromUInt64 [1107]	Creates a new value from the specified unsigned long integer.
	NValueCreateP [1108]	Creates a new value of the specified type.
	NValueGetAttributes [1108]	Retrieves the attributes of the specified value.
	NValueGetFree [1108]	Frees the specified value.
	NValueGetPtr [1109]	Retrieves a pointer to the specified value.
	NValueGetSize [1109]	Retrieves the size of the specified value.
	NValueGetTypeCode [1109]	Retrieves the type code of the specified value.
	NValueGetValueType [1110]	Retrieves the type of the specified value.
	NValueSet [1110]	Sets the specified value.
	NValueSetP [1110]	Sets the specified value.
	NValueToBoolean [1111]	Converts the specified value into a Boolean value.
	NValueToByte [1111]	Converts the specified value into a byte.
	NValueToChar [1112]	Converts the specified value into a char.
	NValueToDouble [1112]	Converts the specified value into a double value.
	NValueToInt16 [1112]	Converts the specified value into a short integer.
	NValueToInt32 [1113]	Converts the specified value to into integer.
	NValueToInt64 [1113]	Converts the specified value into a long value.
	NValueToObject [1114]	Converts the specified value into an object.
	NValueToObjectP [1114]	Converts the specified value into an object.
	NValueToPointer [1114]	Converts the specified value into a pointer.
	NValueToResult [1115]	Converts the specified value into a NResult [1007] value.
	NValueToSByte [1115]	Converts the specified value into a signed byte.
	NValueToSingle [1115]	Converts the specified value into a floating point value.
	NValueToSizeType [1116]	Converts the specified value into a NSizeType [1008] value.
	NValueToSSizeType [1116]	Converts the specified value into a NSSizeType [1009] value.
	NValueToString [1117]	Converts the specified value into a string.
	NValueToUInt16 [1117]	Converts the specified value into an unsigned short integer.
	NValueToUInt32 [1117]	Converts the specified value into an unsigned integer.
	NValueToUInt64 [1118]	Converts the specified value into an unsigned long integer.
	NValueToValue [1118]	Converts the specified value into a value of the specified type.
	NValueToValueP [1119]	Converts the specified value into a value of the specified type.

Macros

Name	Description
NNameValuePairCreate [1120]	Picks a Unicode (W) or ASCII (A) version of the NNameValuePairCreate function.
NValueCreateFromChar [1120]	Picks a Unicode (W) or ASCII (A) version of the NValueCreateFromChar function.
NValueCreateFromString [1120]	Picks a Unicode (W) or ASCII (A) version of the NValueCreateFromString function.
NValueToChar [1120]	Picks a Unicode (W) or ASCII (A) version of the NValueToChar function.

Structs, Records, Enums

	Name	Description
	NKeyValuePair_ [1119]	Represents the key value pair structure.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NNameValuePair_ [1119]	Represents the name key value pair structure.

7.1.2.65.1 Functions

7.1.2.65.1.1 NKeyValuePairCopy Function

Creates a copy of the specified key value pair structure.

C++

```
NResult N_API NKeyValuePairCopy(const struct NKeyValuePair_ * pSrcValue, struct
NKeyValuePair_ * pDstValue);
```

Parameters

Parameters	Description
const struct NKeyValuePair_ * pSrcValue	[in] Pointer to the original key value pair.
struct NKeyValuePair_ * pDstValue	[out] Pointer to the copied key value pair.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.2 NKeyValuePairCreate Function

Creates a new key value pair structure.

C++

```
NResult N_API NKeyValuePairCreate(HNValue hKey, HNValue hValue, struct NKeyValuePair_ *
pValue);
```

Parameters

Parameters	Description
HNValue hKey	[in] Handle to the key part of the pair.
HNValue hValue	[in] Handle to the value part of the pair.
struct NKeyValuePair_ * pValue	[out] Pointer to the new key value pair.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.3 NKeyValuePairDispose Function

Disposes of the specified key value pair structure.

C++

```
NResult N_API NKeyValuePairDispose(struct NKeyValuePair_ * pValue);
```

Parameters

Parameters	Description
struct NKeyValuePair_ * pValue	[in/out] Pointer to the key value pair to be disposed of.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.4 NKeyValuePairSet Function

Sets the value of the specified key value pair structure.

C++

```
NResult N_API NKeyValuePairSet(const struct NKeyValuePair_ * pSrcValue, struct
NKeyValuePair_ * pDstValue);
```

Parameters

Parameters	Description
const struct NKeyValuePair_ * pSrcValue	[in] Pointer to the new key value pair value.
struct NKeyValuePair_ * pDstValue	[in] Pointer to the key value pair to be changed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.5 NNameValuePairCopy Function

Creates a copy of the specified name value pair structure.

C++

```
NResult N_API NNameValuePairCopy(const struct NNameValuePair_ * pSrcValue, struct
NNameValuePair_ * pDstValue);
```

Parameters

Parameters	Description
const struct NNameValuePair_ * pSrcValue	[in] Pointer to the original name value pair.
struct NNameValuePair_ * pDstValue	[out] Pointer to the copied name value pair.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.6 NNameValuePairCreate Function

Creates a new name value pair structure.

C++

```
NResult N_API NNameValuePairCreate(const NChar * szKey, HNValue hValue, NNameValuePair *
pValue);
```

Parameters

Parameters	Description
const NChar * szKey	[in] Pointer to the key part of the pair.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNValue hValue	[in] Handle to the value part of the pair.
NNameValuePair * pValue	[out] Pointer to the new name value pair.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.7 NNameValuePairDispose Function

Disposes of the specified name value pair structure.

C++

```
NResult N_API NNameValuePairDispose(struct NNameValuePair_ * pValue);
```

Parameters

Parameters	Description
struct NNameValuePair_ * pValue	[in/out] Pointer to the name value pair to be disposed of.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.8 NNameValuePairSet Function

Sets the value of the specified name value pair structure.

C++

```
NResult N_API NNameValuePairSet(const struct NNameValuePair_ * pSrcValue, struct
NNameValuePair_ * pDstValue);
```

Parameters

Parameters	Description
const struct NNameValuePair_ * pSrcValue	[in] Pointer to the new name value pair value.
struct NNameValuePair_ * pDstValue	[in] Pointer to the name value pair to be changed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.9 NValueChangeType Function

Changes the type of the specified value to the new type.

C++

```
NResult N_API NValueChangeType(HNValue hSrcValue, HNType hType, NAttributes attributes,
HNString hFormat, HNValue * phValue);
```

Parameters

Parameters	Description
HNValue hSrcValue	[in] Handle to the original value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNType hType	[in] Handle to the new value type.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.10 NValueChangeTypeP Function

Changes the type of the specified value to the new type.

C++

```
NResult N_API NValueChangeTypeP(HNValue hSrcValue, NTypeOfProc pTypeOf, NAttributes
attributes, HNString hFormat, HNValue * phValue);
```

Parameters

Parameters	Description
HNValue hSrcValue	[in] Handle to the original value.
NTypeOfProc pTypeOf	[in] Pointer to a function that retrieves the new value type.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.11 NValueCreate Function

Creates a new value of the specified type.

C++

```
NResult N_API NValueCreate(HNType hType, const void * pValue, NSizeType valueSize, NBool
hasValue, NAttributes attributes, HNValue * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type of the value.
const void * pValue	[in] Pointer to the value of the new value.
NSizeType valueSize	[in] Size of the new value.
NBool hasValue	[in] NTrue [1064] if *pValue [870]* contains a value, else NFalse [1041].
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NValue Unit [1094]

7.1.2.65.1.12 NValueCreateCustom Function

Creates a new value with specified callbacks.

C++

```
NResult N_API NValueCreateCustom(const void * value, NPointerFreeProc pFree, void *
pFreeParam, NPointerGetHashCodeProc pGetHashCode, void * pGetHashCodeParam,
NPointerEqualsProc pEquals, void * pEqualsParam, NAttributes attributes, HNValue * phValue);
```

Parameters

Parameters	Description
const void * value	[in] Pointer to the value of the new value.
NPointerFreeProc pFree	[in] Pointer to the "Free" function for the new value.
void * pFreeParam	[in] Pointer to the parameters for the "Free" function for the new value.
NPointerGetHashCodeProc pGetHashCode	[in] Pointer to the "Get Hash Code" function for the new value.
void * pGetHashCodeParam	[in] Pointer to the parameters for the "Get Hash Code" function for the new value.
NPointerEqualsProc pEquals	[in] Pointer to the "Equals" function for the new value.
void * pEqualsParam	[in] Pointer to the parameters for the "Equals" function for the new value.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.13 NValueCreateCustomN Function

Creates a new value with specified callbacks.

C++

```
NResult N_API NValueCreateCustomN(const void * value, HNCallback hFree, HNCallback
hGetHashCode, HNCallback hEquals, NAttributes attributes, HNValue * phValue);
```

Parameters

Parameters	Description
const void * value	[in] Pointer to the value of the new value.
HNCallback hFree	[in] Handle to the "Free" function callback for the new value.
HNCallback hGetHashCode	[in] Handle to the "Get Hash Code" function callback for the new value.
HNCallback hEquals	[in] Handle to the "Equals" function callback for the new value.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.14 NValueCreateFromBoolean Function

Creates a new value from the specified Boolean value.

C++

```
NResult N_API NValueCreateFromBoolean(NBool value, NAttributes attributes, HNValue *
phValue);
```

Parameters

Parameters	Description
NBool value	[in] The original Boolean value.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.15 NValueCreateFromByte Function

Creates a new value from the specified byte.

C++

```
NResult N_API NValueCreateFromByte(NByte value, NAttributes attributes, HNValue * phValue);
```

Parameters

Parameters	Description
NByte value	[in] The original byte.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.16 NValueCreateFromChar Function

Creates a new value from the specified char.

C++

```
NResult N_API NValueCreateFromChar(NChar value, NAttributes attributes, HNValue * phValue);
```

Parameters

Parameters	Description
NChar value	[in] The original char.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.17 NValueCreateFromDouble Function

Creates a new value from the specified double value.

C++

```
NResult N_API NValueCreateFromDouble(NDouble value, NAttributes attributes, HNValue *
phValue);
```

Parameters

Parameters	Description
NDouble value	[in] The original double value.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.18 NValueCreateFromInt16 Function

Creates a new value from the specified short integer.

C++

```
NResult N_API NValueCreateFromInt16(NShort value, NAttributes attributes, HNValue *
phValue);
```

Parameters

Parameters	Description
NShort value	[in] The original short integer.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.19 NValueCreateFromInt32 Function

Creates a new value from the specified integer.

C++

```
NResult N_API NValueCreateFromInt32(NInt value, NAttributes attributes, HNValue * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NInt value	[in] The original integer.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.20 NValueCreateFromInt64 Function

Creates a new value from the specified long integer.

C++

```
NResult N_API NValueCreateFromInt64(NLong value, NAttributes attributes, HNValue * phValue);
```

Parameters

Parameters	Description
NLong value	[in] The original long integer.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.21 NValueCreateFromObject Function

Creates a new value from the specified object.

C++

```
NResult N_API NValueCreateFromObject(HNType hType, HNObject hValue, NAttributes attributes,
HNValue * phValue);
```

Parameters

Parameters	Description
HNType hType	[in] Handle to the type of the object.
HNObject hValue	[in] The original object.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.65.1.22 NValueCreateFromObjectP Function

Creates a new value from the specified object.

C++

```
NResult N_API NValueCreateFromObjectP(NTypeOfProc pTypeOf, HNObject hValue, NAttributes attributes, HNValue * phValue);
```

Parameters

Parameters	Description
NTypeOfProc pTypeOf	[in] Pointer to a function that retrieves the type of the object.
HNObject hValue	[in] The original object.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.23 NValueCreateFromPointer Function

Creates a new value from the specified pointer.

C++

```
NResult N_API NValueCreateFromPointer(const void * value, NAttributes attributes, HNValue * phValue);
```

Parameters

Parameters	Description
const void * value	[in] The original pointer.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.24 NValueCreateFromResult Function

Creates a new value from the specified NResult [1007] value.

C++

```
NResult N_API NValueCreateFromResult(NResult value, NAttributes attributes, HNValue * phValue);
```

Parameters

Parameters	Description
NResult value	[in] The original NResult [1007] value.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.25 NValueCreateFromSByte Function

Creates a new value from the specified signed byte.

C++

```
NResult N_API NValueCreateFromSByte(NSByte value, NAttributes attributes, HNValue *
phValue);
```

Parameters

Parameters	Description
NSByte value	[in] The original signed byte.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.26 NValueCreateFromSingle Function

Creates a new value from the specified floating point value.

C++

```
NResult N_API NValueCreateFromSingle(NFloat value, NAttributes attributes, HNValue *
phValue);
```

Parameters

Parameters	Description
NFloat value	[in] The original floating point value.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.27 NValueCreateFromSizeType Function

Creates a new value from the specified NSizeType [1008] value.

C++

```
NResult N_API NValueCreateFromSizeType(NSizeType value, NAttributes attributes, HNValue *
phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NSizeType value	[in] The original NSizeType [1008] value.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.28 NValueCreateFromSSizeType Function

Creates a new value from the specified NSSizeType [1009] value.

C++

```
NResult N_API NValueCreateFromSSizeType(NSSizeType value, NAttributes attributes, HNValue *
phValue);
```

Parameters

Parameters	Description
NSSizeType value	[in] The original NSSizeType [1009] value.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.29 NValueCreateFromString Function

Creates a new value from the specified string.

C++

```
NResult N_API NValueCreateFromString(const NChar * szValue, NAttributes attributes, HNValue
* phValue);
```

Parameters

Parameters	Description
const NChar * szValue	[in] The original string.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.30 NValueCreateFromUInt16 Function

Creates a new value from the specified unsigned short integer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NValueCreateFromUInt16(NUShort value, NAttributes attributes, HNValue *
phValue);
```

Parameters

Parameters	Description
NUShort value	[in] The original unsigned short integer.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.31 NValueCreateFromUInt32 Function

Creates a new value from the specified unsigned integer.

C++

```
NResult N_API NValueCreateFromUInt32(NUInt value, NAttributes attributes, HNValue *
phValue);
```

Parameters

Parameters	Description
NUInt value	[in] The original unsigned integer.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.32 NValueCreateFromUInt64 Function

Creates a new value from the specified unsigned long integer.

C++

```
NResult N_API NValueCreateFromUInt64(NULong value, NAttributes attributes, HNValue *
phValue);
```

Parameters

Parameters	Description
NULong value	[in] The original unsigned long integer.
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.65.1.33 NValueCreateP Function

Creates a new value of the specified type.

C++

```
NResult N_API NValueCreateP(NTypeOfProc pTypeOf, const void * pValue, NSizeType valueSize,
NBool hasValue, NAttributes attributes, HNValue * phValue);
```

Parameters

Parameters	Description
NTypeOfProc pTypeOf	[in] Pointer to a function that retrieves the value type.
const void * pValue	[in] Pointer to the value of the new value.
NSizeType valueSize	[in] Size of the new value.
NBool hasValue	[in] NTrue [1064] if *pValue [870]* contains a value, else NFalse [1041].
NAttributes attributes	[in] Value attributes.
HNValue * phValue	[out] Pointer to a handle to the new value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.34 NValueGetAttributes Function

Retrieves the attributes of the specified value.

C++

```
NResult N_API NValueGetAttributes(HNValue hValue, NAttributes * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the value.
NAttributes * pResult	[out] Pointer to the attributes of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.35 NValueGetFree Function

Frees the specified value.

C++

```
NResult N_API NValueGetFree(HNValue hValue, HNCallback * phResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the value to be freed.
HNCallback * phResult	[out] Pointer to a handle to the callback for the "Free" function for the value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.36 NValueGetPtr Function

Retrieves a pointer to the specified value.

C++

```
NResult N_API NValueGetPtr(HNValue hValue, const void * * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the value.
const void * * pResult	[out] Pointer to the pointer to the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.37 NValueGetSize Function

Retrieves the size of the specified value.

C++

```
NResult N_API NValueGetSize(HNValue hValue, NSizeType * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the value.
NSizeType * pResult	[out] Pointer to the size of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.38 NValueGetTypeCode Function

Retrieves the type code of the specified value.

C++

```
NResult N_API NValueGetTypeCode(HNValue hValue, NTypeCode * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the value.
NTypeCode * pResult	[out] Pointer to the type code of the value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.39 NValueGetValueType Function

Retrieves the type of the specified value.

C++

```
NResult N_API NValueGetValueType(HNValue hValue, HNType * phResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the value.
HNType * phResult	[out] Pointer to a handle to the type of the value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.40 NValueSet Function

Sets the specified value.

C++

```
NResult N_API NValueSet(HNValue hValue, HNType hType, NAttributes attributes, void *
pValue, NSizeType valueSize, NBool * pHasValue);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the value to be set.
HNType hType	[in] Handle to the type of the value.
NAttributes attributes	[in] Value attributes.
void * pValue	[in] Pointer to the new value.
NSizeType valueSize	[in] Size of the value.
NBool * pHasValue	[out] NTrue [1064] if *pValue [870]* contains a value; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.41 NValueSetP Function

Sets the specified value.

C++

```
NResult N_API NValueSetP(HNValue hValue, NTypeOfProc pTypeOf, NAttributes attributes, void
* pValue, NSizeType valueSize, NBool * pHasValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the value to be set.
NTypeOfProc pTypeOf	[in] Pointer to a function that retrieves the type of the value.
NAttributes attributes	[in] Value attributes.
void * pValue	[in] Pointer to the new value.
NSizeType valueSize	[in] Size of the value.
NBool * pHasValue	[out] NTrue [1064] if *pValue [870]* contains a value; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.42 NValueToBoolean Function

Converts the specified value into a Boolean value.

C++

```
NResult N_API NValueToBoolean(HNValue hValue, NAttributes attributes, HNString hFormat,
NBool * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NBool * pResult	[out] Pointer to the result Boolean value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.43 NValueToByte Function

Converts the specified value into a byte.

C++

```
NResult N_API NValueToByte(HNValue hValue, NAttributes attributes, HNString hFormat, NByte
* pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NByte * pResult	[out] Pointer to the result byte.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NValue Unit [1094]

7.1.2.65.1.44 NValueToChar Function

Converts the specified value into a char.

C++

```
NResult N_API NValueToChar(HNValue hValue, NAttributes attributes, HNString hFormat, NChar
* pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NChar * pResult	[out] Pointer to the result char.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.45 NValueToDouble Function

Converts the specified value into a double value.

C++

```
NResult N_API NValueToDouble(HNValue hValue, NAttributes attributes, HNString hFormat,
NDouble * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NDouble * pResult	[out] Pointer to the result double value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.46 NValueToInt16 Function

Converts the specified value into a short integer.

C++

```
NResult N_API NValueToInt16(HNValue hValue, NAttributes attributes, HNString hFormat,
NShort * pResult);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NShort * pResult	[out] Pointer to the result short integer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.47 NValueToInt32 Function

Converts the specified value to into integer.

C++

```
NResult N_API NValueToInt32(HNValue hValue, NAttributes attributes, HNString hFormat, NInt
* pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NInt * pResult	[out] Pointer to the result integer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.48 NValueToInt64 Function

Converts the specified value into a long value.

C++

```
NResult N_API NValueToInt64(HNValue hValue, NAttributes attributes, HNString hFormat, NLong
* pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NLong * pResult	[out] Pointer to the result long integer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.65.1.49 NValueToObject Function

Converts the specified value into an object.

C++

```
NResult N_API NValueToObject(HNValue hValue, HNType hType, NAttributes attributes, HNString hFormat, HNObject * phResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
HNType hType	[in] Handle to the type of the object.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
HNObject * phResult	[out] Pointer to the result object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.50 NValueToObjectP Function

Converts the specified value into an object.

C++

```
NResult N_API NValueToObjectP(HNValue hValue, NTypeOfProc pTypeOf, NAttributes attributes, HNString hFormat, HNObject * phResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NTypeOfProc pTypeOf	[in] Pointer to a function that retrieves the type of the object.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
HNObject * phResult	[out] Pointer to the result object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.51 NValueToPointer Function

Converts the specified value into a pointer.

C++

```
NResult N_API NValueToPointer(HNValue hValue, NAttributes attributes, HNString hFormat, void * * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
void * * pResult	[out] Pointer to the result pointer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.52 NValueToResult Function

Converts the specified value into a NResult [1007] value.

C++

```
NResult N_API NValueToResult(HNValue hValue, NAttributes attributes, HNString hFormat,
NResult * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NResult * pResult	[out] Pointer to the result NResult [1007] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.53 NValueToSByte Function

Converts the specified value into a signed byte.

C++

```
NResult N_API NValueToSByte(HNValue hValue, NAttributes attributes, HNString hFormat,
NSByte * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NSByte * pResult	[out] Pointer to the result signed byte.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.54 NValueToSingle Function

Converts the specified value into a floating point value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NValueToSingle(HNValue hValue, NAttributes attributes, HNString hFormat,
NFloat * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NFloat * pResult	[out] Pointer to the result floating point value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.55 NValueToSizeType Function

Converts the specified value into a NSizeType [1008] value.

C++

```
NResult N_API NValueToSizeType(HNValue hValue, NAttributes attributes, HNString hFormat,
NSizeType * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NSizeType * pResult	[out] Pointer to the result NSizeType [1008] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.56 NValueToSSizeType Function

Converts the specified value into a NSSizeType [1009] value.

C++

```
NResult N_API NValueToSSizeType(HNValue hValue, NAttributes attributes, HNString hFormat,
NSSizeType * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NSSizeType * pResult	[out] Pointer to the result NSSizeType [1009] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NValue Unit [1094]

7.1.2.65.1.57 NValueToString Function

Converts the specified value into a string.

C++

```
NResult N_API NValueToString(HNValue hValue, NAttributes attributes, HNString hFormat,
HNString * phResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
HNString * phResult	[out] Pointer to the result string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.58 NValueToUInt16 Function

Converts the specified value into an unsigned short integer.

C++

```
NResult N_API NValueToUInt16(HNValue hValue, NAttributes attributes, HNString hFormat,
NUShort * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NUShort * pResult	[out] Pointer to the result unsigned short integer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.59 NValueToUInt32 Function

Converts the specified value into an unsigned integer.

C++

```
NResult N_API NValueToUInt32(HNValue hValue, NAttributes attributes, HNString hFormat,
NUInt * pResult);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NUInt * pResult	[out] Pointer to the result unsigned integer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.60 NValueToUInt64 Function

Converts the specified value into an unsigned long integer.

C++

```
NResult N_API NValueToUInt64(HNValue hValue, NAttributes attributes, HNString hFormat,
NULong * pResult);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
NULong * pResult	[out] Pointer to the result unsigned long integer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.1.61 NValueToValue Function

Converts the specified value into a value of the specified type.

C++

```
NResult N_API NValueToValue(HNValue hValue, HNType hType, NAttributes attributes, HNString
hFormat, void * pResult, NSizeType resultSize);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
HNType hType	[in] Handle to the type of the value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
void * pResult	[out] Pointer to the result value.
NSizeType resultSize	[in] Size of the resulting value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NValue Unit [1094]

7.1.2.65.1.62 NValueToValueP Function

Converts the specified value into a value of the specified type.

C++

```
NResult N_API NValueToValueP(HNValue hValue, NTypeOfProc pTypeOf, NAttributes attributes,
HNString hFormat, void * pResult, NSizeType resultSize);
```

Parameters

Parameters	Description
HNValue hValue	[in] Handle to the original value.
NTypeOfProc pTypeOf	[in] Pointer to a function that retrieves the type of the value.
NAttributes attributes	[in] Value attributes.
HNString hFormat	[in] Handle to the format string.
void * pResult	[out] Pointer to the result value.
NSizeType resultSize	[in] Size of the resulting value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NValue Unit [1094]

7.1.2.65.2 Structs, Records, Enums

7.1.2.65.2.1 NKeyValuePair_ Structure

Represents the key value pair structure.

C++

```
struct NKeyValuePair_ {
  HNValue hKey;
  HNValue hValue;
};
```

Members

Members	Description
HNValue hKey;	Handle to the key part of the pair.
HNValue hValue;	Handle to the value part of the pair.

Module

NValue Unit [1094]

7.1.2.65.2.2 NNameValuePair_ Structure

Represents the name key value pair structure.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
struct NNameValuePair_ {
  HNString hKey;
  HNValue hValue;
};
```

Members

Members	Description
HNString hKey;	Handle to the key part of the pair.
HNValue hValue;	Handle to the value part of the pair.

Module

NValue Unit [1094]

7.1.2.65.3 Macros

7.1.2.65.3.1 NNameValuePairCreate Macro

Picks a Unicode (W) or ASCII (A) version of the NNameValuePairCreate function.

C++

```
#define NNameValuePairCreate N_FUNC_AW(NNameValuePairCreate)
```

Module

NValue Unit [1094]

7.1.2.65.3.2 NValueCreateFromChar Macro

Picks a Unicode (W) or ASCII (A) version of the NValueCreateFromChar function.

C++

```
#define NValueCreateFromChar N_FUNC_AW(NValueCreateFromChar)
```

Module

NValue Unit [1094]

7.1.2.65.3.3 NValueCreateFromString Macro

Picks a Unicode (W) or ASCII (A) version of the NValueCreateFromString function.

C++

```
#define NValueCreateFromString N_FUNC_AW(NValueCreateFromString)
```

Module

NValue Unit [1094]

7.1.2.65.3.4 NValueToChar Macro

Picks a Unicode (W) or ASCII (A) version of the NValueToChar function.

C++

```
#define NValueToChar N_FUNC_AW(NValueToChar)
```

Module

NValue Unit [1094]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.65.4 Files

7.1.2.65.4.1 NValue.h

Header file for the NValue [1094] unit.

Functions

	Name	Description
	NKeyValuePairCopy [1096]	Creates a copy of the specified key value pair structure.
	NKeyValuePairCreate [1096]	Creates a new key value pair structure.
	NKeyValuePairDispose [1096]	Disposes of the specified key value pair structure.
	NKeyValuePairSet [1097]	Sets the value of the specified key value pair structure.
	NNameValuePairCopy [1097]	Creates a copy of the specified name value pair structure.
	NNameValuePairCreate [1097]	Creates a new name value pair structure.
	NNameValuePairDispose [1098]	Disposes of the specified name value pair structure.
	NNameValuePairSet [1098]	Sets the value of the specified name value pair structure.
	NValueChangeType [1098]	Changes the type of the specified value to the new type.
	NValueChangeTypeP [1099]	Changes the type of the specified value to the new type.
	NValueCreate [1099]	Creates a new value of the specified type.
	NValueCreateCustom [1100]	Creates a new value with specified callbacks.
	NValueCreateCustomN [1100]	Creates a new value with specified callbacks.
	NValueCreateFromBoolean [1101]	Creates a new value from the specified Boolean value.
	NValueCreateFromByte [1101]	Creates a new value from the specified byte.
	NValueCreateFromChar [1101]	Creates a new value from the specified char.
	NValueCreateFromDouble [1102]	Creates a new value from the specified double value.
	NValueCreateFromInt16 [1102]	Creates a new value from the specified short integer.
	NValueCreateFromInt32 [1102]	Creates a new value from the specified integer.
	NValueCreateFromInt64 [1103]	Creates a new value from the specified long integer.
	NValueCreateFromObject [1103]	Creates a new value from the specified object.
	NValueCreateFromObjectP [1104]	Creates a new value from the specified object.
	NValueCreateFromPointer [1104]	Creates a new value from the specified pointer.
	NValueCreateFromResult [1104]	Creates a new value from the specified NResult [1007] value.
	NValueCreateFromSByte [1105]	Creates a new value from the specified signed byte.
	NValueCreateFromSingle [1105]	Creates a new value from the specified floating point value.
	NValueCreateFromSizeType [1105]	Creates a new value from the specified NSizeType [1008] value.
	NValueCreateFromSSizeType [1106]	Creates a new value from the specified NSSizeType [1009] value.
	NValueCreateFromString [1106]	Creates a new value from the specified string.
	NValueCreateFromUInt16 [1106]	Creates a new value from the specified unsigned short integer.
	NValueCreateFromUInt32 [1107]	Creates a new value from the specified unsigned integer.
	NValueCreateFromUInt64 [1107]	Creates a new value from the specified unsigned long integer.
	NValueCreateP [1108]	Creates a new value of the specified type.
	NValueGetAttributes [1108]	Retrieves the attributes of the specified value.
	NValueGetFree [1108]	Frees the specified value.
	NValueGetPtr [1109]	Retrieves a pointer to the specified value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NValueGetSize [1109]	Retrieves the size of the specified value.
	NValueGetTypeCode [1109]	Retrieves the type code of the specified value.
	NValueGetValueType [1110]	Retrieves the type of the specified value.
	NValueSet [1110]	Sets the specified value.
	NValueSetP [1110]	Sets the specified value.
	NValueToBoolean [1111]	Converts the specified value into a Boolean value.
	NValueToByte [1111]	Converts the specified value into a byte.
	NValueToChar [1112]	Converts the specified value into a char.
	NValueToDouble [1112]	Converts the specified value into a double value.
	NValueToInt16 [1112]	Converts the specified value into a short integer.
	NValueToInt32 [1113]	Converts the specified value to into integer.
	NValueToInt64 [1113]	Converts the specified value into a long value.
	NValueToObject [1114]	Converts the specified value into an object.
	NValueToObjectP [1114]	Converts the specified value into an object.
	NValueToPointer [1114]	Converts the specified value into a pointer.
	NValueToResult [1115]	Converts the specified value into a NResult [1007] value.
	NValueToSByte [1115]	Converts the specified value into a signed byte.
	NValueToSingle [1115]	Converts the specified value into a floating point value.
	NValueToSizeType [1116]	Converts the specified value into a NSizeType [1008] value.
	NValueToSSizeType [1116]	Converts the specified value into a NSSizeType [1009] value.
	NValueToString [1117]	Converts the specified value into a string.
	NValueToUInt16 [1117]	Converts the specified value into an unsigned short integer.
	NValueToUInt32 [1117]	Converts the specified value into an unsigned integer.
	NValueToUInt64 [1118]	Converts the specified value into an unsigned long integer.
	NValueToValue [1118]	Converts the specified value into a value of the specified type.
	NValueToValueP [1119]	Converts the specified value into a value of the specified type.

Macros

Name	Description
NNameValuePairCreate [1120]	Picks a Unicode (W) or ASCII (A) version of the NNameValuePairCreate function.
NValueCreateFromChar [1120]	Picks a Unicode (W) or ASCII (A) version of the NValueCreateFromChar function.
NValueCreateFromString [1120]	Picks a Unicode (W) or ASCII (A) version of the NValueCreateFromString function.
NValueToChar [1120]	Picks a Unicode (W) or ASCII (A) version of the NValueToChar function.

Module

NValue Unit [1094]

Structures

	Name	Description
	NKeyValuePair_ [1119]	Represents the key value pair structure.
	NNameValuePair_ [1119]	Represents the name key value pair structure.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.2.66 NWaitObject Unit

Base unit for common functionality of some cross platform synchronization primitives (supporting timed wait). See derived units for complete description:

- NMutex [534]
- NSemaphore [654]
- NSyncEvent [827]

Functions

	Name	Description
	NWaitObjectWaitFor [1123]	Waits (blocks the calling thread) until the specified object is in signalled state (for example NMutex [534] is not owned by any threads).
	NWaitObjectWaitForTimed [1123]	Waits (blocks the calling thread) until the specified object is in signalled state or timeout expires.

Macros

Name	Description
N_INFINITE [1124]	Constant to be used as infinite time value (for example in NWaitObjectTimedWaitFor function).

7.1.2.66.1 Functions

7.1.2.66.1.1 NWaitObjectWaitFor Function

Waits (blocks the calling thread) until the specified object is in signalled state (for example NMutex [534] is not owned by any threads).

C++

```
NResult N_API NWaitObjectWaitFor(HNWaitObject hWaitObject);
```

Parameters

Parameters	Description
HNWaitObject hWaitObject	[in] Handle to the synchronization object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

The successful operation would also perform necessary updated to the object state:

- NMutex [534] will be locked;
- NSemaphore [654] will be decremented;
- NSyncEvent [827] with auto-reset will be reset.

As a result the object might become nonsignalled. Use the functions form the derived unit to make object signalled again.

Module

NWaitObject Unit [1123]

7.1.2.66.1.2 NWaitObjectWaitForTimed Function

Waits (blocks the calling thread) until the specified object is in signalled state or timeout expires.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NWaitObjectWaitForTimed(HNWaitObject hWaitObject, NInt timeOutMilliseconds,
NBool * pResult);
```

Parameters

Parameters	Description
HNWaitObject hWaitObject	[in] Handle to the synchronization object.
NInt timeOutMilliseconds	[in] Maximum time to wait.
NBool * pResult	[out] Is set to NTrue [1064] on successful operation (the lock is acquired), to NFalse [1041] on timeout.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

The successful operation would also perform necessary updated to the object state. See NWaitObjectWaitFor [1123].

Module

NWaitObject Unit [1123]

7.1.2.66.2 Macros

7.1.2.66.2.1 N_INFINITE Macro

Constant to be used as infinite time value (for example in NWaitObjectTimedWaitFor function).

C++

```
#define N_INFINITE -1
```

Module

NWaitObject Unit [1123]

7.1.2.67 NWindows Unit

7.1.3 NDevices Library

Provides functionality for managing devices, like cameras or fingerprint scanners.

Remarks

Requirements (Windows)

- **Import Library**: NDevices.dll.lib
- **DLL**: *NDevices.dll*

Requirements (Linux)

- **Shared object**: *libNDevices.so*

Notes

The library's NDeviceManager unit is used to enumerate available devices of particular type. The device handle obtained from the manager can than be used with functions from appropriate unit to work with the device and to get the information it

Copyright © 2009-2014 Neurotechnology. All rights reserved.

and it's capabilities.

The devices of the library form type hierarchy, where functionality common to different classes of devices is grouped into separate unit. The diagram below shows the relations between different units:

NDevice [1159] is the main entity in NDevices [1178] library. It provides functions to retrieve parent/child entities, plugins or device name/serial number.

NDevice [1159] entity has two child units: NCaptureDevice [1155] and NBiometricDevice [1142]. NCaptureDevice [1155] is used for capture devices (cameras and microphones) to retrieve device formats, media types and to start/stop capturing. Child entities NMicrophone and NCamera [1145] are used to retrieve sound samples or video frames.

NBiometricDevice [1142] is the main unit of fingerprint and palmprint scanning devices. This unit can be used to retrieve device vendor Id, check spoof detection and to start/stop capturing sequence. NFScanner and NIrisScanner are children of NBiometricDevice [1142]. These entities are used to capture fingerprint and iris images.

Device availability

Device manager keeps track of the devices available in the system, and devices can be connected and disconnected from the computer during runtime. In addition the library uses flexible plugin system. Each particular device is provided to the library by a special plugin (dynamic library). The plugins used can be controlled dynamically through NPluginManager [607] unit (from NCore [257] library). New plugins can also be written by the integrators to add support for custom devices.

Modules

Name	Description
Camera Unit [1126]	Provides functions for working with cameras.
CameraMan Unit [1138]	Manages, enumerates and creates Cameras.
NBiometricDevice Unit [1142]	NBiometricDevice is the main unit of fingerprint and palmprint scanning devices. This unit can be used to retrieve device vendor Id, check spoof detection and to start/stop capturing sequence.
NCamera Unit [1145]	Provides functions for working with cameras.
NCaptureDevice Unit [1155]	Provides functions for working with capture devices.
NDevice Unit [1159]	Provides functionality for retrieving general device information.
NDeviceManager Unit [1168]	Provides functions for device enumeration (listing the devices available in the system) and getting events when device is added or removed.
NDevices Unit [1178]	Provides information about NDevices library.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.1 Camera Unit

Provides functions for working with cameras.

Notes

This module is obsolete. Use NDeviceManager [1168] instead.

Files

Name	Description
Camera.h [1137]	Header file for Camera module. Provides functionality for managing cameras.
CameraParams.h [1137]	Header file for CameraParams module. Provides parameters for camera.

Functions

	Name	Description
	CameraGetCurrentFrame [1127]	Gets current frame from the capture device.
	CameraGetDevice [1127]	Gets the NCamera [1145] device from the specified camera.
	CameraGetId [1128]	Gets associated device identifier.
	CameraGetVideoFormat [1128]	Gets current video format for the capture device.
	CameraGetVideoFormats [1129]	Gets all supported video format for the capture device.
	CameraIsCapturing [1129]	Checks camera status.
	CameraSetVideoFormat [1130]	Sets video format for the capture device.
	CameraStartCapturing [1130]	Starts capturing.
	CameraStopCapturing [1130]	Stops capturing.
	HCamera [1131]	Handle to Camera object.

Macros

Name	Description
CameraCopyParameters [1132]	Copies parameters from one camera to another camera.
CameraGetParameterA [1132]	Gets the specified parameter of the specified camera.
CameraGetParameterW [1132]	Gets the specified parameter of the specified camera.
CAMERAP_AUTOMATIC_SETTINGS [1133]	Identifier of type N_TYPE_BOOL [576] specifying whether automatic camera settings should be used.
CAMERAP_EXPOSURE [1133]	Identifier of type N_TYPE_UINT [578] specifying camera exposure (total amount of light allowed to fall on the camera during taking video frame). The exposure range can be acquired using CAMERAP_EXPOSURE_MIN [1133] and CAMERAP_EXPOSURE_MAX [1133].
CAMERAP_EXPOSURE_MAX [1133]	Read only identifier of type N_TYPE_UINT [578] specifying camera maximum exposure value.
CAMERAP_EXPOSURE_MIN [1133]	Read only identifier of type N_TYPE_UINT [578] specifying camera's minimum exposure value.
CAMERAP_GAIN [1134]	Identifier of type N_TYPE_UINT [578] specifying camera gain value. The gain range can be acquired using CAMERAP_GAIN_MIN [1134] and CAMERAP_GAIN_MAX [1134].
CAMERAP_GAIN_MAX [1134]	Read only identifier of type N_TYPE_UINT [578] specifying maximum camera gain value.
CAMERAP_GAIN_MIN [1134]	Read only identifier of type N_TYPE_UINT [578] specifying minimum camera gain value.
CAMERAP_IP_CHANNEL_ID [1134]	Identifier of type N_TYPE_STRING [578] specifying Cisco [18] IP camera streaming Channel Id.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

CAMERAP_IP_CHANNEL_NAME [1135]	Identifier of type N_TYPE_STRING [578] specifying Cisco [18] IP camera streaming Channel name.
CAMERAP_IP_PASSWORD [1135]	Identifier of type N_TYPE_STRING [578] specifying Cisco [18] IP camera password.
CAMERAP_IP_USERNAME [1135]	Identifier of type N_TYPE_STRING [578] specifying Cisco [18] IP camera user name.
CAMERAP_MIRROR_HORIZONTAL [1135]	Identifier of type N_TYPE_BOOL [576] specifying whether received frame should be mirrored horizontally.
CAMERAP_MIRROR_VERTICAL [1135]	Identifier of type N_TYPE_BOOL [576] specifying whether received frame should be mirrored vertically.
CAMERAP_VIDEO_DROP_FRAMES [1135]	Identifier specifying whether Virtual video file camera should drop frames or not. Default is set to False so that every returned frame is the next frame in video file.
CAMERAP_VIDEO_FILE_NAME [1136]	Identifier of type N_TYPE_STRING [578] specifying video file name.
CameraSetParameterA [1136]	Sets the value of specified parameter of the specified camera.
CameraSetParameterW [1136]	Sets the value of specified parameter of the specified camera.

Structs, Records, Enums

	Name	Description
	CameraVideoFormat_ [1131]	Describes camera video format.
	CameraVideoFormat [1131]	Describes camera video format.

7.1.3.1.1 Functions

7.1.3.1.1.1 CameraGetCurrentFrame Function

Gets current frame from the capture device.

C++

```
NResult N_API CameraGetCurrentFrame(HCamera hCamera, HNImage * pHImage);
```

Parameters

Parameters	Description
HCamera hCamera	[in] Handle to the Camera object.
HNImage * pHImage	[out] Points to HNImage [1276] object which receives current frame from the camera. Return Value List

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hCamera* or *pHImage* is NULL [1075].

Module

Camera Unit [1126]

7.1.3.1.1.2 CameraGetDevice Function

Gets the NCamera [1145] device from the specified camera.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API CameraGetDevice(HCamera hCamera, HNCamera * phDevice);
```

Parameters

Parameters	Description
HCamera hCamera	[in] Handle to the camera.
HNCamera * phDevice	[out] Pointer to a handle to the NCamera [1145] device.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Camera Unit [1126]

7.1.3.1.1.3 CameraGetId Function

Gets associated device identifier.

C++

```
NResult N_API CameraGetId(HCamera hCamera, NChar * szValue);
```

Parameters

Parameters	Description
HCamera hCamera	[in] Handle to the Camera object.
NChar * szValue	[out] Pointer to string that receives camera identifier.

Returns

If the function succeeds and *szValue* is NULL [1075], the return value is length of the string (not including the NULL [1075]-terminator) *szValue* should point to.

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hCamera* is NULL [1075].

Module

Camera Unit [1126]

7.1.3.1.1.4 CameraGetVideoFormat Function

Gets current video format for the capture device.

C++

```
NResult N_API CameraGetVideoFormat(HCamera hCamera, CameraVideoFormat * pVideoFormat);
```

Parameters

Parameters	Description
HCamera hCamera	[in] Handle to the Camera object.
CameraVideoFormat * pVideoFormat	[out] Points to CameraVideoFormat [1131] structure that describes current camera video format.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hCamera* or *pVideoFormat* is NULL [1075].

Module

Camera Unit [1126]

7.1.3.1.1.5 CameraGetVideoFormats Function

Gets all supported video format for the capture device.

C++

```
NResult N_API CameraGetVideoFormats(HCamera hCamera, CameraVideoFormat * arVideoFormats);
```

Parameters

Parameters	Description
HCamera hCamera	[in] Handle to the Camera object.
CameraVideoFormat * arVideoFormats	[out] Points to CameraVideoFormat [1131] structure that describes all supported video formats for the current capture device.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hCamera* or *arVideoFormats* is NULL [1075].

Module

Camera Unit [1126]

7.1.3.1.1.6 CameraIsCapturing Function

Checks camera status.

C++

```
NResult N_API CameraIsCapturing(HCamera hCamera, NBool * pValue);
```

Parameters

Parameters	Description
HCamera hCamera	[in] Handle to the Camera object.
NBool * pValue	[out] Pointer to NBool [1004] that receives value indicating whether camera is already capturing.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hCamera* or *pValue [870]* is NULL [1075].

Module

Camera Unit [1126]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.1.1.7 CameraSetVideoFormat Function

Sets video format for the capture device.

C++

```
NResult N_API CameraSetVideoFormat(HCamera hCamera, const CameraVideoFormat * pVideoFormat);
```

Parameters

Parameters	Description
HCamera hCamera	[in] Handle to the Camera object.
const CameraVideoFormat * pVideoFormat	[in] Points to CameraVideoFormat [1131] structure that describes camera video format.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hCamera* or *pVideoFormat* is NULL [1075].
N_E_INVALID_OPERATION [430]	Capturing is already started

Module

Camera Unit [1126]

7.1.3.1.1.8 CameraStartCapturing Function

Starts capturing.

C++

```
NResult N_API CameraStartCapturing(HCamera hCamera);
```

Parameters

Parameters	Description
HCamera hCamera	[in] Handle to the Camera object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hCamera* is NULL [1075].
N_E_INVALID_OPERATION [430]	The camera is already capturing.
N_E_FAILED [429]	Unspecified error has occurred.

Module

Camera Unit [1126]

7.1.3.1.1.9 CameraStopCapturing Function

Stops capturing.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API CameraStopCapturing(HCamera hCamera);
```

Parameters

Parameters	Description
HCamera hCamera	[in] Handle to the Camera object.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*hCamera* is NULL [1075].

Module

Camera Unit [1126]

7.1.3.1.1.10 HCamera

Handle to Camera object.

Remarks

The following parameters can be passed to function with corresponding handle:

- CAMERAP_AUTOMATIC_SETTINGS [1133]
- CAMERAP_MIROR_HORIZONTAL
- CAMERAP_MIROR_VERTICAL

Module

Camera Unit [1126]

7.1.3.1.2 Structs, Records, Enums

7.1.3.1.2.1 CameraVideoFormat Structure

Describes camera video format.

C++

```
typedef struct CameraVideoFormat_ {
  NInt FrameWidth;
  NInt FrameHeight;
  NFloat FrameRate;
} CameraVideoFormat;
```

Members

Members	Description
NInt FrameWidth;	Number of frames captured per second.
NInt FrameHeight;	Width of the frame image.
NFloat FrameRate;	Height of the frame image.

See Also

CameraVideoFormat_

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

Camera Unit [1126]

7.1.3.1.3 Macros

7.1.3.1.3.1 CameraCopyParameters Macro

Copies parameters from one camera to another camera.

C++

```
#define CameraCopyParameters(hDstCamera, hSrcCamera) NObjectCopyParameters(hDstCamera,
hSrcCamera)
```

Parameters

Parameters	Description
hDstCamera	[in] Handle to the destination camera.
hSrcCamera	[in] Handle to the source camera.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Camera Unit [1126]

7.1.3.1.3.2 CameraGetParameterA Macro

Gets the specified parameter of the specified camera.

C++

```
#define CameraGetParameterA(hCamera, parameterId, pValue) NObjectGetParameterA(hCamera,
parameterId, pValue)
```

Parameters

Parameters	Description
hCamera	[in] Handle to the camera.
parameterId	[in] ID of the parameter.
pValue	[out] Pointer to the parameter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Camera Unit [1126]

7.1.3.1.3.3 CameraGetParameterW Macro

Gets the specified parameter of the specified camera.

C++

```
#define CameraGetParameterW(hCamera, parameterId, pValue) NObjectGetParameterW(hCamera,
parameterId, pValue)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
hCamera	[in] Handle to the camera.
parameterId	[in] ID of the parameter.
pValue	[out] Pointer to the parameter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Camera Unit [1126]

7.1.3.1.3.4 CAMERAP_AUTOMATIC_SETTINGS Macro

Identifier of type N_TYPE_BOOL [576] specifying whether automatic camera settings should be used.

C++

```
#define CAMERAP_AUTOMATIC_SETTINGS 10301
```

Module

Camera Unit [1126]

7.1.3.1.3.5 CAMERAP_EXPOSURE Macro

Identifier of type N_TYPE_UINT [578] specifying camera exposure (total amount of light allowed to fall on the camera during taking video frame). The exposure range can be acquired using CAMERAP_EXPOSURE_MIN [1133] and CAMERAP_EXPOSURE_MAX [1133].

C++

```
#define CAMERAP_EXPOSURE 10420
```

Remarks

This parameter is used only when automatic settings is disabled. See CAMERAP_AUTOMATIC_SETTINGS [1133].

Module

Camera Unit [1126]

7.1.3.1.3.6 CAMERAP_EXPOSURE_MAX Macro

Read only identifier of type N_TYPE_UINT [578] specifying camera maximum exposure value.

C++

```
#define CAMERAP_EXPOSURE_MAX 10422
```

Remarks

This parameter is used only when automatic settings is disabled. See CAMERAP_AUTOMATIC_SETTINGS [1133].

Module

Camera Unit [1126]

7.1.3.1.3.7 CAMERAP_EXPOSURE_MIN Macro

Read only identifier of type N_TYPE_UINT [578] specifying camera's minimum exposure value.

C++

```
#define CAMERAP_EXPOSURE_MIN 10421
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

This parameter is used only when automatic settings is disabled. See CAMERAP_AUTOMATIC_SETTINGS [1133].

Module

Camera Unit [1126]

7.1.3.1.3.8 CAMERAP_GAIN Macro

Identifier of type N_TYPE_UINT [578] specifying camera gain value. The gain range can be acquired using CAMERAP_GAIN_MIN [1134] and CAMERAP_GAIN_MAX [1134].

C++

```
#define CAMERAP_GAIN 10410
```

Remarks

This parameter is used only when automatic settings is disabled. See CAMERAP_AUTOMATIC_SETTINGS [1133].

Module

Camera Unit [1126]

7.1.3.1.3.9 CAMERAP_GAIN_MAX Macro

Read only identifier of type N_TYPE_UINT [578] specifying maximum camera gain value.

C++

```
#define CAMERAP_GAIN_MAX 10412
```

Remarks

This parameter is used only when automatic settings is disabled. See CAMERAP_AUTOMATIC_SETTINGS [1133]

Module

Camera Unit [1126]

7.1.3.1.3.10 CAMERAP_GAIN_MIN Macro

Read only identifier of type N_TYPE_UINT [578] specifying minimum camera gain value.

C++

```
#define CAMERAP_GAIN_MIN 10411
```

Remarks

This parameter is used only when automatic settings is disabled. See CAMERAP_AUTOMATIC_SETTINGS [1133].

Module

Camera Unit [1126]

7.1.3.1.3.11 CAMERAP_IP_CHANNEL_ID Macro

Identifier of type N_TYPE_STRING [578] specifying Cisco [18] IP camera streaming Channel Id.

C++

```
#define CAMERAP_IP_CHANNEL_ID 10503
```

Module

Camera Unit [1126]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.1.3.12 CAMERAP_IP_CHANNEL_NAME Macro

Identifier of type N_TYPE_STRING [578] specifying Cisco [18] IP camera streaming Channel name.

C++

```
#define CAMERAP_IP_CHANNEL_NAME 10504
```

Module

Camera Unit [1126]

7.1.3.1.3.13 CAMERAP_IP_PASSWORD Macro

Identifier of type N_TYPE_STRING [578] specifying Cisco [18] IP camera password.

C++

```
#define CAMERAP_IP_PASSWORD 10502
```

Module

Camera Unit [1126]

7.1.3.1.3.14 CAMERAP_IP_USERNAME Macro

Identifier of type N_TYPE_STRING [578] specifying Cisco [18] IP camera user name.

C++

```
#define CAMERAP_IP_USERNAME 10501
```

Module

Camera Unit [1126]

7.1.3.1.3.15 CAMERAP_MIRROR_HORIZONTAL Macro

Identifier of type N_TYPE_BOOL [576] specifying whether received frame should be mirrored horizontally.

C++

```
#define CAMERAP_MIRROR_HORIZONTAL 10200
```

Module

Camera Unit [1126]

7.1.3.1.3.16 CAMERAP_MIRROR_VERTICAL Macro

Identifier of type N_TYPE_BOOL [576] specifying whether received frame should be mirrored vertically.

C++

```
#define CAMERAP_MIRROR_VERTICAL 10201
```

Module

Camera Unit [1126]

7.1.3.1.3.17 CAMERAP_VIDEO_DROP_FRAMES Macro

Identifier specifying whether Virtual video file camera should drop frames or not.

Default is set to False so that every returned frame is the next frame in video file.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define CAMERAP_VIDEO_DROP_FRAMES 10602
```

Module

Camera Unit [1126]

7.1.3.1.3.18 CAMERAP_VIDEO_FILE_NAME Macro

Identifier of type N_TYPE_STRING [578] specifying video file name.

C++

```
#define CAMERAP_VIDEO_FILE_NAME 10601
```

Module

Camera Unit [1126]

7.1.3.1.3.19 CameraSetParameterA Macro

Sets the value of specified parameter of the specified camera.

C++

```
#define CameraSetParameterA(hCamera, parameterId, pValue) NObjectSetParameterA(hCamera,
parameterId, pValue)
```

Parameters

Parameters	Description
hCamera	[in] Handle to the camera.
parameterId	[in] ID of the parameter.
pValue	[in] Pointer to the new parameter value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Camera Unit [1126]

7.1.3.1.3.20 CameraSetParameterW Macro

Sets the value of specified parameter of the specified camera.

C++

```
#define CameraSetParameterW(hCamera, parameterId, pValue) NObjectSetParameterW(hCamera,
parameterId, pValue)
```

Parameters

Parameters	Description
hCamera	[in] Handle to the camera.
parameterId	[in] ID of the parameter.
pValue	[in] Pointer to the new parameter value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Camera Unit [1126]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.1.4 Files

7.1.3.1.4.1 Camera.h

Header file for Camera module. Provides functionality for managing cameras.

Functions

	Name	Description
	CameraGetCurrentFrame [1127]	Gets current frame from the capture device.
	CameraGetDevice [1127]	Gets the NCamera [1145] device from the specified camera.
	CameraGetId [1128]	Gets associated device identifier.
	CameraGetVideoFormat [1128]	Gets current video format for the capture device.
	CameraGetVideoFormats [1129]	Gets all supported video format for the capture device.
	CameraIsCapturing [1129]	Checks camera status.
	CameraSetVideoFormat [1130]	Sets video format for the capture device.
	CameraStartCapturing [1130]	Starts capturing.
	CameraStopCapturing [1130]	Stops capturing.

Macros

Name	Description
CameraCopyParameters [1132]	Copies parameters from one camera to another camera.
CameraGetParameterA [1132]	Gets the specified parameter of the specified camera.
CameraGetParameterW [1132]	Gets the specified parameter of the specified camera.
CameraSetParameterA [1136]	Sets the value of specified parameter of the specified camera.
CameraSetParameterW [1136]	Sets the value of specified parameter of the specified camera.

Module

Camera Unit [1126]

Structures

	Name	Description
	CameraVideoFormat_ [1131]	Describes camera video format.
	CameraVideoFormat [1131]	Describes camera video format.

7.1.3.1.4.2 CameraParams.h

Header file for CameraParams module. Provides parameters for camera.

Macros

Name	Description
CAMERAP_AUTOMATIC_SETTINGS [1133]	Identifier of type N_TYPE_BOOL [576] specifying whether automatic camera settings should be used.
CAMERAP_EXPOSURE [1133]	Identifier of type N_TYPE_UINT [578] specifying camera exposure (total amount of light allowed to fall on the camera during taking video frame). The exposure range can be acquired using CAMERAP_EXPOSURE_MIN [1133] and CAMERAP_EXPOSURE_MAX [1133].
CAMERAP_EXPOSURE_MAX [1133]	Read only identifier of type N_TYPE_UINT [578] specifying camera maximum exposure value.
CAMERAP_EXPOSURE_MIN [1133]	Read only identifier of type N_TYPE_UINT [578] specifying camera's minimum exposure value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

CAMERAP_GAIN [1134]	Identifier of type N_TYPE_UINT [578] specifying camera gain value. The gain range can be acquired using CAMERAP_GAIN_MIN [1134] and CAMERAP_GAIN_MAX [1134].
CAMERAP_GAIN_MAX [1134]	Read only identifier of type N_TYPE_UINT [578] specifying maximum camera gain value.
CAMERAP_GAIN_MIN [1134]	Read only identifier of type N_TYPE_UINT [578] specifying minimum camera gain value.
CAMERAP_IP_CHANNEL_ID [1134]	Identifier of type N_TYPE_STRING [578] specifying Cisco [18] IP camera streaming Channel Id.
CAMERAP_IP_CHANNEL_NAME [1135]	Identifier of type N_TYPE_STRING [578] specifying Cisco [18] IP camera streaming Channel name.
CAMERAP_IP_PASSWORD [1135]	Identifier of type N_TYPE_STRING [578] specifying Cisco [18] IP camera password.
CAMERAP_IP_USERNAME [1135]	Identifier of type N_TYPE_STRING [578] specifying Cisco [18] IP camera user name.
CAMERAP_MIRROR_HORIZONTAL [1135]	Identifier of type N_TYPE_BOOL [576] specifying whether received frame should be mirrored horizontally.
CAMERAP_MIRROR_VERTICAL [1135]	Identifier of type N_TYPE_BOOL [576] specifying whether received frame should be mirrored vertically.
CAMERAP_VIDEO_DROP_FRAMES [1135]	Identifier specifying whether Virtual video file camera should drop frames or not. Default is set to False so that every returned frame is the next frame in video file.
CAMERAP_VIDEO_FILE_NAME [1136]	Identifier of type N_TYPE_STRING [578] specifying video file name.

Module

Camera Unit [1126]

7.1.3.2 CameraMan Unit

Manages, enumerates and creates Cameras.

Notes

This module is obsolete. Use NDeviceManager [1168] instead.

Files

Name	Description
CameraMan.h [1141]	Header file for CameraMan module. Provides functionality for managing, enumerating and creating Cameras.

Functions

	Name	Description
	CameraManGetCamera [1139]	Retrieves the camera at the specified index.
	CameraManGetCameraById [1139]	Retrieves the camera by specified identifier.
	CameraManGetCameraCount [1140]	Retrieves the number of cameras.
	CameraManGetCameraFromDevice [1140]	Gets a camera from NCamera [1145] object.
	CameraManGetDeviceManager [1140]	Retrieves the device manager.
	CameraManInitialize [1141]	Initializes CameraMan library.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	CameraManUninitialize [1141]	Uninitializes CameraMan library.

7.1.3.2.1 Functions

7.1.3.2.1.1 CameraManGetCamera Function

Retrieves the camera at the specified index.

C++

```
NResult N_API CameraManGetCamera(NInt index, HCamera * phCamera);
```

Parameters

Parameters	Description
NInt index	[in] Index of camera to retrieve.
HCamera * phCamera	[out] Points to HCamera [1131] that receives handle to Camera.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*phCamera* is NULL [1075].
N_E_INVALID_OPERATION [430]	There are no initialized CameraMan [1138] objects.

Module

CameraMan Unit [1138]

7.1.3.2.1.2 CameraManGetCameraById Function

Retrieves the camera by specified identifier.

C++

```
NResult N_API CameraManGetCameraById(const NChar * szId, HCamera * phCamera);
```

Parameters

Parameters	Description
const NChar * szId	[in] Points to string that specifies the camera.
HCamera * phCamera	[out] Points to HCamera [1131] that receives handle to Camera.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*phCamera* is NULL [1075].
N_E_INVALID_OPERATION [430]	There are no initialized CameraMan [1138] objects.

Module

CameraMan Unit [1138]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.2.1.3 CameraManGetCameraCount Function

Retrieves the number of cameras.

C++

```
NResult N_API CameraManGetCameraCount(NInt * pValue);
```

Parameters

Parameters	Description
NInt * pValue	[out] Pointer to NInt [1006] that receives number of cameras.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*pHCamera* is NULL [1075].
N_E_INVALID_OPERATION [430]	There are no initialized CameraMan [1138] objects.

Module

CameraMan Unit [1138]

7.1.3.2.1.4 CameraManGetCameraFromDevice Function

Gets a camera from NCamera [1145] object.

C++

```
NResult N_API CameraManGetCameraFromDevice(HNCamera hDevice, HCamera * phCamera);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the NCamera [1145] device.
HCamera * phCamera	[out] Pointer to a handle to the Camera object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

CameraMan Unit [1138]

7.1.3.2.1.5 CameraManGetDeviceManager Function

Retrieves the device manager.

C++

```
NResult N_API CameraManGetDeviceManager(HNDeviceManager * phDeviceManager);
```

Parameters

Parameters	Description
HNDeviceManager * phDeviceManager	[out] Pointer to a handle to the NDeviceManager.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

CameraMan Unit [1138]

7.1.3.2.1.6 CameraManInitialize Function

Initializes CameraMan [1138] library.

C++

```
NResult N_API CameraManInitialize();
```

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_FAILED [429]	Unspecified error has occurred.

Module

CameraMan Unit [1138]

7.1.3.2.1.7 CameraManUninitialize Function

Uninitializes CameraMan [1138] library.

C++

```
void N_API CameraManUninitialize();
```

Module

CameraMan Unit [1138]

7.1.3.2.2 Files

7.1.3.2.2.1 CameraMan.h

Header file for CameraMan [1138] module. Provides functionality for managing, enumerating and creating Cameras.

Functions

	Name	Description
	CameraManGetCamera [1139]	Retrieves the camera at the specified index.
	CameraManGetCameraById [1139]	Retrieves the camera by specified identifier.
	CameraManGetCameraCount [1140]	Retrieves the number of cameras.
	CameraManGetCameraFromDevice [1140]	Gets a camera from NCamera [1145] object.
	CameraManGetDeviceManager [1140]	Retrieves the device manager.
	CameraManInitialize [1141]	Initializes CameraMan [1138] library.
	CameraManUninitialize [1141]	Uninitializes CameraMan [1138] library.

Module

CameraMan Unit [1138]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.3 NBiometricDevice Unit

NBiometricDevice is the main unit of fingerprint and palmprint scanning devices. This unit can be used to retrieve device vendor Id, check spoof detection and to start/stop capturing sequence.

Functions

	Name	Description
	NBiometricDeviceCancel [1142]	Cancels (stops) current capture operation of NBiometricDevice (NFScanner or NIrisScanner).
	NBiometricDeviceEndSequence [1142]	Ends multiple images capturing sequence.
	NBiometricDeviceGetBiometricType [1143]	Gets biometric types supported by specified device.
	NBiometricDeviceGetProductId [1143]	Gets device Id.
	NBiometricDeviceGetSpoofDetection [1144]	Checks if the device is set to detect spoof objects.
	NBiometricDeviceGetVendorId [1144]	Gets device vendor Id.
	NBiometricDeviceIsSpoofDetectionSupported [1144]	Checks if the device is capable to detect spoof objects.
	NBiometricDeviceSetSpoofDetection [1145]	Sets the device to detect spoof objects.
	NBiometricDeviceStartSequence [1145]	Starts sequence which prepares device for multiple capture operations.

7.1.3.3.1 Functions

7.1.3.3.1.1 NBiometricDeviceCancel Function

Cancels (stops) current capture operation of NBiometricDevice [1142] (NFScanner or NIrisScanner).

C++

```
NResult N_API NBiometricDeviceCancel(HNBiometricDevice hDevice);
```

Parameters

Parameters	Description
HNBiometricDevice hDevice	[in] Handle to NBiometricDevice [1142] which should be canceled.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

Also NBiometricDeviceEndSequence [1142] function can be called which cancels capturing and finishes the sequence.

Module

NBiometricDevice Unit [1142]

7.1.3.3.1.2 NBiometricDeviceEndSequence Function

Ends multiple images capturing sequence.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NBiometricDeviceEndSequence(HNBiometricDevice hDevice);
```

Parameters

Parameters	Description
HNBiometricDevice hDevice	[in] Handle to NBiometricDevice [1142] .

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If some capturing has not finished yet, it will be interrupted (see NBiometricDeviceCancel [1142]).

See Also

NBiometricDeviceStartSequence [1145]

Module

NBiometricDevice Unit [1142]

7.1.3.3.1.3 NBiometricDeviceGetBiometricType Function

Gets biometric types supported by specified device.

C++

```
NResult N_API NBiometricDeviceGetBiometricType(HNBiometricDevice hDevice, NBiometricType *
pValue);
```

Parameters

Parameters	Description
HNBiometricDevice hDevice	[in] Handle to NBiometricDevice [1142] to retrieve supported biometric types.
NBiometricType * pValue	[out]Supported biometric types.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricDevice Unit [1142]

7.1.3.3.1.4 NBiometricDeviceGetProductId Function

Gets device Id.

C++

```
NResult N_API NBiometricDeviceGetProductId(HNBiometricDevice hDevice, NUShort * pValue);
```

Parameters

Parameters	Description
HNBiometricDevice hDevice	[in] Handle to NBiometricDevice [1142].
NUShort * pValue	[out] Device product Id.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricDevice Unit [1142]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.3.1.5 NBiometricDeviceGetSpoofDetection Function

Checks if the device is set to detect spoof objects.

C++

```
NResult N_API NBiometricDeviceGetSpoofDetection(HNBiometricDevice hDevice, NBool * pValue);
```

Parameters

Parameters	Description
HNBiometricDevice hDevice	[in] Handle to NBiometricDevice [1142].
NBool * pValue	[out] NBool [1004] type value which indicates if the device is set to detect spoof objects.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricDevice Unit [1142]

7.1.3.3.1.6 NBiometricDeviceGetVendorId Function

Gets device vendor Id.

C++

```
NResult N_API NBiometricDeviceGetVendorId(HNBiometricDevice hDevice, NUShort * pValue);
```

Parameters

Parameters	Description
HNBiometricDevice hDevice	[in] Handle to NBiometricDevice [1142].
NUShort * pValue	[out] Device vendor Id.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricDevice Unit [1142]

7.1.3.3.1.7 NBiometricDeviceIsSpoofDetectionSupported Function

Checks if the device is capable to detect spoof objects.

C++

```
NResult N_API NBiometricDeviceIsSpoofDetectionSupported(HNBiometricDevice hDevice, NBool *
pValue);
```

Parameters

Parameters	Description
HNBiometricDevice hDevice	[in] Handle to NBiometricDevice [1142].
NBool * pValue	[out] NBool [1004] type value which indicates if the device is capable to detect spoof objects.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricDevice Unit [1142]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.3.1.8 NBiometricDeviceSetSpoofDetection Function

Sets the device to detect spoof objects.

C++

```
NResult N_API NBiometricDeviceSetSpoofDetection(HNBiometricDevice hDevice, NBool value);
```

Parameters

Parameters	Description
HNBiometricDevice hDevice	[in] Handle to NBiometricDevice [1142].
NBool value	[out] NBool [1004] type value: NTrue [1064] - to enable spoof objects detection; NFalse [1041] - to disable spoof objects detection.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NBiometricDevice Unit [1142]

7.1.3.3.1.9 NBiometricDeviceStartSequence Function

Starts sequence which prepares device for multiple capture operations.

C++

```
NResult N_API NBiometricDeviceStartSequence(HNBiometricDevice hDevice);
```

Parameters

Parameters	Description
HNBiometricDevice hDevice	[in] Handle to NBiometricDevice [1142] which should be prepared for multiple capturing.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

Some devices do not support multiple capturing (sequences). When called for such devices, this function does nothing.

NBiometricDeviceStartSequence function does not start capturing. Capturing should be started using NFScannerCaptureEx or NIrisScannerCaptureEx functions. These functions should be called for each fingerprint/palmprint or iris image.

If capturing sequence was started, after capturing it should be ended by calling NBiometricDeviceEndSequence [1142] function.

Note: It is not recommended but user can capture multiple images without calling NBiometricDeviceStartSequence and NBiometricDeviceEndSequence [1142] functions.

Module

NBiometricDevice Unit [1142]

7.1.3.4 NCamera Unit

Provides functions for working with cameras.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NCameraAddStillCaptured [1146]	Adds user specified callback (controlling captured still image event) to NCamera.
	NCameraAddStillCapturedCallback [1147]	Adds user specified callback for captured still image event. It is called when still image is captured. User can specify how to handle captured image.
	NCameraCaptureStill [1147]	Gets still image from capture device (camera).
	NCameraFocus [1148]	Adjusts camera focus.
	NCameraGetCurrentStillFormat [1148]	Gets current still image format.
	NCameraGetFocusRegion [1148]	Gets coordinates of rectangular camera focus area.
	NCameraGetFrame [1149]	Gets current frame from the capture device.
	NCameraGetStillFormats [1149]	Gets still image formats supported by specified camera.
	NCameraIsFocusRegionSupported [1149]	Checks if focus region is supported on specified camera.
	NCameraIsFocusSupported [1150]	Checks if specified camera supports autofocus.
	NCameraIsStillCaptureSupported [1150]	Checks if still image obtain from camera is supported.
	NCameraRemoveStillCaptured [1151]	Removed user specified callback (controlling captured still image event) to NCamera.
	NCameraRemoveStillCapturedCallback [1151]	Removes user specified callback for captured still image event.
	NCameraResetFocus [1151]	Resets camera focus.
	NCameraSetCurrentStillFormat [1152]	Sets current still image format.
	NCameraSetFocusRegion [1152]	Sets coordinates of rectangular camera focus area.

Macros

Name	Description
NCameraAddIsCapturingChangedCallback [1153]	Adds user specified callback (controlling capturing changed event) to NCamera.
NCameraIsCapturing [1153]	Checks if the camera is capturing.
NCameraRemoveIsCapturingChangedCallback [1154]	Removes user specified callback (controlling capturing changed event) from NCamera.
NCameraStartCapturing [1154]	Starts capturing.
NCameraStopCapturing [1154]	Stops capturing.

Structs, Records, Enums

	Name	Description
	NCameraStatus_ [1152]	Enumerates camera status values.
	NCameraStatus [1152]	Enumerates camera status values.

7.1.3.4.1 Functions

7.1.3.4.1.1 NCameraAddStillCaptured Function

Adds user specified callback (controlling captured still image event) to NCamera [1145].

C++

```
NResult N_API NCameraAddStillCaptured(HNCamera hDevice, HNCallback hCallback);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to NCamera [1145] object.
HNCallback hCallback	[in] Pointer to user defined function (callback) for handling captured still images.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.1.2 NCameraAddStillCapturedCallback Function

Adds user specified callback for captured still image event. It is called when still image is captured. User can specify how to handle captured image.

C++

```
NResult N_API NCameraAddStillCapturedCallback(HNCamera hDevice,
NCameraStillCapturedCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to NCamera [1145] object.
NCameraStillCapturedCallback pCallback	[in] Pointer to user defined function (callback) for handling captured still images.
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.1.3 NCameraCaptureStill Function

Gets still image from capture device (camera).

C++

```
NResult N_API NCameraCaptureStill(HNCamera hDevice, NCameraStatus * pStatus);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the NCamera [1145] object.
NCameraStatus * pStatus	[out] Camera status.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

Captured still image (NImage [1232]) is retrieved by calling NCameraAddStillCapturedCallback [1147] (user defined function).

Module

NCamera Unit [1145]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.4.1.4 NCameraFocus Function

Adjusts camera focus.

C++

```
NResult N_API NCameraFocus(HNCamera hDevice, NCameraStatus * pStatus);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the NCamera [1145] object.
NCameraStatus * pStatus	[out] Camera status.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.1.5 NCameraGetCurrentStillFormat Function

Gets current still image format.

C++

```
NResult N_API NCameraGetCurrentStillFormat(HNCamera hDevice, HNVideoFormat * phFormat);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the NCamera [1145] object.
HNVideoFormat * phFormat	[out] One of NVideoFormat [1395] values.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.1.6 NCameraGetFocusRegion Function

Gets coordinates of rectangular camera focus area.

C++

```
NResult N_API NCameraGetFocusRegion(HNCamera hDevice, struct NRectF_ * pValue, NBool *
pHasValue);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the NCamera [1145] object.
struct NRectF_ * pValue	[out] NRectF [1227] type structure defining a rectangle of focus region.
NBool * pHasValue	[out] Nullable structure indicating if *pValue [870]* has value. If *pHasValue == NTrue [1064]*, **pValue [870]* has a value. If *pHasValue == NFalse [1041]*, **pValue [870]* has not a value. If *pHasValue == NULL [1075]*, no value for **pValue [870]* was returned.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.1.7 NCameraGetFrame Function

Gets current frame from the capture device.

C++

```
NResult N_API NCameraGetFrame(HNCamera hDevice, HNImage * phImage);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the Camera object.
HNImage * phImage	[out] Points to HNImage [1276] object which receives current frame from the camera.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

The function always returns recent frame only. I.e. if all the frames are required, the NCameraGetFrame function should be called often enough, otherwise the older frames are dropped.

Module

NCamera Unit [1145]

7.1.3.4.1.8 NCameraGetStillFormats Function

Gets still image formats supported by specified camera.

C++

```
NResult N_API NCameraGetStillFormats(HNCamera hDevice, HNVideoFormat * * parhFormats, NInt
* pFormatCount);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the NCamera [1145] object.
HNVideoFormat * * parhFormats	[out] The allocated array of NVideoFormat [1395] structures describing all supported formats for the current capture device (camera).
NInt * pFormatCount	[out] Number of items returned in *parhFormats*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.1.9 NCameraIsFocusRegionSupported Function

Checks if focus region is supported on specified camera.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NCameraIsFocusRegionSupported(HNCamera hDevice, NBool * pValue);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the NCamera [1145] object.
NBool * pValue	[out] NBool [1004] type value indicating if focus region is supported on the camera.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.1.10 NCameraIsFocusSupported Function

Checks if specified camera supports autofocus.

C++

```
NResult N_API NCameraIsFocusSupported(HNCamera hDevice, NBool * pValue);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the NCamera [1145] object.
NBool * pValue	[out] NBool [1004] type value indicating if autofocus is supported on the specified camera.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.1.11 NCameraIsStillCaptureSupported Function

Checks if still image obtain from camera is supported.

C++

```
NResult N_API NCameraIsStillCaptureSupported(HNCamera hDevice, NBool * pValue);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the NCamera [1145] object.
NBool * pValue	[out] NBool [1004] type value indicating if still image obtain is supported on the specified camera.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.4.1.12 NCameraRemoveStillCaptured Function

Removed user specified callback (controlling captured still image event) to NCamera [1145].

C++

```
NResult N_API NCameraRemoveStillCaptured(HNCamera hDevice, HNCallback hCallback);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to NCamera [1145] object.
HNCallback hCallback	[in] Pointer to user defined function (callback) for handling captured still images.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.1.13 NCameraRemoveStillCapturedCallback Function

Removes user specified callback for captured still image event.

C++

```
NResult N_API NCameraRemoveStillCapturedCallback(HNCamera hDevice,
NCameraStillCapturedCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to NCamera [1145] object.
NCameraStillCapturedCallback pCallback	[in] Pointer to user defined function (callback) for handling captured still images.
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

NCameraRemoveStillCapturedCallback

Module

NCamera Unit [1145]

7.1.3.4.1.14 NCameraResetFocus Function

Resets camera focus.

C++

```
NResult N_API NCameraResetFocus(HNCamera hDevice);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the NCamera [1145] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NCamera Unit [1145]

7.1.3.4.1.15 NCameraSetCurrentStillFormat Function

Sets current still image format.

C++

```
NResult N_API NCameraSetCurrentStillFormat(HNCamera hDevice, HNVideoFormat hFormat);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the NCamera [1145] object.
HNVideoFormat hFormat	[in] One of NVideoFormat [1395] values.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.1.16 NCameraSetFocusRegion Function

Sets coordinates of rectangular camera focus area.

C++

```
NResult N_API NCameraSetFocusRegion(HNCamera hDevice, const struct NRectF_ * pValue);
```

Parameters

Parameters	Description
HNCamera hDevice	[in] Handle to the NCamera [1145] object.
const struct NRectF_ * pValue	[in] NRectF [1227] type structure defining a rectangle of focus region.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.2 Structs, Records, Enums

7.1.3.4.2.1 NCameraStatus Enumeration

Enumerates camera status values.

C++

```
typedef enum NCameraStatus_ {
  ncsNone = 0,
  ncsOk = 1,
  ncsAutoFocusFailure = 2,
  ncsLensClosed = 3,
  ncsMirrorUp = 4,
  ncsSensorCleaning = 5,
  ncsSilentOperation = 6,
  ncsStroboCharge = 7,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
    ncsNoLens = 8
  } NCameraStatus;
```

Members

Members	Description
ncsNone = 0	Reserved for future use.
ncsOk = 1	Indicates that camera is ready to use.
ncsAutoFocusFailure = 2	Indicates that camera failed to auto focus an object.
ncsLensClosed = 3	Indicates that camera lens are closed.
ncsMirrorUp = 4	Indicates that camera mirror is up (when DSLR is used).
ncsSensorCleaning = 5	Indicates that camera is performing sensor cleaning.
ncsSilentOperation = 6	Indicates that camera is operating in silent mode.
ncsStroboCharge = 7	Indicates that camera strobe is charging.
ncsNoLens = 8	Indicates that no lens were attached to the camera.

Module

NCamera Unit [1145]

7.1.3.4.3 Macros

7.1.3.4.3.1 NCameraAddIsCapturingChangedCallback Macro

Adds user specified callback (controlling capturing changed event) to NCamera [1145].

C++

```
#define NCameraAddIsCapturingChangedCallback(hDevice, pCallback, pParam)
NCaptureDeviceAddIsCapturingChangedCallback(hDevice, pCallback, pParam)
```

Parameters

Parameters	Description
hDevice	[in] Handle to the NCamera [1145] object.
pCallback	[in] Pointer to user defined function (callback).
pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.3.2 NCameraIsCapturing Macro

Checks if the camera is capturing.

C++

```
#define NCameraIsCapturing(hDevice, pValue) NCaptureDeviceIsCapturing(hDevice, pValue)
```

Parameters

Parameters	Description
hDevice	[in] Handle to the NCamera [1145] object.
pValue	[out] NBool [1004] type value which indicates if the specified NCamera [1145] is capturing media data.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.3.3 NCameraRemoveIsCapturingChangedCallback Macro

Removes user specified callback (controlling capturing changed event) from NCamera [1145].

C++

```
#define NCameraRemoveIsCapturingChangedCallback(hDevice, pCallback, pParam)
NCaptureDeviceRemoveIsCapturingChangedCallback(hDevice, pCallback, pParam)
```

Parameters

Parameters	Description
hDevice	[in] Handle to the NCamera [1145] object.
pCallback	[in] Pointer to user defined function (callback).
pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.3.4 NCameraStartCapturing Macro

Starts capturing.

C++

```
#define NCameraStartCapturing(hDevice) NCaptureDeviceStartCapturing(hDevice)
```

Parameters

Parameters	Description
hDevice	[in] Handle to the NCamera [1145] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.4.3.5 NCameraStopCapturing Macro

Stops capturing.

C++

```
#define NCameraStopCapturing(hDevice) NCaptureDeviceStopCapturing(hDevice)
```

Parameters

Parameters	Description
hDevice	[in] Handle to the NCamera [1145] object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCamera Unit [1145]

7.1.3.5 NCaptureDevice Unit

Provides functions for working with capture devices.

Functions

	Name	Description
	NCaptureDeviceAddIsCapturingChanged [1155]	Adds user specified callback (controlling capturing changed event) to NCaptureDevice.
	NCaptureDeviceAddIsCapturingChangedCallback [1155]	Adds user specified callback (controlling capturing changed event) to NCaptureDevice.
	NCaptureDeviceGetCurrentFormat [1156]	Gets current media format of the capture device.
	NCaptureDeviceGetFormats [1156]	Gets all supported formats by the capture device.
	NCaptureDeviceGetMediaType [1157]	Gets device media type.
	NCaptureDeviceIsCapturing [1157]	Checks if the device is capturing.
	NCaptureDeviceRemoveIsCapturingChanged [1157]	Removes user specified callback (controlling capturing changed event) from NCaptureDevice.
	NCaptureDeviceRemoveIsCapturingChangedCallback [1158]	Removes user specified callback (controlling capturing changed event) from NCaptureDevice.
	NCaptureDeviceSetCurrentFormat [1158]	Sets current media format for the capture device.
	NCaptureDeviceStartCapturing [1158]	Starts capturing.
	NCaptureDeviceStopCapturing [1159]	Stops capturing.

7.1.3.5.1 Functions

7.1.3.5.1.1 NCaptureDeviceAddIsCapturingChanged Function

Adds user specified callback (controlling capturing changed event) to NCaptureDevice [1155].

C++

```
NResult N_API NCaptureDeviceAddIsCapturingChanged(HNCaptureDevice hDevice, HNCallback
hCallback);
```

Parameters

Parameters	Description
HNCaptureDevice hDevice	[in] Handle to the NCaptureDevice [1155] object.
pCallback	[in] Pointer to user defined function (callback).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCaptureDevice Unit [1155]

7.1.3.5.1.2 NCaptureDeviceAddIsCapturingChangedCallback Function

Adds user specified callback (controlling capturing changed event) to NCaptureDevice [1155].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NCaptureDeviceAddIsCapturingChangedCallback(HNCaptureDevice hDevice,
NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNCaptureDevice hDevice	[in] Handle to the NCaptureDevice [1155] object.
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCaptureDevice Unit [1155]

7.1.3.5.1.3 NCaptureDeviceGetCurrentFormat Function

Gets current media format of the capture device.

C++

```
NResult N_API NCaptureDeviceGetCurrentFormat(HNCaptureDevice hDevice, HNMediaFormat *
phFormat);
```

Parameters

Parameters	Description
HNCaptureDevice hDevice	[in] Handle to the NCaptureDevice [1155] object.
HNMediaFormat * phFormat	[out] Current media format of capture device.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

See Also

See NMediaFormat [1334] unit for more information on how to retrieve information about media format from handle.

Module

NCaptureDevice Unit [1155]

7.1.3.5.1.4 NCaptureDeviceGetFormats Function

Gets all supported formats by the capture device.

C++

```
NResult N_API NCaptureDeviceGetFormats(HNCaptureDevice hDevice, HNMediaFormat * *
parhFormats, NInt * pFormatCount);
```

Parameters

Parameters	Description
HNCaptureDevice hDevice	[in] Handle to the NCaptureDevice [1155] object.
HNMediaFormat * * parhFormats	[out] Address of the array pointer. The allocated array of NMediaFormat [1334] structures describing all supported formats for the current capture device. **Note**: the returned array should be freed with NObjectFreeArray [543] function (which will free each element using NObjectFree [542] and the array itself using NFree [506]).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt * pFormatCount	[out] Number of items returned in parhFormats.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCaptureDevice Unit [1155]

7.1.3.5.1.5 NCaptureDeviceGetMediaType Function

Gets device media type.

C++

```
NResult N_API NCaptureDeviceGetMediaType(HNCaptureDevice hDevice, NMediaType * pValue);
```

Parameters

Parameters	Description
HNCaptureDevice hDevice	[in] Handle to the NCaptureDevice [1155] object.
NMediaType * pValue	[out] Device media type value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCaptureDevice Unit [1155]

7.1.3.5.1.6 NCaptureDeviceIsCapturing Function

Checks if the device is capturing.

C++

```
NResult N_API NCaptureDeviceIsCapturing(HNCaptureDevice hDevice, NBool * pValue);
```

Parameters

Parameters	Description
HNCaptureDevice hDevice	[in] Handle to the NCaptureDevice [1155] object.
NBool * pValue	[out] NBool [1004] type value which indicates if the specified NCaptureDevice [1155] is capturing media data.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCaptureDevice Unit [1155]

7.1.3.5.1.7 NCaptureDeviceRemoveIsCapturingChanged Function

Removes user specified callback (controlling capturing changed event) from NCaptureDevice [1155].

C++

```
NResult N_API NCaptureDeviceRemoveIsCapturingChanged(HNCaptureDevice hDevice, HNCallback
hCallback);
```

Parameters

Parameters	Description
HNCaptureDevice hDevice	[in] Handle to the NCaptureDevice [1155] object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

pCallback	[in] Pointer to user defined function (callback).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCaptureDevice Unit [1155]

7.1.3.5.1.8 NCaptureDeviceRemoveIsCapturingChangedCallback Function

Removes user specified callback (controlling capturing changed event) from NCaptureDevice [1155].

C++

```
NResult N_API NCaptureDeviceRemoveIsCapturingChangedCallback(HNCaptureDevice hDevice,
NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNCaptureDevice hDevice	[in] Handle to the NCaptureDevice [1155] object.
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCaptureDevice Unit [1155]

7.1.3.5.1.9 NCaptureDeviceSetCurrentFormat Function

Sets current media format for the capture device.

C++

```
NResult N_API NCaptureDeviceSetCurrentFormat(HNCaptureDevice hDevice, HNMediaFormat
hFormat);
```

Parameters

Parameters	Description
HNCaptureDevice hDevice	[in] Handle to the NCaptureDevice [1155] object.
HNMediaFormat hFormat	[in] Current media format to set.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCaptureDevice Unit [1155]

7.1.3.5.1.10 NCaptureDeviceStartCapturing Function

Starts capturing.

C++

```
NResult N_API NCaptureDeviceStartCapturing(HNCaptureDevice hDevice);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNCaptureDevice hDevice	[in] Handle to the NCaptureDevice [1155] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCaptureDevice Unit [1155]

7.1.3.5.1.11 NCaptureDeviceStopCapturing Function

Stops capturing.

C++

```
NResult N_API NCaptureDeviceStopCapturing(HNCaptureDevice hDevice);
```

Parameters

Parameters	Description
HNCaptureDevice hDevice	[in] Handle to the NCaptureDevice [1155] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NCaptureDevice Unit [1155]

7.1.3.6 NDevice Unit

Provides functionality for retrieving general device information.

Files

Name	Description
NDevice.h [1167]	Header file for NDevice Module.

Functions

	Name	Description
	NDeviceGetChild [1160]	Returns the child of specified multimodal device.
	NDeviceGetChildCount [1160]	Returns the child count of specified multimodal device.
	NDeviceGetDeviceType [1161]	Returns the type of specified multimodal device.
	NDeviceGetDisplayName [1161]	Returns device name.
	NDeviceGetId [1162]	Returns device Id,
	NDeviceGetMake [1162]	Returns the manufacturer of the specified device.
	NDeviceGetModel [1162]	Returns model of current biometric device..
	NDeviceGetParent [1163]	Returns the parent of specified multimodal device module.
	NDeviceGetPlugin [1163]	Returns handle to device plugin.
	NDeviceGetSerialNumber [1163]	Returns serial number of current biometric device..
	NDeviceIsAvailable [1164]	Checks if the device is available.
	NDeviceIsDisconnectable [1164]	Checks if specified device can be disconnected.
	NDeviceIsPrivate [1165]	Checks whether the specified device is private.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Macros

Name	Description
NDeviceGetDisplayName [1166]	Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetDisplayName function.
NDeviceGetId [1166]	Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetID function.
NDeviceGetMake [1166]	Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetMake function.
NDeviceGetModel [1167]	Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetModel function.
NDeviceGetSerialNumber [1167]	Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetSerialNumber function.

Structs, Records, Enums

	Name	Description
	NDeviceType_ [1165]	Enumerates device type values.
	NDeviceType [1165]	Enumerates device type values.

7.1.3.6.1 Functions

7.1.3.6.1.1 NDeviceGetChild Function

Returns the child of specified multimodal device.

C++

```
NResult N_API NDeviceGetChild(HNDevice hDevice, NInt index, HNDevice * phValue);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to NDevice [1159] to retrieve its child.
NInt index	[in] Index of child element.
HNDevice * phValue	[out] Pointer to HNDevice which receives child of multimodal device.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NDevice Unit [1159]

7.1.3.6.1.2 NDeviceGetChildCount Function

Returns the child count of specified multimodal device.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NDeviceGetChildCount(HNDevice hDevice, NInt * pValue);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to NDevice [1159] to retrieve its child count.
NInt * pValue	[out] Child count.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDevice Unit [1159]

7.1.3.6.1.3 NDeviceGetDeviceType Function

Returns the type of specified multimodal device.

C++

```
NResult N_API NDeviceGetDeviceType(HNDevice hDevice, NDeviceType * pValue);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to NDevice [1159] to retrieve its type.
NDeviceType * pValue	[out] Pointer to HNDevice which receives type multimodal device.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDevice Unit [1159]

7.1.3.6.1.4 NDeviceGetDisplayName Function

Returns device name.

C++

```
NResult N_API NDeviceGetDisplayName(HNDevice hDevice, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to NDevice [1159] to retrieve its display name.
NChar * szValue	[out] Pointer to HNDevice which receives display name of the device.
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDevice Unit [1159]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.6.1.5 NDeviceGetId Function

Returns device Id,

C++

```
NResult N_API NDeviceGetId(HNDevice hDevice, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to NDevice [1159] to retrieve its Id.
NChar * szValue	[out] Pointer to device Id.
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDevice Unit [1159]

7.1.3.6.1.6 NDeviceGetMake Function

Returns the manufacturer of the specified device.

C++

```
NResult N_API NDeviceGetMake(HNDevice hDevice, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to NDevice [1159] to retrieve its manufacturer.
NChar * szValue	[out] Pointer to device manufacturer name.
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDevice Unit [1159]

7.1.3.6.1.7 NDeviceGetModel Function

Returns model of current biometric device..

C++

```
NResult N_API NDeviceGetModel(HNDevice hDevice, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to NDevice [1159] to retrieve its model.
NChar * szValue	[out] Pointer to device model.
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDevice Unit [1159]

7.1.3.6.1.8 NDeviceGetParent Function

Returns the parent of specified multimodal device module.

C++

```
NResult N_API NDeviceGetParent(HNDevice hDevice, HNDevice * phValue);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to NDevice [1159] to retrieve its parent.
HNDevice * phValue	[out] Pointer to HNDevice handle which receives parent of specified child device.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NDevice Unit [1159]

7.1.3.6.1.9 NDeviceGetPlugin Function

Returns handle to device plugin.

C++

```
NResult N_API NDeviceGetPlugin(HNDevice hDevice, HNPlugin * phValue);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to NDevice [1159] to retrieve its handle.
HNPlugin * phValue	[out] Pointer to HNPlugin handle which receives plugin of specified device.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NDevice Unit [1159]

7.1.3.6.1.10 NDeviceGetSerialNumber Function

Returns serial number of current biometric device..

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NDeviceGetSerialNumber(HNDevice hDevice, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to NDevice [1159] to retrieve its serial number.
NChar * szValue	[out] Pointer to device serial number.
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDevice Unit [1159]

7.1.3.6.1.11 NDeviceIsAvailable Function

Checks if the device is available.

C++

```
NResult N_API NDeviceIsAvailable(HNDevice hDevice, NBool * pValue);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to NDevice [1159] to check if it is available.
NBool * pValue	[out] NBool [1004] value indicating whether or not specified device is available.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDevice Unit [1159]

7.1.3.6.1.12 NDeviceIsDisconnectable Function

Checks if specified device can be disconnected.

C++

```
NResult N_API NDeviceIsDisconnectable(HNDevice hDevice, NBool * pValue);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to NDevice [1159].
NBool * pValue	[out] NBool [1004] value indicating whether or not specified device is disconnectable.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NDevice Unit [1159]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.6.1.13 NDeviceIsPrivate Function

Checks whether the specified device is private.

C++

```
NResult N_API NDeviceIsPrivate(HNDevice hDevice, NBool * pValue);
```

Parameters

Parameters	Description
HNDevice hDevice	[in] Handle to the NDevice [1159] object to be checked.
NBool * pValue	[out] Pointer to a Boolean value: NTrue [1064] if the device is private; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NDevice Unit [1159]

7.1.3.6.2 Structs, Records, Enums

7.1.3.6.2.1 NDeviceType Enumeration

Enumerates device type values.

C++

```
typedef enum NDeviceType_ {
  ndtNone = 0,
  ndtCamera = 0x01,
  ndtBiometricDevice = 0x02,
  ndtFScanner = 0x04,
  ndtFingerScanner = 0x08,
  ndtPalmScanner = 0x10,
  ndtIrisScanner = 0x20,
  ndtCaptureDevice = 0x40,
  ndtMicrophone = 0x80,
  ndtAny = 0x80000000
} NDeviceType;
```

Members

Members	Description
ndtNone = 0	This values is used for primitive devices (e.g. for parent devices). `NDeviceManager` has such devices only if `NDeviceManager` was created using `ndtAny`.
ndtCamera = 0x01	Indicates that device type is camera.
ndtBiometricDevice = 0x02	Indicates `ndtFScanner` and `ndtIrisScanner` values. For instance, if this value is returned by `NDeviceGetDeviceType` [1161] function, it means that `NDeviceManager` type is one of the following: `ndtFScanner` or `ndtIrisScanner`.
ndtFScanner = 0x04	Indicates that devices is a scanner which can be used for friction ridges (fingers or palms) scanning. If `ndtFScanner` value is passed to `NDeviceManagerCreate` [1171] function, it means that device which supports all friction ridge scanning devices will be created.
ndtFingerScanner = 0x08	Indicates that device type is fingerprints scanner (`ndtFScanner` scanner which can also scan fingerprints).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

ndtPalmScanner = 0x10	Indicates that device type is palmprints scanner (`ndtFScanner` scanner which can also scan palmprints).
ndtIrisScanner = 0x20	Indicates that device type is iris scanner.
ndtCaptureDevice = 0x40	Indicates `ndtCamera` and `ndtMicrophone` values.
ndtMicrophone = 0x80	Indicates that device type is microphone.
ndtAny = 0x80000000	This value indicates that `NDeviceManager` supports all available device types. For instance, when `ndtAny` value is passed to function `NDeviceManagerCreate` [1171], new `NDeviceManager` which supports all device modules (cameras, fingerprint, iris, palmprint scanners, etc.) is created. Also this value is returned when `NDeviceManager` does not recognize device's type (e.g. device's module version is newer than NDeviceManager). `NDeviceManager` type can be retrieved using `NDeviceGetDeviceType` [1161] function.

Module

NDevice Unit [1159]

7.1.3.6.3 Macros

7.1.3.6.3.1 NDeviceGetDisplayName Macro

Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetDisplayName function.

C++

```
#define NDeviceGetDisplayName N_FUNC_AW(NDeviceGetDisplayName)
```

Module

NDevice Unit [1159]

7.1.3.6.3.2 NDeviceGetId Macro

Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetID function.

C++

```
#define NDeviceGetId N_FUNC_AW(NDeviceGetId)
```

Module

NDevice Unit [1159]

7.1.3.6.3.3 NDeviceGetMake Macro

Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetMake function.

C++

```
#define NDeviceGetMake N_FUNC_AW(NDeviceGetMake)
```

Module

NDevice Unit [1159]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.6.3.4 NDeviceGetModel Macro

Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetModel function.

C++

```
#define NDeviceGetModel N_FUNC_AW(NDeviceGetModel)
```

Module

NDevice Unit [1159]

7.1.3.6.3.5 NDeviceGetSerialNumber Macro

Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetSerialNumber function.

C++

```
#define NDeviceGetSerialNumber N_FUNC_AW(NDeviceGetSerialNumber)
```

Module

NDevice Unit [1159]

7.1.3.6.4 Files

7.1.3.6.4.1 NDevice.h

Header file for NDevice [1159] Module.

Enumerations

	Name	Description
	NDeviceType_ [1165]	Enumerates device type values.
	NDeviceType [1165]	Enumerates device type values.

Functions

	Name	Description
	NDeviceGetChild [1160]	Returns the child of specified multimodal device.
	NDeviceGetChildCount [1160]	Returns the child count of specified multimodal device.
	NDeviceGetDeviceType [1161]	Returns the type of specified multimodal device.
	NDeviceGetDisplayName [1161]	Returns device name.
	NDeviceGetId [1162]	Returns device Id,
	NDeviceGetMake [1162]	Returns the manufacturer of the specified device.
	NDeviceGetModel [1162]	Returns model of current biometric device..
	NDeviceGetParent [1163]	Returns the parent of specified multimodal device module.
	NDeviceGetPlugin [1163]	Returns handle to device plugin.
	NDeviceGetSerialNumber [1163]	Returns serial number of current biometric device..
	NDeviceIsAvailable [1164]	Checks if the device is available.
	NDeviceIsDisconnectable [1164]	Checks if specified device can be disconnected.
	NDeviceIsPrivate [1165]	Checks whether the specified device is private.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Macros

Name	Description
NDeviceGetDisplayName [1166]	Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetDisplayName function.
NDeviceGetId [1166]	Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetID function.
NDeviceGetMake [1166]	Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetMake function.
NDeviceGetModel [1167]	Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetModel function.
NDeviceGetSerialNumber [1167]	Picks a Unicode (W) or ANSCII (A) version of the NDeviceGetSerialNumber function.

Module

NDevice Unit [1159]

7.1.3.7 NDeviceManager Unit

Provides functions for device enumeration (listing the devices available in the system) and getting events when device is added or removed.

Files

Name	Description
NDeviceManager.h [1177]	Header file for NDeviceManager library. Provides general functions for working with devices.

Functions

	Name	Description
	NDeviceManagerAddDeviceAddedCallback [1169]	Adds user specified device added callback to NDeviceManager.
	NDeviceManagerAddDeviceRemovedCallback [1169]	Adds user specified device removed callback to NDeviceManager.
	NDeviceManagerAddDevicesChangedCallback [1170]	Adds user specified end device update callback to NDeviceManager.
	NDeviceManagerAddDevicesChangingCallback [1170]	Adds user specified begin device update callback to NDeviceManager.
	NDeviceManagerAddDevicesRefreshedCallback [1170]	Adds user specified device refreshed callback to NDeviceManager.
	NDeviceManagerCreate [1171]	Creates new NDeviceManager.
	NDeviceManagerGetAutoUpdate [1172]	Checks whether NDeviceManager automatically updates devices list.
	NDeviceManagerGetDevice [1172]	Retrieves device by specified index.
	NDeviceManagerGetDeviceById [1172]	Retrieves device by specified Id.
	NDeviceManagerGetDeviceCount [1173]	Retrieves the count of connected devices.
	NDeviceManagerGetDeviceTypes [1173]	Returns supported device types by specified NDeviceManager.
	NDeviceManagerGetPluginManager [1174]	Returns handle to NPluginManager [607].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NDeviceManagerRefresh [1174]	Refreshes plugged devices list.
	NDeviceManagerRemoveDeviceAddedCallback [1174]	Removes user specified device added callback from NDeviceManager.
	NDeviceManagerRemoveDeviceRemovedCallback [1175]	Removes user specified device removed callback from NDeviceManager.
	NDeviceManagerRemoveDevicesChangedCallback [1175]	Removes user specified end devices changed callback from NDeviceManager.
	NDeviceManagerRemoveDevicesChangingCallback [1176]	Removes user specified begin device changing callback from NDeviceManager.
	NDeviceManagerRemoveDevicesRefreshedCallback [1176]	Removes user specified device refreshed callback to NDeviceManager.

Macros

Name	Description
NDeviceManagerGetDeviceById [1176]	Picks a Unicode (W) or ANSCII (A) version of the NDeviceManagerGetDeviceById function.
NDeviceManagerGetInfoA [1177]	Retrieves NDevices [1178] library information.
NDeviceManagerGetInfoW [1177]	Retrieves NDevices [1178] library information.

7.1.3.7.1 Functions

7.1.3.7.1.1 NDeviceManagerAddDeviceAddedCallback Function

Adds user specified device added callback to NDeviceManager.

C++

```
NResult N_API NDeviceManagerAddDeviceAddedCallback(HNDeviceManager hDeviceManager,
NDeviceManagerDeviceCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NDeviceManagerDeviceCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.2 NDeviceManagerAddDeviceRemovedCallback Function

Adds user specified device removed callback to NDeviceManager.

C++

```
NResult N_API NDeviceManagerAddDeviceRemovedCallback(HNDeviceManager hDeviceManager,
NDeviceManagerDeviceCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NDeviceManagerDeviceCallback pCallback	[in] Pointer to user defined function (callback).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

void * pParam	[in] User data used by a callback function.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.3 NDeviceManagerAddDevicesChangedCallback Function

Adds user specified end device update callback to NDeviceManager.

C++

```
NResult N_API NDeviceManagerAddDevicesChangedCallback(HNDeviceManager hDeviceManager,
NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.4 NDeviceManagerAddDevicesChangingCallback Function

Adds user specified begin device update callback to NDeviceManager.

C++

```
NResult N_API NDeviceManagerAddDevicesChangingCallback(HNDeviceManager hDeviceManager,
NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.5 NDeviceManagerAddDevicesRefreshedCallback Function

Adds user specified device refreshed callback to NDeviceManager.

C++

```
NResult N_API NDeviceManagerAddDevicesRefreshedCallback(HNDeviceManager hDeviceManager,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.6 NDeviceManagerCreate Function

Creates new NDeviceManager.

C++

```
NResult N_API NDeviceManagerCreate(NDeviceType deviceTypes, NBool autoPlug, NBool
autoUpdate, HNDeviceManager * phDeviceManager);
```

Parameters

Parameters	Description
NDeviceType deviceTypes	[in] Device types supported by new NDeviceManager. See NDeviceType [1165] enumeration for more information.
NBool autoPlug	[in] NBool [1004] value for automatic devices listing. If set to NTrue [1064], associated plugin manager will automatically load the plugins (if this haven't been done already).
NBool autoUpdate	[in] NBool [1004] value for automatic devices update. If set to NTrue [1064], NDeviceManager will automatically update devices list when new device is added or removed.
HNDeviceManager * phDeviceManager	[out] Pointer to HNDeviceManager handle which receives created device.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Notes

- When ndtAny is used as a parameter deviceType value, new NDeviceManager which enumerates all devices will be created.
- *autoUpdate* parameter allows one of the two behaviors. The device manager without *autoUpdate* is simpler/safer to use. The device handle returned by device manager would remain valid even if device becomes unavailable. The use of such obsolete handle will be detected and reported as appropriate return value. The user of device manager would than manually perform device list update (NDeviceManagerRefresh [1174] function) when it is safe to do so. The alternative *autoUpdate* device manger allows to always have up-to date list of devices present in the system. But changes to the device availability can happen at any time, the becomes responsible for not using handles of removed devices. And should register to the NDeviceManagerAddDeviceRemovedCallback [1169] and similar callbacks, and take appropriate actions.
- All created device managers use the same plugin manager (NDeviceManagerGetPluginManager [1174] function). It can be used to refresh the device manager plugins or to manually select which plugins should be activated.

Module

NDeviceManager Unit [1168]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.7.1.7 NDeviceManagerGetAutoUpdate Function

Checks whether NDeviceManager automatically updates devices list.

C++

```
NResult N_API NDeviceManagerGetAutoUpdate(HNDeviceManager hDeviceManager, NBool * pValue);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NBool * pValue	[out] NBool [1004] value indicating whether NDeviceManager automatically updates devices list.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.8 NDeviceManagerGetDevice Function

Retrieves device by specified index.

C++

```
NResult N_API NDeviceManagerGetDevice(HNDeviceManager hDeviceManager, NInt index, HNDevice
* phValue);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NInt index	[in] Device index within NDeviceManager.
HNDevice * phValue	[out] Pointer to HNDevice handle which receives device specified by index.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.9 NDeviceManagerGetDeviceById Function

Retrieves device by specified Id.

C++

```
NResult N_API NDeviceManagerGetDeviceById(HNDeviceManager hDeviceManager, const NChar *
szId, HNDevice * phValue);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const NChar * szId	[in] Device identifier.
HNDevice * phValue	[out] Pointer to HNDevice handle which receives device specified by Id.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.10 NDeviceManagerGetDeviceCount Function

Retrieves the count of connected devices.

C++

```
NResult N_API NDeviceManagerGetDeviceCount(HNDeviceManager hDeviceManager, NInt * pValue);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NInt * pValue	[out] Connected devices count. If 0 is returned, it means that no devices are connected.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Notes

If `NDeviceManagerCreate [1171]` parameter `autoPlug` has `NFalse [1041]` value assigned, connected devices count is not updated when new device is connected.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.11 NDeviceManagerGetDeviceTypes Function

Returns supported device types by specified NDeviceManager.

C++

```
NResult N_API NDeviceManagerGetDeviceTypes(HNDeviceManager hDeviceManager, NDeviceType *
pValue);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NDeviceType * pValue	[out] Pointer to NDeviceType [1165] enumeration which contains supported device types.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.12 NDeviceManagerGetPluginManager Function

Returns handle to NPluginManager [607].

C++

```
NResult N_API NDeviceManagerGetPluginManager(HNPluginManager * phValue);
```

Parameters

Parameters	Description
pValue	[out] Pointer to handle of current device NPluginManager [607].

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.13 NDeviceManagerRefresh Function

Refreshes plugged devices list.

C++

```
NResult N_API NDeviceManagerRefresh(HNDeviceManager hDeviceManager);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager to refresh its devices list.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Notes

When `autoUpdate` parameter for function `NDeviceManagerCreate` [1171] was set to `NFalse` [1041], user should call this function to refresh devices list. When `autoUpdate` was set to `NTrue` [1064], `NDeviceManagerRefresh` function is called automatically by `NDeviceManager`.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.14 NDeviceManagerRemoveDeviceAddedCallback Function

Removes user specified device added callback from NDeviceManager.

C++

```
NResult N_API NDeviceManagerRemoveDeviceAddedCallback(HNDeviceManager hDeviceManager,
NDeviceManagerDeviceCallback pCallback, void * pParam);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NDeviceManagerDeviceCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.15 NDeviceManagerRemoveDeviceRemovedCallback Function

Removes user specified device removed callback from NDeviceManager.

C++

```
NResult N_API NDeviceManagerRemoveDeviceRemovedCallback(HNDeviceManager hDeviceManager,
NDeviceManagerDeviceCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NDeviceManagerDeviceCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.16 NDeviceManagerRemoveDevicesChangedCallback Function

Removes user specified end devices changed callback from NDeviceManager.

C++

```
NResult N_API NDeviceManagerRemoveDevicesChangedCallback(HNDeviceManager hDeviceManager,
NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDeviceManager Unit [1168]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.7.1.17 NDeviceManagerRemoveDevicesChangingCallback Function

Removes user specified begin device changing callback from NDeviceManager.

C++

```
NResult N_API NDeviceManagerRemoveDevicesChangingCallback(HNDeviceManager hDeviceManager,
NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDeviceManager Unit [1168]

7.1.3.7.1.18 NDeviceManagerRemoveDevicesRefreshedCallback Function

Removes user specified device refreshed callback to NDeviceManager.

C++

```
NResult N_API NDeviceManagerRemoveDevicesRefreshedCallback(HNDeviceManager hDeviceManager,
NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNDeviceManager hDeviceManager	[in] Handle to NDeviceManager.
NObjectCallback pCallback	[in] Pointer to user defined function (callback).
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NDeviceManager Unit [1168]

7.1.3.7.2 Macros

7.1.3.7.2.1 NDeviceManagerGetDeviceById Macro

Picks a Unicode (W) or ANSCII (A) version of the NDeviceManagerGetDeviceById function.

C++

```
#define NDeviceManagerGetDeviceById N_FUNC_AW(NDeviceManagerGetDeviceById)
```

Module

NDeviceManager Unit [1168]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.3.7.2.2 NDeviceManagerGetInfoA Macro

Retrieves NDevices [1178] library information.

C++

```
#define NDeviceManagerGetInfoA NDevicesGetInfoA
```

Returns

NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDeviceManager Unit [1168]

7.1.3.7.2.3 NDeviceManagerGetInfoW Macro

Retrieves NDevices [1178] library information.

C++

```
#define NDeviceManagerGetInfoW NDevicesGetInfoW
```

Returns

NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NDeviceManager Unit [1168]

7.1.3.7.3 Files

7.1.3.7.3.1 NDeviceManager.h

Header file for NDeviceManager library. Provides general functions for working with devices.

Functions

	Name	Description
	NDeviceManagerAddDeviceAddedCallback [1169]	Adds user specified device added callback to NDeviceManager.
	NDeviceManagerAddDeviceRemovedCallback [1169]	Adds user specified device removed callback to NDeviceManager.
	NDeviceManagerAddDevicesChangedCallback [1170]	Adds user specified end device update callback to NDeviceManager.
	NDeviceManagerAddDevicesChangingCallback [1170]	Adds user specified begin device update callback to NDeviceManager.
	NDeviceManagerAddDevicesRefreshedCallback [1170]	Adds user specified device refreshed callback to NDeviceManager.
	NDeviceManagerCreate [1171]	Creates new NDeviceManager.
	NDeviceManagerGetAutoUpdate [1172]	Checks whether NDeviceManager automatically updates devices list.
	NDeviceManagerGetDevice [1172]	Retrieves device by specified index.
	NDeviceManagerGetDeviceById [1172]	Retrieves device by specified Id.
	NDeviceManagerGetDeviceCount [1173]	Retrieves the count of connected devices.
	NDeviceManagerGetDeviceTypes [1173]	Returns supported device types by specified NDeviceManager.
	NDeviceManagerGetPluginManager [1174]	Returns handle to NPluginManager [607].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NDeviceManagerRefresh [1174]	Refreshes plugged devices list.
	NDeviceManagerRemoveDeviceAddedCallback [1174]	Removes user specified device added callback from NDeviceManager.
	NDeviceManagerRemoveDeviceRemovedCallback [1175]	Removes user specified device removed callback from NDeviceManager.
	NDeviceManagerRemoveDevicesChangedCallback [1175]	Removes user specified end devices changed callback from NDeviceManager.
	NDeviceManagerRemoveDevicesChangingCallback [1176]	Removes user specified begin device changing callback from NDeviceManager.
	NDeviceManagerRemoveDevicesRefreshedCallback [1176]	Removes user specified device refreshed callback to NDeviceManager.

Macros

Name	Description
NDeviceManagerGetDeviceById [1176]	Picks a Unicode (W) or ANSCII (A) version of the NDeviceManagerGetDeviceById function.
NDeviceManagerGetInfoA [1177]	Retrieves NDevices [1178] library information.
NDeviceManagerGetInfoW [1177]	Retrieves NDevices [1178] library information.

Module

NDeviceManager Unit [1168]

7.1.3.8 NDevices Unit

Provides information about NDevices library.

Functions

	Name	Description
	NDevicesGetInfo [1178]	Retrieves device library information and returns it as a pointer.

7.1.3.8.1 Functions

7.1.3.8.1.1 NDevicesGetInfo Function

Retrieves device library information and returns it as a pointer.

C++

```
NResult N_API NDevicesGetInfo(NLibraryInfo * pValue);
```

Parameters

Parameters	Description
NLibraryInfo * pValue	[out] A pointer to device library information.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NDevices Unit [1178]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.4 NLicensing Library

Provides functionality for getting, releasing licenses.

Remarks

Requirements (Windows)

- **Import Library**: NLicensing.dll.lib
- **DLLs**: *NLicensing.dll*

Requirements (Linux)

- **Shared objects**: *libNLicensing.so*

Concurrent license. Client and License Server scenario.

Modules

Name	Description
NLicense Unit [1179]	Main functions for working with licenses.
NLicensing Unit [1194]	Provides functionality for getting, releasing licenses.
NLicensingService Unit [1195]	Manages licensing services.

7.1.4.1 NLicense Unit

Main functions for working with licenses.

Files

Name	Description
NLicense.h [1193]	Header file for the NLicense library. Provides internal license functionality.

Functions

	Name	Description
	NLicenseActivateOnline [1180]	Activates specified Id (that was obtained through calling NLicenseGenerateId [1184] function or using ActivationWizardDotNet) and obtains license file for this Id via internet.
	NLicenseGetInfo [1181]	Retrieves license information of specified product.
	NLicenseDeactivateOnline [1181]	Deactivates specified license via internet using specified Id generator.
	NLicenseDeactivateOnlineWithId [1181]	Deactivates specified license via internet using specified deactivation Id.
	NLicenseGetInfoForComponent [1182]	Retrieves license information of specified component.
	NLicenseGenerateDeactivationId [1183]	Generates deactivation Id for specified provided product on current computer.
	NLicenseGenerateDeactivationIdForLicense [1183]	Generates deactivation Id for specified license on current computer.
	NLicenseGenerateId [1184]	Generates Id for activation from serial number for a particular machine.
	NLicenseIsComponentActivated [1184]	Checks if component specified by name is activated.
	NLicenseObtain [1185]	Obtains product license from the specified server.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NLicenseObtainComponents [1187]	Obtains component license from the specified server.
	NLicenseRelease [1189]	Releases license.
	NLicenseReleaseComponents [1190]	Releases component license.

Macros

Name	Description
NLicenseActivateOnline [1191]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseActivateOnline function.
NLicenseDeactivateOnline [1191]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseDeactivateOnline function.
NLicenseDeactivateOnlineWithId [1191]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseDeactivateOnlineWithId function.
NLicenseGenerateDeactivationId [1192]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseGenerateDeactivationId function.
NLicenseGenerateDeactivationIdForLicense [1192]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseGenerateDeactivationIdForLicense function.
NLicenseGenerateId [1192]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseGenerateId function.
NLicenseGetInfoForComponent [1192]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseGetInfoForComponent function.
NLicenseIsComponentActivated [1192]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseIsComponentActivated function.
NLicenseObtainComponents [1193]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseObtainComponents function.
NLicenseReleaseComponents [1193]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseReleaseComponents function.

Structs, Records, Enums

	Name	Description
	NLicenseInfo_ [1190]	Enumerates values describing license.

Types

Name	Description
NLicenseInfo [1191]	Enumerates values describing license.

7.1.4.1.1 Functions

7.1.4.1.1.1 NLicenseActivateOnline Function

Activates specified Id (that was obtained through calling NLicenseGenerateId [1184] function or using ActivationWizardDotNet) and obtains license file for this Id via internet.

C++

```
NResult N_API NLicenseActivateOnline(const NChar * szId, NChar ** pszLicense);
```

Parameters

Parameters	Description
const NChar * szId	[in] Id generated from serial number.
NChar ** pszLicense	[out] Char buffer that contains activated license data.

Returns

If the function succeeds, the return value is N_OK [433]. If an error appears one of NError [398] value is returned.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

Important! Internet connection is required for this function to succeed.

NLicenseActivateOnline returns internally allocated char array for activated license (*pszLicense*). *pszLicense* must be freed using NObjectFree [542] function from NCore [257] library after it is no longer used.

Module

NLicense Unit [1179]

7.1.4.1.1.2 NLicenseGetInfo Function

Retrieves license information of specified product.

C++

```
NResult N_API NLicenseGetInfo(const NChar * szProduct, NLicenseInfo * pLicenseInfo);
```

Parameters

Parameters	Description
const NChar * szProduct	[in] Zero terminated string which contains product name to retrieve license information.
NLicenseInfo * pLicenseInfo	[out] Pointer to NLicenseInfo [1191] structure.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*szProduct* or *pLicenseInfo* is NULL [1075].

Module

NLicense Unit [1179]

7.1.4.1.1.3 NLicenseDeactivateOnline Function

Deactivates specified license via internet using specified Id generator.

C++

```
NResult N_API NLicenseDeactivateOnline(const NChar * szIdGenPath, const NChar * szLicense);
```

Parameters

Parameters	Description
const NChar * szIdGenPath	[in] Path to Id generator program (if NULL [1075] is passed, it is assumed, that id_gen application is in current directory).
const NChar * szLicense	[in] Char buffer that contains activated license's full product name OR license file content.

Returns

If the function succeeds, the return value is N_OK [433]. If an error appears one of NError [398] value is returned.

Module

NLicense Unit [1179]

7.1.4.1.1.4 NLicenseDeactivateOnlineWithId Function

Deactivates specified license via internet using specified deactivation Id.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NLicenseDeactivateOnlineWithId(const NChar * szLicense, const NChar *
szDeactivationId);
```

Parameters

Parameters	Description
const NChar * szLicense	[in] Char buffer that contains activated license name.
const NChar * szDeactivationId	[in] Deactivation Id.

Returns

If the function succeeds, the return value is N_OK [433]. If an error appears one of NError [398] value is returned.

Remarks

Important! Both license and generated deactivation Id should be sent to support@neurotechnology.com in order to complete deactivation process. Deactivation on Neurotechnology side is done manually.

Important! Deactivation must be performed on **the same computer** supplied license was generated for. If license is deactivated on another computer, although call to this function will succeed, but deactivation might be refused by support@neurotechnology.com.

Important! If function succeeds - license will be deactivated on **this** computer and will not be usable.

Notes

Deactivation Id for product can be retrieved using NLicenseGenerateDeactivationId [1183] function.

Module

NLicense Unit [1179]

7.1.4.1.1.5 NLicenseGetInfoForComponent Function

Retrieves license information of specified component.

C++

```
NResult N_API NLicenseGetInfoForComponent(const NChar * szComponent, NLicenseInfo *
pLicenseInfo);
```

Parameters

Parameters	Description
const NChar * szComponent	[in] Zero terminated string which contains component name to retrieve its license information. See NLicenseObtainComponents [1187] function remarks for the list of supported component names for this parameter.
NLicenseInfo * pLicenseInfo	[out] Pointer to NLicenseInfo [1191] structure.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*szComponent* or *pLicenseInfo* is NULL [1075].

Module

NLicense Unit [1179]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.4.1.1.6 NLicenseGenerateDeactivationId Function

Generates deactivation Id for specified provided product on current computer.

C++

```
NResult N_API NLicenseGenerateDeactivationId(const NChar * szIdGenPath, const NChar *
szProductName, NChar ** pszDeactivationId);
```

Parameters

Parameters	Description
const NChar * szIdGenPath	[in] Path to Id generator program (if NULL [1075] is passed, it is assumed, that id_gen application is in current directory).
const NChar * szProductName	[in] Long product name (i.e. Face Extractor, Finger Matcher and etc.).
NChar ** pszDeactivationId	[out] Generated deactivation Id.

Returns

If the function succeeds, the return value is N_OK [433]. If an error appears one of NError [398] value is returned.

Remarks

Important! Both license and generated deactivation Id should be sent to support@neurotechnology.com in order to complete deactivation process. Deactivation on Neurotechnology side is done manually.

Important! Deactivation must be performed on **the same computer** supplied license was generated for. If license is deactivated on another computer, although call to this function will succeed, but deactivation might be refused by support@neurotechnology.com.

Important! If function succeeds - license will be deactivated on **this** computer and will not be usable.

`NLicenseGenerateDeactivationId` returns internally allocated char array for generated Id (*pszDeactivationId*). *pszDeactivationId* must be freed using NObjectFree [542] function from NCore [257] library after it is no longer used.

Module

NLicense Unit [1179]

7.1.4.1.1.7 NLicenseGenerateDeactivationIdForLicense Function

Generates deactivation Id for specified license on current computer.

C++

```
NResult N_API NLicenseGenerateDeactivationIdForLicense(const NChar * szIdGenPath, const
NChar * szLicense, NChar ** pszDeactivationId);
```

Parameters

Parameters	Description
const NChar * szIdGenPath	[in] Path to Id generator program (if NULL [1075] is passed, it is assumed, that id_gen application is in current directory).
const NChar * szLicense	[in] License to be deactivated.
NChar ** pszDeactivationId	[out] Generated deactivation Id.

Returns

If the function succeeds, the return value is N_OK [433]. If an error appears one of NError [398] value is returned.

Remarks

Important! Internet connection is required for this function to succeed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Important! Both license and generated deactivation Id should be sent to support@neurotechnology.com in order to complete deactivation process. Deactivation on Neurotechnology side is done manually.

Important! Deactivation must be performed on **the same computer** supplied license was generated for. If license is deactivated on another computer, although call to this function will succeed, but deactivation might be refused by support@neurotechnology.com.

`NLicenseGenerateDeactivationIdForLicense` returns internally allocated char array for generated Id (*pszDeactivationId*). *pszDeactivationId* must be freed using NObjectFree [542] function from NCore [257] library after it is no longer used.

Module

NLicense Unit [1179]

7.1.4.1.1.8 NLicenseGenerateId Function

Generates Id for activation from serial number for a particular machine.

C++

```
NResult N_API NLicenseGenerateId(const NChar * szIdGenPath, const NChar * szSerial, NChar
** pszId);
```

Parameters

Parameters	Description
const NChar * szIdGenPath	[in] Path to Id generator program (if NULL [1075] is passed, it is assumed, that id_gen application is in current directory).
const NChar * szSerial	[in] Serial number used for generating Id.
NChar ** pszId	[out] Generated Id.

Returns

If the function succeeds, the return value is N_OK [433]. If an error appears one of NError [398] value is returned.

Remarks

NLicenseGenerateId returns internally allocated char array for generated Id (*pszId*). *pszId* must be freed using NObjectFree [542] function from NCore [257] library after it is no longer used.

Example

```
NChar * szSerial = N_T("3502-266E-60C6-6738-7D1C-3872-5AD8-3C42");
NChar * szId = NULL;
NGenerateId(N_T("d:\\Activation\\id_gen.exe"), szSerial, &szId);
```

Module

NLicense Unit [1179]

7.1.4.1.1.9 NLicenseIsComponentActivated Function

Checks if component specified by name is activated.

C++

```
NResult N_API NLicenseIsComponentActivated(const NChar * szName, NBool * pValue);
```

Parameters

Parameters	Description
const NChar * szName	[in] Zero terminated string which contains component name. See NLicenseObtainComponents [1187] function remarks for the list of supported component names for this parameter.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBool * pValue	[out] NBool [1004] value indicating if the component was activated.

Returns

If the function succeeds, the return value is N_OK [433]. If application working folder does not contain required files, N_E_FILE_NOT_FOUND [432] error is thrown.

Module

NLicense Unit [1179]

7.1.4.1.1.10 NLicenseObtain Function

Obtains product license from the specified server.

C++

```
NResult N_API NLicenseObtain(const NChar * szAddress, const NChar * szPort, const NChar *
szProducts, NBool * pAvailable);
```

Parameters

Parameters	Description
const NChar * szAddress	[in] Server address where license manager is installed as a server. License manager can be in the same or in other computer. **/local** used when license is tied with the computer using serial number.
const NChar * szPort	[in] License manager server port.
const NChar * szProducts	[in] Zero terminated string which contains license name. For a complete list of available values for this parameter see Remarks section.
NBool * pAvailable	[out] Returns NTrue [1064] if license available; otherwise, NFalse [1041].

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*szAddress*, *szPort*, *szProducts* or *pAvailable* is NULL [1075].
N_E_FORMAT [429]	*szProducts* incorrect format.
N_E_FAILED [429]	Unspecified error has occurred.

Remarks

These values are available for *szProducts* parameter (by default concurrent license is obtained):

1. **VeriFinger SDK**:
   1. SingleComputerLicense:VFExtractor or concurrent:VFExtractor
   2. SingleComputerLicense:VFMatcher

VFExtractor - VeriFinger Extractor

VFMatcher - VeriFinger Matcher

2. **VeriLook SDK**:
   1. SingleComputerLicense:VLExtractor or concurrent:VLExtractor
   2. SingleComputerLicense:VLMatcher

Copyright © 2009-2014 Neurotechnology. All rights reserved.

VLExtractor - VeriLook Extractor

VLMatcher - VeriLook Matcher

3. **VeriLook Surveillance SDK**:
   1. SingleComputerLicense:VLSurveillance
4. **MegaMatcher SDK**:
   1. SingleComputerLicense:MegaMatcherServer
   2. SingleComputerLicense:MegaMatcherClusterServer
   3. SingleComputerLicense:MegaMatcherClusterNode
   4. SingleComputerLicense:MegaMatcherClient
   5. concurrent:MegaMatcherClient
5. **VeriEye SDK**:
   1. SingleComputerLicense:VEExtractor
   2. SingleComputerLicense:VEMatcher

VEExtractor - VeriEye Extractor

VEMatcher - VeriEye Matcher

6. **BSS SDK**:
   1. SingleComputerLicense:VFBSS
   2. SingleComputerLicense:VLBSS
   3. SingleComputerLicense:VEBSS
7. **VeriFinger VAR SDK**:
   1. SingleComputerLicense:VFExtractorV
   2. SingleComputerLicense:VFMatcherV

VFExtractorV - VeriFinger Extractor

VFMatcherV - VeriFinger Matcher

8. **VeriLook VAR SDK**:
   1. SingleComputerLicense:VLExtractorV
   2. SingleComputerLicense:VLMatcherV

VLExtractorV - VeriLook Extractor

VLMatcherV - VeriLook Matcher

9. **VeriEye VAR SDK**:
   1. SingleComputerLicense:VEExtractorV
   2. SingleComputerLicense:VEMatcherV

VEExtractorV - VeriEye Extractor

VEMatcherV - VeriEye Matcher

10. **SentiSight SDK:**
    1. SingleComputerLicense:SentiSight

Note: nonconcurrent licenses (SingleComputerLicense) **can not be released** with NLicenseRelease [1189] function, but .NLicenseRelease [1189] function must be called before program exits.

If concurrent license is in use, then NLicenseRelease [1189] function should be called after features extraction.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Notes

Acquired license must be released using NLicenseRelease [1189] function.

This function should be called when license name is known. Otherwise, NLicenseObtainComponents [1187] should be called.

Module

NLicense Unit [1179]

7.1.4.1.1.11 NLicenseObtainComponents Function

Obtains component license from the specified server.

C++

```
NResult N_API NLicenseObtainComponents(const NChar * szAddress, const NChar * szPort, const NChar * szComponents, NBool * pAvailable);
```

Parameters

Parameters	Description
const NChar * szAddress	[in] Server address where license manager is installed as a server. License manager can be in the same or in other computer. **/local** is used when license is tied with the computer using serial number.
const NChar * szPort	[in] License manager server port.
const NChar * szComponents	[in] Zero terminated string which contains license name for component. For a complete list of available values for this parameter see Remarks section.
NBool * pAvailable	[out] Returns NTrue [1064] if license for component isavailable; otherwise, NFalse [1041].

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Remarks

`NLicenseObtainComponents` is a wrapper of `NLicenseObtain [1185]` function. This function should be used when a license(-s) for component(-s) (not a product) is required. `NLicenseObtainComponents` function starts by attempting to retrieve *Concurrent* licenses for specified components. If no *Concurrent* licenses were found, this function retrieves *Single computer license*.

Note: no special string or identifier is required to be added to *szComponents* parameter to differentiate between *Concurrent* and *Single computer license*.

These values are available for *szComponents* parameter:

1. NImages:
   1. Images.WSQ
   2. Images.IHead [1201]
   3. Images.JPEG2000
   4. Images.LosslessJPEG
2. NImageProc:
   1. Images.Processing.FFT
3. NDevices [1178]:
   1. Devices.Cameras

Copyright © 2009-2014 Neurotechnology. All rights reserved.

   2. Devices.FingerScanners
   3. Devices.IrisScanners
   4. Devices.PalmScanners
4. NBiometricStandards:
   1. Biometrics.Standards.Base
   2. Biometrics.Standards.FingerTemplates
   3. Biometrics.Standards.FingerCardTemplates
   4. Biometrics.Standards.Fingers
   5. Biometrics.Standards.Faces [45]
   6. Biometrics.Standards.Irises
   7. Biometrics.Standards.Palms
   8. Biometrics.Standards.PalmTemplates
   9. Biometrics.Standards.Other
5. bioapi20
   1. BioAPI.Base
   2. BioAPI.Fingers
   3. BioAPI.Faces [45]
   4. BioAPI.Irises
   5. BioAPI.Palms
6. NBiometricTools:
   1. Biometrics.Tools.NFIQ
   2. Biometrics.Tools.FingerClassification
7. NSurveillance [1439]:
   1. Surveillance
8. SentiSight:
   1. SentiSight
9. ClusterServer, ClusterNode:
   1. Cluster.Server
   2. Cluster.ClusterServer
   3. Cluster.ClusterNode
10. NBiometrics:
   1. Biometrics.FingerDetectionBase
   2. Biometrics.FingerDetection
   3. Biometrics.FingerExtractionBase
   4. Biometrics.FingerExtraction
   5. Biometrics.PalmDetectionBase
   6. Biometrics.PalmDetection
   7. Biometrics.PalmExtractionBase
   8. Biometrics.PalmExtraction
   9. Biometrics.FaceDetectionBase
   10. Biometrics.FaceDetection

Copyright © 2009-2014 Neurotechnology. All rights reserved.

11. Biometrics.FaceExtractionBase
12. Biometrics.FaceExtraction
13. Biometrics.IrisDetectionBase
14. Biometrics.IrisDetection
15. Biometrics.IrisExtractionBase
16. Biometrics.IrisExtraction
17. Biometrics.FingerSegmentation
18. Biometrics.PalmSegmentation
19. Biometrics.FaceSegmentation
20. Biometrics.IrisSegmentation
21. Biometrics.FingerQualityAssessment
22. Biometrics.PalmQualityAssessment
23. Biometrics.FaceQualityAssessment
24. Biometrics.IrisQualityAssessment
25. Biometrics.FingerMatching
26. Biometrics.FingerMatchingFast
27. Biometrics.PalmMatching
28. Biometrics.FaceMatching
29. Biometrics.FaceMatchingFast
30. Biometrics.IrisMatching
31. Biometrics.IrisMatchingFast
32. Biometrics.MatchingFusion

11. NSmartCards:
    1. SmartCards

Notes

- Acquired license must be released using `NLicenseReleaseComponents` [1190] function.
- `NLicenseReleaseComponents` [1190] function should be called as many times as function NLicenseObtainComponents was called.
- `NLicenseObtainComponents` needs to check availability of all the licenses which can provide requested functionality. As an optimization NLicensing.cfg file can be used to limit the checks to be performed. The file is saved to current directory by the Activation Wizard on Windows platform, on other OS the file can be created manually by a user. Each line of the file specifies functionality name (supported by NLicenseObtainComponents) and the list of the licenses to check for. For example:
  - Images.WSQ = FingerClient, WSQ, PalmClient
- `NLicenseObtainComponents` checks all available licenses so it can take time to find a required license.
- If license name is known it is recommended to call `NLicenseObtain` [1185] function.

Module

NLicense Unit [1179]

7.1.4.1.1.12 NLicenseRelease Function

Releases license.

C++

```
NResult N_API NLicenseRelease(const NChar * szProducts);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const NChar * szProducts	[in] Product names to release licenses. See NLicenseObtain [1185] function remarks for the list of supported values of this parameter.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*szProducts* is NULL [1075].
N_E_FORMAT [429]	*szProducts* incorrect format.
N_E_FAILED [429]	Unspecified error has occurred.

Remarks

Nonconcurrent licenses (SingleComputerLicense) **can not be released** with NLicenseRelease function, but NLicenseRelease function must be called before program exits.

If concurrent license is in use, then NLicenseRelease function should be called after features extraction.

Module

NLicense Unit [1179]

7.1.4.1.1.13 NLicenseReleaseComponents Function

Releases component license.

C++

```
NResult N_API NLicenseReleaseComponents(const NChar * szComponents);
```

Parameters

Parameters	Description
const NChar * szComponents	[in] Zero terminated string which contains component name.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Remarks

- See NLicenseObtainComponents [1187] function remarks for the list of supported component names for this parameter.
- This function should be called as many times as function `NLicenseObtainComponents [1187]` was called.

Module

NLicense Unit [1179]

7.1.4.1.2 Structs, Records, Enums

7.1.4.1.2.1 NLicenseInfo_ Structure

Enumerates values describing license.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
struct NLicenseInfo_ {
  NBool IsObtained;
  NInt DistributorId;
  NInt SerialNumber;
};
```

Members

Members	Description
NBool IsObtained;	Indicates if a license obtain was successful.
NInt DistributorId;	Id of a distributor which generated license.
NInt SerialNumber;	License serial number.

Module

NLicense Unit [1179]

7.1.4.1.3 Types

7.1.4.1.3.1 NLicenseInfo Type

Enumerates values describing license.

C++

```
typedef struct NLicenseInfo_ NLicenseInfo;
```

Module

NLicense Unit [1179]

7.1.4.1.4 Macros

7.1.4.1.4.1 NLicenseActivateOnline Macro

Picks a Unicode (W) or ANSCII (A) version of the NLicenseActivateOnline function.

C++

```
#define NLicenseActivateOnline N_FUNC_AW(NLicenseActivateOnline)
```

Module

NLicense Unit [1179]

7.1.4.1.4.2 NLicenseDeactivateOnline Macro

Picks a Unicode (W) or ANSCII (A) version of the NLicenseDeactivateOnline function.

C++

```
#define NLicenseDeactivateOnline N_FUNC_AW(NLicenseDeactivateOnline)
```

Module

NLicense Unit [1179]

7.1.4.1.4.3 NLicenseDeactivateOnlineWithId Macro

Picks a Unicode (W) or ANSCII (A) version of the NLicenseDeactivateOnlineWithId function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NLicenseDeactivateOnlineWithId N_FUNC_AW(NLicenseDeactivateOnlineWithId)
```

Module

NLicense Unit [1179]

7.1.4.1.4.4 NLicenseGenerateDeactivationId Macro

Picks a Unicode (W) or ANSCII (A) version of the NLicenseGenerateDeactivationId function.

C++

```
#define NLicenseGenerateDeactivationId N_FUNC_AW(NLicenseGenerateDeactivationId)
```

Module

NLicense Unit [1179]

7.1.4.1.4.5 NLicenseGenerateDeactivationIdForLicense Macro

Picks a Unicode (W) or ANSCII (A) version of the NLicenseGenerateDeactivationIdForLicense function.

C++

```
#define NLicenseGenerateDeactivationIdForLicense
N_FUNC_AW(NLicenseGenerateDeactivationIdForLicense)
```

Module

NLicense Unit [1179]

7.1.4.1.4.6 NLicenseGenerateId Macro

Picks a Unicode (W) or ANSCII (A) version of the NLicenseGenerateId function.

C++

```
#define NLicenseGenerateId N_FUNC_AW(NLicenseGenerateId)
```

Module

NLicense Unit [1179]

7.1.4.1.4.7 NLicenseGetInfoForComponent Macro

Picks a Unicode (W) or ANSCII (A) version of the NLicenseGetInfoForComponent function.

C++

```
#define NLicenseGetInfoForComponent N_FUNC_AW(NLicenseGetInfoForComponent)
```

Module

NLicense Unit [1179]

7.1.4.1.4.8 NLicenseIsComponentActivated Macro

Picks a Unicode (W) or ANSCII (A) version of the NLicenseIsComponentActivated function.

C++

```
#define NLicenseIsComponentActivated N_FUNC_AW(NLicenseIsComponentActivated)
```

Module

NLicense Unit [1179]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.4.1.4.9 NLicenseObtainComponents Macro

Picks a Unicode (W) or ANSCII (A) version of the NLicenseObtainComponents function.

C++

```
#define NLicenseObtainComponents N_FUNC_AW(NLicenseObtainComponents)
```

Module

NLicense Unit [1179]

7.1.4.1.4.10 NLicenseReleaseComponents Macro

Picks a Unicode (W) or ANSCII (A) version of the NLicenseReleaseComponents function.

C++

```
#define NLicenseReleaseComponents N_FUNC_AW(NLicenseReleaseComponents)
```

Module

NLicense Unit [1179]

7.1.4.1.5 Files

7.1.4.1.5.1 NLicense.h

Header file for the NLicense library. Provides internal license functionality.

Functions

	Name	Description
	NLicenseActivateOnline [1180]	Activates specified Id (that was obtained through calling NLicenseGenerateId [1184] function or using ActivationWizardDotNet) and obtains license file for this Id via internet.
	NLicenseDeactivateOnline [1181]	Deactivates specified license via internet using specified Id generator.
	NLicenseDeactivateOnlineWithId [1181]	Deactivates specified license via internet using specified deactivation Id.
	NLicenseGenerateDeactivationId [1183]	Generates deactivation Id for specified provided product on current computer.
	NLicenseGenerateDeactivationIdForLicense [1183]	Generates deactivation Id for specified license on current computer.
	NLicenseGenerateId [1184]	Generates Id for activation from serial number for a particular machine.
	NLicenseGetInfo [1181]	Retrieves license information of specified product.
	NLicenseGetInfoForComponent [1182]	Retrieves license information of specified component.
	NLicenseIsComponentActivated [1184]	Checks if component specified by name is activated.
	NLicenseObtain [1185]	Obtains product license from the specified server.
	NLicenseObtainComponents [1187]	Obtains component license from the specified server.
	NLicenseRelease [1189]	Releases license.
	NLicenseReleaseComponents [1190]	Releases component license.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Macros

Name	Description
NLicenseActivateOnline [1191]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseActivateOnline function.
NLicenseDeactivateOnline [1191]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseDeactivateOnline function.
NLicenseDeactivateOnlineWithId [1191]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseDeactivateOnlineWithId function.
NLicenseGenerateDeactivationId [1192]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseGenerateDeactivationId function.
NLicenseGenerateDeactivationIdForLicense [1192]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseGenerateDeactivationIdForLicense function.
NLicenseGenerateId [1192]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseGenerateId function.
NLicenseGetInfoForComponent [1192]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseGetInfoForComponent function.
NLicenseIsComponentActivated [1192]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseIsComponentActivated function.
NLicenseObtainComponents [1193]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseObtainComponents function.
NLicenseReleaseComponents [1193]	Picks a Unicode (W) or ANSCII (A) version of the NLicenseReleaseComponents function.

Module

NLicense Unit [1179]

Structures

	Name	Description
	NLicenseInfo_ [1190]	Enumerates values describing license.

Types

Name	Description
NLicenseInfo [1191]	Enumerates values describing license.

7.1.4.2 NLicensing Unit

Provides functionality for getting, releasing licenses.

Files

Name	Description
NLicensing.h [1195]	Header file for the NLicensing library. Provides functionality for getting and releasing licenses.

Functions

	Name	Description
	NLicensingGetInfo [1195]	Gets information about the library.
	NLicensingModuleOf [1195]	

7.1.4.2.1 Functions

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.4.2.1.1 NLicensingGetInfo Function

Gets information about the library.

C++

```
NResult N_API NLicensingGetInfo(NLibraryInfo * pValue);
```

Parameters

Parameters	Description
NLibraryInfo * pValue	[out] Pointer to NLibraryInfo structure that receives library information.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_ARGUMENT_NULL [427]	*pValue [870]* is NULL [1075].

Module

NLicensing Unit [1194]

7.1.4.2.1.2 NLicensingModuleOf

Returns

Handle of the module that can be used with functions from NModule [516] unit to get the information about the module.

Module

NLicensing Unit [1194]

7.1.4.2.2 Files

7.1.4.2.2.1 NLicensing.h

Header file for the NLicensing library. Provides functionality for getting and releasing licenses.

Functions

	Name	Description
	NLicensingGetInfo [1195]	Gets information about the library.

Module

NLicensing Unit [1194]

7.1.4.3 NLicensingService Unit

Manages licensing services.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5 NMedia Library

Provides functionality for working with various type of media items: images, audio and video. This library has functions you can use to manipulate media.

Remarks

Requirements (Windows)

- **Import Library**: NMedia.dll.lib
- **DLL**: *NMedia.dll*

Requirements (Linux)

- **Shared object**: *libNMedia.so*

Modules

Name	Description
Bmp Unit [1196]	Provides functionality for loading and saving images in BMP format.
IHead Unit [1201]	Provides functionality for loading and saving images in NIST IHead format.
Jpeg Unit [1205]	Provides functionality for loading and saving images in JPEG format.
Jpeg2K Unit [1212]	Provides functionality for loading and saving images in JPEG 2000 format.
NGeometry Unit [1222]	Provides definitions of geometrical structures types.
NGrayscaleImage Unit [1228]	Provides functionality for managing 8-bit grayscale images.
NImage Unit [1232]	Provides functionality for managing images.
NImageFile Unit [1294]	Provides functionality for reading image files in format-neutral way.
NImageFormat Unit [1299]	Provides functionality for loading and saving images in format-neutral way.
NImages Unit [1331]	Provides library registration and other additional functionality.
NMediaFormat Unit [1334]	Provides functions to retrieve information about media formats
NMediaReader Unit [1336]	Provides functions which can be used to read media data.
NMediaSource Unit [1345]	Provides functions for working with media source.
NMediaTypes Unit [1352]	Enumerates media types and provides functions to check if media type is valid.
NMonochromeImage Unit [1355]	Creates color wrapper for grayscale image.
NPixelFormat Unit [1359]	Provides functionality for working with image pixel format.
NRgb Unit [1389]	Provides functionality for working with Rgb image data.
NRgbImage Unit [1392]	Provides functionality for managing 24-bit RGB images.
NVideoFormat Unit [1395]	
NVideoReader Unit [1414]	
NVideoWriter Module [1419]	Provides functionality for writing to a video file.
Png Unit [1423]	Provides functionality for loading and saving images in PNG format.
Tiff Unit [1429]	Provides functionality for loading images in TIFF format.
Wsq Unit [1431]	Provides functionality for loading and saving images in WSQ format.

7.1.5.1 Bmp Unit

Provides functionality for loading and saving images in BMP format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Files

Name	Description
Bmp.h [1200]	Header file for Bmp module. Provides functionality for loading and saving images in BMP format.

Functions

	Name	Description
	BmpLoadImageFromFile [1197]	Loads image from BMP file.
	BmpLoadImageFromMemory [1198]	Loads image from memory buffer containing BMP file.
	BmpLoadImageFromStream [1198]	Loads image from stream containing Bmp file.
	BmpSaveImageToFile [1199]	Saves image to file in BMP format.
	BmpSaveImageToMemory [1199]	Saves image to memory buffer in BMP format.
	BmpSaveImageToStream [1200]	Saves image to stream in BMP format.

Macros

Name	Description
BmpLoadImageFromHBitmap [1200]	Picks a Unicode (W) or ANSCII (A) version of the BmpLoadImageFromHBitmap function.
BmpSaveImageToHBitmap [1200]	Picks a Unicode (W) or ANSCII (A) version of the BmpSaveImageToHBitmap function.

7.1.5.1.1 Functions

7.1.5.1.1.1 BmpLoadImageFromFile Function

Loads image from BMP file.

C++

```
NResult N_API BmpLoadImageFromFile(const NChar * szFileName, HNImage * phImage);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Points to string that specifies file name.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. It is returned when *szFileName* or *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. It is returned when format of file specified by *szFileName* is invalid.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Bmp Unit [1196]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.1.1.2 BmpLoadImageFromMemory Function

Loads image from memory buffer containing BMP file.

C++

```
NResult N_API BmpLoadImageFromMemory(const void * pBuffer, NSizeType bufferSize, HNImage *
phImage);
```

Parameters

Parameters	Description
const void * pBuffer	[out] Pointer to memory buffer.
NSizeType bufferSize	[in] Size of memory buffer.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. It is returned when *pBuffer* is NULL [1075] and *bufferSize* is not equal to zero.- or -*phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. It is returned when format of file contained in buffer specified by *pBuffer* is invalid.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Bmp Unit [1196]

7.1.5.1.1.3 BmpLoadImageFromStream Function

Loads image from stream containing Bmp [1196] file.

C++

```
NResult N_API BmpLoadImageFromStream(HNStream hStream, HNImage * phImage);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to HNStream containing Bmp [1196] image.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].

Module

Bmp Unit [1196]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.1.1.4 BmpSaveImageToFile Function

Saves image to file in BMP format.

C++

```
NResult N_API BmpSaveImageToFile(HNImage hImage, const NChar * szFileName);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
const NChar * szFileName	[out] Points to string that specifies file name.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. It is returned when hImage or szFileName is NULL [1075].

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Bmp Unit [1196]

7.1.5.1.1.5 BmpSaveImageToMemory Function

Saves image to memory buffer in BMP format.

C++

```
NResult N_API BmpSaveImageToMemory(HNImage hImage, void * * ppBuffer, NSizeType *
pBufferSize);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
void * * ppBuffer	[out] Pointer to void * that receives pointer to allocated memory buffer.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. It is returned when *hImage*, *ppBuffer* or *pBufferSize* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. It is returned when there was not enough memory to allocate memory buffer.

Remarks

This is a low-level function and can be changed in future version of the library.

Memory buffer allocated by the function must be deallocated using NFree [506] function when it is no longer needed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

Bmp Unit [1196]

7.1.5.1.1.6 BmpSaveImageToStream Function

Saves image to stream in BMP format.

C++

```
NResult N_API BmpSaveImageToStream(HNImage hImage, HNStream hStream);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to HNStream containing Bmp [1196] image.
HNStream hStream	[out] Handle to HNStream where image will be written.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].

Module

Bmp Unit [1196]

7.1.5.1.2 Macros

7.1.5.1.2.1 BmpLoadImageFromHBitmap Macro

Picks a Unicode (W) or ANSCII (A) version of the BmpLoadImageFromHBitmap function.

C++

```
#define BmpLoadImageFromHBitmap(handle, phImage) NImageCreateFromHBitmap((HBITMAP)handle,
0, phImage)
```

Module

Bmp Unit [1196]

7.1.5.1.2.2 BmpSaveImageToHBitmap Macro

Picks a Unicode (W) or ANSCII (A) version of the BmpSaveImageToHBitmap function.

C++

```
#define BmpSaveImageToHBitmap(hImage, pHandle) NImageToHBitmap(hImage, (HBITMAP *)pHandle)
```

Module

Bmp Unit [1196]

7.1.5.1.3 Files

7.1.5.1.3.1 Bmp.h

Header file for Bmp [1196] module. Provides functionality for loading and saving images in BMP format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	BmpLoadImageFromFile [1197]	Loads image from BMP file.
	BmpLoadImageFromMemory [1198]	Loads image from memory buffer containing BMP file.
	BmpLoadImageFromStream [1198]	Loads image from stream containing Bmp [1196] file.
	BmpSaveImageToFile [1199]	Saves image to file in BMP format.
	BmpSaveImageToMemory [1199]	Saves image to memory buffer in BMP format.
	BmpSaveImageToStream [1200]	Saves image to stream in BMP format.

Macros

Name	Description
BmpLoadImageFromHBitmap [1200]	Picks a Unicode (W) or ANSCII (A) version of the BmpLoadImageFromHBitmap function.
BmpSaveImageToHBitmap [1200]	Picks a Unicode (W) or ANSCII (A) version of the BmpSaveImageToHBitmap function.

Module

Bmp Unit [1196]

7.1.5.2 IHead Unit

Provides functionality for loading and saving images in NIST IHead format.

Files

Name	Description
IHead.h [1205]	Header file for IHead Module. Provides functionality for loading and saving images in NIST IHead format.

Functions

	Name	Description
	IHeadLoadImageFromFile [1201]	Loads image from NIST IHead file.
	IHeadLoadImageFromMemory [1202]	Loads image from memory buffer containing NIST IHead file.
	IHeadLoadImageFromStream [1203]	Loads image data from memory stream.
	IHeadSaveImageToFile [1203]	Saves image to the file in NIST IHead format.
	IHeadSaveImageToMemory [1203]	Saves image to the memory buffer in NIST IHead format.
	IHeadSaveImageToStream [1204]	Saves an image to specified memory stream.

7.1.5.2.1 Functions

7.1.5.2.1.1 IHeadLoadImageFromFile Function

Loads image from NIST IHead [1201] file.

C++

```
NResult N_API IHeadLoadImageFromFile(const NChar * szFileName, HNImage * phImage);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const NChar * szFileName	[in] Points to string that specifies file name.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. It is returned when szFileName or phImage is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. It is returned when format of file specified by szFileName is invalid.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

IHead Unit [1201]

7.1.5.2.1.2 IHeadLoadImageFromMemory Function

Loads image from memory buffer containing NIST IHead [1201] file.

C++

```
NResult N_API IHeadLoadImageFromMemory(const void * pBuffer, NSizeType bufferSize, HNImage
* phImage);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer.
NSizeType bufferSize	[in] Size of memory buffer.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. It is returned when: *pBuffer* is NULL [1075] and bufferSize is not equal to zero. - or - *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. It is returned when format of file contained in buffer specified by *pBuffer* is invalid.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

IHead Unit [1201]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.2.1.3 IHeadLoadImageFromStream Function

Loads image data from memory stream.

C++

```
NResult N_API IHeadLoadImageFromStream(HNStream hStream, HNImage * phImage);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to memory stream that contains image data.
HNImage * phImage	[out] Pointer to a loaded image.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_NOT_SUPPORTED [431]	Image format is nor supported.

Module

IHead Unit [1201]

7.1.5.2.1.4 IHeadSaveImageToFile Function

Saves image to the file in NIST IHead [1201] format.

C++

```
NResult N_API IHeadSaveImageToFile(HNImage hImage, const NChar * szFileName);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
const NChar * szFileName	[in] Points to string that specifies file name.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. It is returned when hImage or szFileName is NULL [1075].

Remarks

This is a low-level function and can be changed in future version of the library.

Notes

This function is used if compiling for platform without ANSI versions of the functions support.

Module

IHead Unit [1201]

7.1.5.2.1.5 IHeadSaveImageToMemory Function

Saves image to the memory buffer in NIST IHead [1201] format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API IHeadSaveImageToMemory(HNImage hImage, void * * ppBuffer, NSizeType *
pBufferSize);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
void * * ppBuffer	[out] Pointer to void * that receives pointer to allocated memory buffer.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. It is returned when *hImage*, *ppBuffer* or *pBufferSize* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. It is returned when there was not enough memory to allocate memory buffer.

Remarks

This is a low-level function and can be changed in future version of the library.

Memory buffer allocated by the function must be deallocated using NFree [506] function when it is no longer needed.

Module

IHead Unit [1201]

7.1.5.2.1.6 IHeadSaveImageToStream Function

Saves an image to specified memory stream.

C++

```
NResult N_API IHeadSaveImageToStream(HNImage hImage, HNStream hStream);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image to save to stream.
HNStream hStream	[out] Handle to HNStream where image will be written.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_NOT_SUPPORTED [431]	Image format is nor supported.

Module

IHead Unit [1201]

7.1.5.2.2 Files

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.2.2.1 IHead.h

Header file for IHead [1201] Module. Provides functionality for loading and saving images in NIST IHead [1201] format.

Functions

	Name	Description
	IHeadLoadImageFromFile [1201]	Loads image from NIST IHead [1201] file.
	IHeadLoadImageFromMemory [1202]	Loads image from memory buffer containing NIST IHead [1201] file.
	IHeadLoadImageFromStream [1203]	Loads image data from memory stream.
	IHeadSaveImageToFile [1203]	Saves image to the file in NIST IHead [1201] format.
	IHeadSaveImageToMemory [1203]	Saves image to the memory buffer in NIST IHead [1201] format.
	IHeadSaveImageToStream [1204]	Saves an image to specified memory stream.

Module

IHead Unit [1201]

7.1.5.3 Jpeg Unit

Provides functionality for loading and saving images in JPEG format.

Files

Name	Description
Jpeg.h [1212]	Header file for Jpeg module. Provides functionality for loading and saving images in JPEG format.

Functions

	Name	Description
	JpegInfoGetQuality [1206]	Retrieves a NInt [1006] value which defines the quality of a specified JPEG image.
	JpegInfoIsLossless [1206]	Checks if a JPEG image is lossless.
	JpegInfoSetLossless [1206]	Defines if a JPEG image is lossless.
	JpegInfoSetQuality [1207]	Sets the quality value for a JPEG image.
	JpegLoadImageFromFile [1207]	Loads image from JPEG file.
	JpegLoadImageFromMemory [1208]	Loads image from the memory buffer containing JPEG file.
	JpegLoadImageFromStream [1208]	Loads image from the stream containing JPEG file.
	JpegSaveImageToFile [1209]	Saves image to file in JPEG format.
	JpegSaveImageToMemory [1209]	Saves image to the memory buffer in JPEG format.
	JpegSaveImageToStream [1210]	Saves image to the stream in JPEG format.
	LosslessJpegSaveImageToFile [1210]	Saves specified by HNImage [1276] handle image to file.
	LosslessJpegSaveImageToMemory [1211]	Saves specified lossless JPEG image to memory.
	LosslessJpegSaveImageToStream [1211]	Saves specified lossless JPEG image to memory stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Macros

Name	Description
JPEG_DEFAULT_QUALITY [1212]	Specifies default JPEG quality.

7.1.5.3.1 Functions

7.1.5.3.1.1 JpegInfoGetQuality Function

Retrieves a NInt [1006] value which defines the quality of a specified JPEG image.

C++

```
NResult N_API JpegInfoGetQuality(HJpegInfo hInfo, NInt * pValue);
```

Parameters

Parameters	Description
HJpegInfo hInfo	[in] A handle to JpegInfo.
NInt * pValue	[out] A pointer to the quality value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Jpeg Unit [1205]

7.1.5.3.1.2 JpegInfoIsLossless Function

Checks if a JPEG image is lossless.

C++

```
NResult N_API JpegInfoIsLossless(HJpegInfo hInfo, NBool * pValue);
```

Parameters

Parameters	Description
HJpegInfo hInfo	[in] A handle to the JpegInfo.
NBool * pValue	[out] A pointer to a NBool [1004] value. NTrue [1064] if the image is lossless; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Jpeg Unit [1205]

7.1.5.3.1.3 JpegInfoSetLossless Function

Defines if a JPEG image is lossless.

C++

```
NResult N_API JpegInfoSetLossless(HJpegInfo hInfo, NBool value);
```

Parameters

Parameters	Description
HJpegInfo hInfo	[in] A handle to the JpegInfo.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBool value	[in] A NBool [1004] value to be set. NTrue [1064] for lossless; else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Jpeg Unit [1205]

7.1.5.3.1.4 JpegInfoSetQuality Function

Sets the quality value for a JPEG image.

C++

```
NResult N_API JpegInfoSetQuality(HJpegInfo hInfo, NInt value);
```

Parameters

Parameters	Description
HJpegInfo hInfo	[in] A handle to the JpegInfo.
NInt value	[in] NInt [1006] to be set as quality value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Jpeg Unit [1205]

7.1.5.3.1.5 JpegLoadImageFromFile Function

Loads image from JPEG file.

C++

```
NResult N_API JpegLoadImageFromFile(const NChar * szFileName, HNImage * phImage);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Points to string that specifies file name.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *szFileName* or *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned.This error code is returned when format of file specified by *szFileName* is invalid.

Module

Jpeg Unit [1205]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.3.1.6 JpegLoadImageFromMemory Function

Loads image from the memory buffer containing JPEG file.

C++

```
NResult N_API JpegLoadImageFromMemory(const void * pBuffer, NSizeType bufferSize, HNImage *
phImage);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer.
NSizeType bufferSize	[in] Size of memory buffer.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pBuffer* is NULL [1075] and *bufferSize* is not equal to zero. - or - *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned.This error code is returned when format of file contained in buffer specified by *pBuffer* is invalid.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Jpeg Unit [1205]

7.1.5.3.1.7 JpegLoadImageFromStream Function

Loads image from the stream containing JPEG file.

C++

```
NResult N_API JpegLoadImageFromStream(HNStream hStream, HNImage * phImage);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to NStream [660].
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hStream* or *phImage* is NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Jpeg Unit [1205]

7.1.5.3.1.8 JpegSaveImageToFile Function

Saves image to file in JPEG format.

C++

```
NResult N_API JpegSaveImageToFile(HNImage hImage, NInt quality, const NChar * szFileName);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
NInt quality	[in] Specifies quality of JPEG compression.
const NChar * szFileName	[in] Points to string that specifies file name.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *szFileName* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned.This error code is returned when quality is less than 1 or greater than 100.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Jpeg Unit [1205]

7.1.5.3.1.9 JpegSaveImageToMemory Function

Saves image to the memory buffer in JPEG format.

C++

```
NResult N_API JpegSaveImageToMemory(HNImage hImage, NInt quality, void * * ppBuffer,
NSizeType * pBufferSize);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
NInt quality	[in] Specifies quality of JPEG compression.
void * * ppBuffer	[out] Pointer to void * that receives pointer to allocated memory buffer.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage*, *ppBuffer* or *pBufferSize* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned.This error code is returned when *quality* is less than 1 or greater than 100.
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned.This error code is returned when there was not enough memory to allocate memory buffer.

Remarks

This is a low-level function and can be changed in future version of the library.

Memory buffer allocated by this function must be deallocated by NFree [506] function when it is no longer needed.

Module

Jpeg Unit [1205]

7.1.5.3.1.10 JpegSaveImageToStream Function

Saves image to the stream in JPEG format.

C++

```
NResult N_API JpegSaveImageToStream(HNImage hImage, NInt quality, HNStream hStream);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
NInt quality	[in] Specifies quality of JPEG compression.
HNStream hStream	[out] Handle to NStream [660] where image will be written.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage*, *pBuffer* or *pBufferLength* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned.This error code is returned when quality is less than 1 or greater than 100.
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned.This error code is returned when there was not enough memory to allocate memory buffer.

Remarks

This is a low-level function and can be changed in future version of the library.

Memory buffer allocated by this function must be deallocated by NFree [506] function when it is no longer needed.

Module

Jpeg Unit [1205]

7.1.5.3.1.11 LosslessJpegSaveImageToFile Function

Saves specified by HNImage [1276] handle image to file.

C++

```
NResult N_API LosslessJpegSaveImageToFile(HNImage hImage, const NChar * szFileName);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImage hImage	[in] Handle to NImage [1232] object which contains image to be saved to file.
const NChar * szFileName	[in] Zero terminated string which contains file name where Jpeg [1205] image will be saved.

Returns

If the function succeeds, the return value is N_OK [433].

Module

Jpeg Unit [1205]

7.1.5.3.1.12 LosslessJpegSaveImageToMemory Function

Saves specified lossless JPEG image to memory.

C++

```
NResult N_API LosslessJpegSaveImageToMemory(HNImage hImage, void * * ppBuffer, NSizeType *
pBufferSize);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to JPEG image to be saved to memory.
void * * ppBuffer	[in] Pointer to memory buffer to save image.
NSizeType * pBufferSize	[in] Pointer to size of memory buffer where JPEG image will be saved.

Returns

If the function succeeds, the return value is N_OK [433].

Module

Jpeg Unit [1205]

7.1.5.3.1.13 LosslessJpegSaveImageToStream Function

Saves specified lossless JPEG image to memory stream.

C++

```
NResult N_API LosslessJpegSaveImageToStream(HNImage hImage, HNStream hStream);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to JPEG image to be saved to memory.
HNStream hStream	[out] Handle to NStream [660] which will be used to save lossless JPEG image.

Returns

If the function succeeds, the return value is N_OK [433].

Module

Jpeg Unit [1205]

7.1.5.3.2 Macros

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.3.2.1 JPEG_DEFAULT_QUALITY Macro

Specifies default JPEG quality.

C++

```
#define JPEG_DEFAULT_QUALITY 75
```

Module

Jpeg Unit [1205]

7.1.5.3.3 Files

7.1.5.3.3.1 Jpeg.h

Header file for Jpeg [1205] module. Provides functionality for loading and saving images in JPEG format.

Functions

	Name	Description
	JpegInfoGetQuality [1206]	Retrieves a NInt [1006] value which defines the quality of a specified JPEG image.
	JpegInfoIsLossless [1206]	Checks if a JPEG image is lossless.
	JpegInfoSetLossless [1206]	Defines if a JPEG image is lossless.
	JpegInfoSetQuality [1207]	Sets the quality value for a JPEG image.
	JpegLoadImageFromFile [1207]	Loads image from JPEG file.
	JpegLoadImageFromMemory [1208]	Loads image from the memory buffer containing JPEG file.
	JpegLoadImageFromStream [1208]	Loads image from the stream containing JPEG file.
	JpegSaveImageToFile [1209]	Saves image to file in JPEG format.
	JpegSaveImageToMemory [1209]	Saves image to the memory buffer in JPEG format.
	JpegSaveImageToStream [1210]	Saves image to the stream in JPEG format.
	LosslessJpegSaveImageToFile [1210]	Saves specified by HNImage [1276] handle image to file.
	LosslessJpegSaveImageToMemory [1211]	Saves specified lossless JPEG image to memory.
	LosslessJpegSaveImageToStream [1211]	Saves specified lossless JPEG image to memory stream.

Macros

Name	Description
JPEG_DEFAULT_QUALITY [1212]	Specifies default JPEG quality.

Module

Jpeg Unit [1205]

7.1.5.4 Jpeg2K Unit

Provides functionality for loading and saving images in JPEG 2000 format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Files

Name	Description
Jpeg2K.h [1221]	Header file for Jpeg2K Module. Provides functionality for loading and saving images in JPEG 2000 format.

Functions

	Name	Description
	Jpeg2KInfoGetProfile [1213]	Returns Jpeg [1205] 2000 profile.
	Jpeg2KInfoGetRatio [1214]	Returns Jpeg [1205] 2000 image compression ration.
	Jpeg2KInfoSetProfile [1214]	Sets Jpeg [1205] 2000 profile value.
	Jpeg2KInfoSetRatio [1214]	Sets Jpeg [1205] 2000 image compression ration.
	Jpeg2KLoadImageFromFile [1215]	Creates Jpeg [1205] 2K image from JPEG 2000 file.
	Jpeg2KLoadImageFromMemory [1215]	Loads image from the memory buffer containing JPEG 2000 file.
	Jpeg2KLoadImageFromStream [1216]	Loads image from the stream containing JPEG 2000 file.
	Jpeg2KSaveImageToFile [1216]	Saves image to file in JPEG 2000 format.
	Jpeg2KSaveImageToFileWithProfile [1217]	Saves image to file in JPEG 2000 format.
	Jpeg2KSaveImageToMemory [1217]	Saves image to the memory buffer in JPEG 2000 format.
	Jpeg2KSaveImageToMemoryWithProfile [1218]	Saves image to the memory buffer in JPEG 2000 format.
	Jpeg2KSaveImageToStream [1219]	Saves image to the stream in JPEG 2000 format.
	Jpeg2KSaveImageToStreamWithProfile [1219]	Saves image to the stream in JPEG 2000 format.

Macros

Name	Description
JPEG_2K_DEFAULT_RATIO [1221]	Specifies default JPEG 2000 quality.

Structs, Records, Enums

	Name	Description
	Jpeg2KProfile_ [1220]	Enumerates Jpeg [1205] 2000 Profile values.
	Jpeg2KProfile [1220]	Enumerates Jpeg [1205] 2000 Profile values.

7.1.5.4.1 Functions

7.1.5.4.1.1 Jpeg2KInfoGetProfile Function

Returns Jpeg [1205] 2000 profile.

C++

```
NResult N_API Jpeg2KInfoGetProfile(HJpeg2KInfo hInfo, Jpeg2KProfile * pValue);
```

Parameters

Parameters	Description
HJpeg2KInfo hInfo	[in] Handle to Jpeg2KInfo.
Jpeg2KProfile * pValue	[out] Pointer to Jpeg2KProfile [1220].

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

Jpeg2K Unit [1212]

7.1.5.4.1.2 Jpeg2KInfoGetRatio Function

Returns Jpeg [1205] 2000 image compression ration.

C++

```
NResult N_API Jpeg2KInfoGetRatio(HJpeg2KInfo hInfo, NFloat * pValue);
```

Parameters

Parameters	Description
HJpeg2KInfo hInfo	[in] Handle to Jpeg2KInfo.
NFloat * pValue	[out] Jpeg [1205] 2000 image compression ratio.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

Jpeg2K Unit [1212]

7.1.5.4.1.3 Jpeg2KInfoSetProfile Function

Sets Jpeg [1205] 2000 profile value.

C++

```
NResult N_API Jpeg2KInfoSetProfile(HJpeg2KInfo hInfo, Jpeg2KProfile value);
```

Parameters

Parameters	Description
HJpeg2KInfo hInfo	[in] Handle to Jpeg2KInfo.
Jpeg2KProfile value	[in] Jpeg2KProfile [1220] value to set.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Notes

When one of `Jpeg2KProfile` [1220] values are used (except `jpeg2kpNone`), then JPEG_2K_DEFAULT_RATIO [1221] parameter value is ignored.

Module

Jpeg2K Unit [1212]

7.1.5.4.1.4 Jpeg2KInfoSetRatio Function

Sets Jpeg [1205] 2000 image compression ration.

C++

```
NResult N_API Jpeg2KInfoSetRatio(HJpeg2KInfo hInfo, NFloat value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HJpeg2KInfo hInfo	[in] Handle to Jpeg2KInfo.
NFloat value	[in] Jpeg [1205] 2000 image compression ratio to set.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

Jpeg2K Unit [1212]

7.1.5.4.1.5 Jpeg2KLoadImageFromFile Function

Creates Jpeg [1205] 2K image from JPEG 2000 file.

C++

```
NResult N_API Jpeg2KLoadImageFromFile(const NChar * szFileName, HNImage * phImage);
```

Parameters

Parameters	Description
const NChar * szFileName	[out] Points to string that specifies file name.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *szFileName* or *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file specified by *szFileName* is invalid.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Jpeg2K Unit [1212]

7.1.5.4.1.6 Jpeg2KLoadImageFromMemory Function

Loads image from the memory buffer containing JPEG 2000 file.

C++

```
NResult N_API Jpeg2KLoadImageFromMemory(const void * pBuffer, NSizeType bufferSize, HNImage
* phImage);
```

Parameters

Parameters	Description
const void * pBuffer	[out] Pointer to memory buffer.
NSizeType bufferSize	[in] Size of memory buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pBuffer* is NULL [1075] and *bufferSize* is not equal to zero or *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file contained in memory buffer specified by *pBuffer* is invalid.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Jpeg2K Unit [1212]

7.1.5.4.1.7 Jpeg2KLoadImageFromStream Function

Loads image from the stream containing JPEG 2000 file.

C++

```
NResult N_API Jpeg2KLoadImageFromStream(HNStream hStream, HNImage * phImage);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to HNStream.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hStream* or *phImage* is NULL [1075].

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Jpeg2K Unit [1212]

7.1.5.4.1.8 Jpeg2KSaveImageToFile Function

Saves image to file in JPEG 2000 format.

C++

```
NResult N_API Jpeg2KSaveImageToFile(HNImage hImage, NFloat ratio, const NChar * szFileName);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
NFloat ratio	[in] Specifies ratio of JPEG 2000 compression (value 0.0f means lossless compression).
const NChar * szFileName	[out] Points to string that specifies file name.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *szFileName* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when ratio is less than 0.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Jpeg2K Unit [1212]

7.1.5.4.1.9 Jpeg2KSaveImageToFileWithProfile Function

Saves image to file in JPEG 2000 format.

C++

```
NResult N_API Jpeg2KSaveImageToFileWithProfile(HNImage hImage, Jpeg2KProfile profile, const
NChar * szFileName);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
Jpeg2KProfile profile	[in] Specifies JPEG 2000 profile compression.
const NChar * szFileName	[in] Points to string that specifies file name.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *szFileName* is NULL [1075].

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Jpeg2K Unit [1212]

7.1.5.4.1.10 Jpeg2KSaveImageToMemory Function

Saves image to the memory buffer in JPEG 2000 format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API Jpeg2KSaveImageToMemory(HNImage hImage, NFloat ratio, void * * ppBuffer,
NSizeType * pBufferSize);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
NFloat ratio	[in] Specifies ratio of JPEG 2000 compression (value 0.0f means lossless compression).
void * * ppBuffer	[out] Pointer to void * that receives pointer to allocated memory buffer.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage*, *ppBuffer* or *pBufferSize* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. This error code is returned when there was not enough memory to allocate memory buffer.
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *ratio* is less than 0.

Remarks

This is a low-level function and can be changed in future version of the library.

Memory buffer allocated by this function must be deallocated by NFree [506] function when it is no longer needed.

Module

Jpeg2K Unit [1212]

7.1.5.4.1.11 Jpeg2KSaveImageToMemoryWithProfile Function

Saves image to the memory buffer in JPEG 2000 format.

C++

```
NResult N_API Jpeg2KSaveImageToMemoryWithProfile(HNImage hImage, Jpeg2KProfile profile,
void * * ppBuffer, NSizeType * pBufferSize);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
Jpeg2KProfile profile	[in] Specifies JPEG 2000 profile compression.
void * * ppBuffer	[out] Pointer to void * that receives pointer to allocated memory buffer.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage*, *ppBuffer* or *pBufferSize* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. This error code is returned when there was not enough memory to allocate memory buffer.

Remarks

This is a low-level function and can be changed in future version of the library.

Memory buffer allocated by this function must be deallocated by NFree [506] function when it is no longer needed.

Module

Jpeg2K Unit [1212]

7.1.5.4.1.12 Jpeg2KSaveImageToStream Function

Saves image to the stream in JPEG 2000 format.

C++

```
NResult N_API Jpeg2KSaveImageToStream(HNImage hImage, NFloat ratio, HNStream hStream);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
NFloat ratio	[in] Specifies ratio of JPEG 2000 compression (value 0.0f means lossless compression).
HNStream hStream	[out] Handle to HNStream where image will be written.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage*, *pBuffer* or *pBufferLength* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. This error code is returned when there was not enough memory to allocate memory buffer.
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when ratio is less than 0.

Remarks

This is a low-level function and can be changed in future version of the library.

Memory buffer allocated by this function must be deallocated by NFree [506] function when it is no longer needed.

Module

Jpeg2K Unit [1212]

7.1.5.4.1.13 Jpeg2KSaveImageToStreamWithProfile Function

Saves image to the stream in JPEG 2000 format.

C++

```
NResult N_API Jpeg2KSaveImageToStreamWithProfile(HNImage hImage, Jpeg2KProfile profile,
HNStream hStream);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
Jpeg2KProfile profile	[in] Specifies JPEG 2000 profile compression.
HNStream hStream	[out] Handle to HNStream where image will be written.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage*, *pBuffer* or *pBufferLength* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. This error code is returned when there was not enough memory to allocate memory buffer.

Remarks

This is a low-level function and can be changed in future version of the library.

Memory buffer allocated by this function must be deallocated by NFree [506] function when it is no longer needed.

Module

Jpeg2K Unit [1212]

7.1.5.4.2 Structs, Records, Enums

7.1.5.4.2.1 Jpeg2KProfile Enumeration

Enumerates Jpeg [1205] 2000 Profile values.

C++

```
typedef enum Jpeg2KProfile_ {
  jpeg2kpNone = 0,
  jpeg2kpFingerprint1000Ppi = 1000,
  jpeg2kpFaceLossy = 2000,
  jpeg2kpFaceLossless = 2001
} Jpeg2KProfile;
```

Members

Members	Description
jpeg2kpNone = 0	Default JPEG profile.
jpeg2kpFingerprint1000Ppi = 1000	MTR 04B0000022 (Mitre Technical Report), Margaret Lepley, Profile for 1000ppi Fingerprint Compression, Version 1.1, April 2004.
jpeg2kpFaceLossy = 2000	ANSI/NIST-ITL 1-2007
jpeg2kpFaceLossless = 2001	ANSI/NIST-ITL 1-2007

Module

Jpeg2K Unit [1212]

7.1.5.4.3 Macros

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.4.3.1 JPEG_2K_DEFAULT_RATIO Macro

Specifies default JPEG 2000 quality.

C++

```
#define JPEG_2K_DEFAULT_RATIO 10.0f
```

Notes

This parameter controls compression ratio when saving images to a file. E.g., when 10 is specified compressed image will have 10:1 size ratio compared to that of the uncompressed file. If lossless image is required, then 0 for JPEG_2K_DEFAULT_RATIO parameter should be specified.

Compression ratio can be set using `Jpeg2KInfoSetRatio` [1214] function.

When one of Jpeg2KProfile [1220] values are used (except jpeg2kpNone), then JPEG_2K_DEFAULT_RATIO parameter value is ignored.

Module

Jpeg2K Unit [1212]

7.1.5.4.4 Files

7.1.5.4.4.1 Jpeg2K.h

Header file for Jpeg2K Module. Provides functionality for loading and saving images in JPEG 2000 format.

Enumerations

	Name	Description
	Jpeg2KProfile_ [1220]	Enumerates Jpeg [1205] 2000 Profile values.
	Jpeg2KProfile [1220]	Enumerates Jpeg [1205] 2000 Profile values.

Functions

	Name	Description
	Jpeg2KInfoGetProfile [1213]	Returns Jpeg [1205] 2000 profile.
	Jpeg2KInfoGetRatio [1214]	Returns Jpeg [1205] 2000 image compression ration.
	Jpeg2KInfoSetProfile [1214]	Sets Jpeg [1205] 2000 profile value.
	Jpeg2KInfoSetRatio [1214]	Sets Jpeg [1205] 2000 image compression ration.
	Jpeg2KLoadImageFromFile [1215]	Creates Jpeg [1205] 2K image from JPEG 2000 file.
	Jpeg2KLoadImageFromMemory [1215]	Loads image from the memory buffer containing JPEG 2000 file.
	Jpeg2KLoadImageFromStream [1216]	Loads image from the stream containing JPEG 2000 file.
	Jpeg2KSaveImageToFile [1216]	Saves image to file in JPEG 2000 format.
	Jpeg2KSaveImageToFileWithProfile [1217]	Saves image to file in JPEG 2000 format.
	Jpeg2KSaveImageToMemory [1217]	Saves image to the memory buffer in JPEG 2000 format.
	Jpeg2KSaveImageToMemoryWithProfile [1218]	Saves image to the memory buffer in JPEG 2000 format.
	Jpeg2KSaveImageToStream [1219]	Saves image to the stream in JPEG 2000 format.
	Jpeg2KSaveImageToStreamWithProfile [1219]	Saves image to the stream in JPEG 2000 format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Macros

Name	Description
JPEG_2K_DEFAULT_RATIO [1221]	Specifies default JPEG 2000 quality.

Module

Jpeg2K Unit [1212]

7.1.5.5 NGeometry Unit

Provides definitions of geometrical structures types.

Files

Name	Description
NGeometry.h [1228]	Header file for the NGeometry module. Provides definitions of geometrical structures types.

Structs, Records, Enums

	Name	Description
	NInterpolationMode_ [1223]	Enumerates interpolation modes.
	NRect_ [1222]	Structure defining a rectangle figure in 2D space.
	NPoint_ [1223]	Structure defining point coordinates in 2D space.
	NSize_ [1223]	Structure defining rectangle size.
	NInterpolationMode [1223]	Enumerates interpolation modes.
	NPointD_ [1224]	Structure defining point coordinates in 2D space.
	NPointF_ [1224]	Structure defining point coordinates in 2D space.
	NRectD_ [1225]	Structure defining a rectangle figure in 2D space.
	NRectF_ [1225]	Structure defining a rectangle figure in 2D space.
	NSizeD_ [1225]	Structure defining rectangle size.
	NSizeF_ [1226]	Structure defining rectangle size.

Types

Name	Description
NPoint [1226]	Structure defining point coordinates in 2D space.
NRect [1226]	Structure defining a rectangle figure in 2D space.
NSize [1226]	Structure defining rectangle size.
NPointD [1227]	Structure defining point coordinates in 2D space.
NPointF [1227]	Structure defining point coordinates in 2D space.
NRectD [1227]	Structure defining a rectangle figure in 2D space.
NRectF [1227]	Structure defining a rectangle figure in 2D space.
NSizeD [1227]	Structure defining rectangle size.
NSizeF [1227]	Structure defining rectangle size.

7.1.5.5.1 Structs, Records, Enums

7.1.5.5.1.1 NRect_ Structure

Structure defining a rectangle figure in 2D space.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
struct NRect_ {
  NInt X;
  NInt Y;
  NInt Width;
  NInt Height;
};
```

Members

Members	Description
NInt X;	Upper left rectangle corner coordinate on x axis.
NInt Y;	Upper left rectangle corner coordinate on y axis.
NInt Width;	Rectangle width.
NInt Height;	Rectangle height.

Module

NGeometry Unit [1222]

7.1.5.5.1.2 NPoint_ Structure

Structure defining point coordinates in 2D space.

C++

```
struct NPoint_ {
  NInt X;
  NInt Y;
};
```

Members

Members	Description
NInt X;	Point coordinate on x axis.
NInt Y;	Point coordinate on y axis.

Module

NGeometry Unit [1222]

7.1.5.5.1.3 NSize_ Structure

Structure defining rectangle size.

C++

```
struct NSize_ {
  NInt Width;
  NInt Height;
};
```

Members

Members	Description
NInt Width;	Width.
NInt Height;	Height.

Module

NGeometry Unit [1222]

7.1.5.5.1.4 NInterpolationMode Enumeration

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Enumerates interpolation modes.

C++

```
typedef enum NInterpolationMode_ {
  nimBilinear = 3,
  nimBicubic = 4,
  nimNearestNeighbor = 5
} NInterpolationMode;
```

Members

Members	Description
nimBilinear = 3	Bilinear interpolation.
nimBicubic = 4	Bicubic interpolation.
nimNearestNeighbor = 5	Nearest neighbor interpolation.

Module

NGeometry Unit [1222]

7.1.5.5.1.5 NPointD_ Structure

Structure defining point coordinates in 2D space.

C++

```
struct NPointD_ {
  NDouble X;
  NDouble Y;
};
```

Members

Members	Description
NDouble X;	Point coordinate on x axis.
NDouble Y;	Point coordinate on y axis.

Module

NGeometry Unit [1222]

7.1.5.5.1.6 NPointF_ Structure

Structure defining point coordinates in 2D space.

C++

```
struct NPointF_ {
  NFloat X;
  NFloat Y;
};
```

Members

Members	Description
NFloat X;	Point coordinate on x axis.
NFloat Y;	Point coordinate on y axis.

Module

NGeometry Unit [1222]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.5.1.7 NRectD_ Structure

Structure defining a rectangle figure in 2D space.

C++

```
struct NRectD_ {
  NDouble X;
  NDouble Y;
  NDouble Width;
  NDouble Height;
};
```

Members

Members	Description
NDouble X;	Upper left rectangle corner coordinate on x axis.
NDouble Y;	Upper left rectangle corner coordinate on y axis.
NDouble Width;	Rectangle width.
NDouble Height;	Rectangle height.

Module

NGeometry Unit [1222]

7.1.5.5.1.8 NRectF_ Structure

Structure defining a rectangle figure in 2D space.

C++

```
struct NRectF_ {
  NFloat X;
  NFloat Y;
  NFloat Width;
  NFloat Height;
};
```

Members

Members	Description
NFloat X;	Upper left rectangle corner coordinate on x axis.
NFloat Y;	Upper left rectangle corner coordinate on y axis.
NFloat Width;	Rectangle width.
NFloat Height;	Rectangle height.

Module

NGeometry Unit [1222]

7.1.5.5.1.9 NSizeD_ Structure

Structure defining rectangle size.

C++

```
struct NSizeD_ {
  NDouble Width;
  NDouble Height;
};
```

Members

Members	Description
NDouble Width;	Width.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDouble Height;	Height.

Module

NGeometry Unit [1222]

7.1.5.5.1.10 NSizeF_ Structure

Structure defining rectangle size.

C++

```
struct NSizeF_ {
  NFloat Width;
  NFloat Height;
};
```

Members

Members	Description
NFloat Width;	Width.
NFloat Height;	Height.

Module

NGeometry Unit [1222]

7.1.5.5.2 Types

7.1.5.5.2.1 NPoint Type

Structure defining point coordinates in 2D space.

C++

```
typedef struct NPoint_ NPoint;
```

Module

NGeometry Unit [1222]

7.1.5.5.2.2 NRect Type

Structure defining a rectangle figure in 2D space.

C++

```
typedef struct NRect_ NRect;
```

Module

NGeometry Unit [1222]

7.1.5.5.2.3 NSize Type

Structure defining rectangle size.

C++

```
typedef struct NSize_ NSize;
```

Module

NGeometry Unit [1222]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.5.2.4 NPointD Type

Structure defining point coordinates in 2D space.

C++

```
typedef struct NPointD_ NPointD;
```

Module

NGeometry Unit [1222]

7.1.5.5.2.5 NPointF Type

Structure defining point coordinates in 2D space.

C++

```
typedef struct NPointF_ NPointF;
```

Module

NGeometry Unit [1222]

7.1.5.5.2.6 NRectD Type

Structure defining a rectangle figure in 2D space.

C++

```
typedef struct NRectD_ NRectD;
```

Module

NGeometry Unit [1222]

7.1.5.5.2.7 NRectF Type

Structure defining a rectangle figure in 2D space.

C++

```
typedef struct NRectF_ NRectF;
```

Module

NGeometry Unit [1222]

7.1.5.5.2.8 NSizeD Type

Structure defining rectangle size.

C++

```
typedef struct NSizeD_ NSizeD;
```

Module

NGeometry Unit [1222]

7.1.5.5.2.9 NSizeF Type

Structure defining rectangle size.

C++

```
typedef struct NSizeF_ NSizeF;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NGeometry Unit [1222]

7.1.5.5.3 Files

7.1.5.5.3.1 NGeometry.h

Header file for the NGeometry [1222] module. Provides definitions of geometrical structures types.

Enumerations

	Name	Description
	NInterpolationMode_ [1223]	Enumerates interpolation modes.
	NInterpolationMode [1223]	Enumerates interpolation modes.

Module

NGeometry Unit [1222]

Structures

	Name	Description
	NPoint_ [1223]	Structure defining point coordinates in 2D space.
	NPointD_ [1224]	Structure defining point coordinates in 2D space.
	NPointF_ [1224]	Structure defining point coordinates in 2D space.
	NRect_ [1222]	Structure defining a rectangle figure in 2D space.
	NRectD_ [1225]	Structure defining a rectangle figure in 2D space.
	NRectF_ [1225]	Structure defining a rectangle figure in 2D space.
	NSize_ [1223]	Structure defining rectangle size.
	NSizeD_ [1225]	Structure defining rectangle size.
	NSizeF_ [1226]	Structure defining rectangle size.

Types

Name	Description
NPoint [1226]	Structure defining point coordinates in 2D space.
NPointD [1227]	Structure defining point coordinates in 2D space.
NPointF [1227]	Structure defining point coordinates in 2D space.
NRect [1226]	Structure defining a rectangle figure in 2D space.
NRectD [1227]	Structure defining a rectangle figure in 2D space.
NRectF [1227]	Structure defining a rectangle figure in 2D space.
NSize [1226]	Structure defining rectangle size.
NSizeD [1227]	Structure defining rectangle size.
NSizeF [1227]	Structure defining rectangle size.

7.1.5.6 NGrayscaleImage Unit

Provides functionality for managing 8-bit grayscale images.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Files

Name	Description
NGrayscaleImage.h [1232]	Header file for NGrayscaleImage module. Provides functionality for managing 8-bit grayscale images.

Functions

	Name	Description
	NGrayscaleImageGetMaxValue [1229]	Retrieves maximum pixel value for the specified grayscale image.
	NGrayscaleImageGetMinValue [1229]	Retrieves minimum pixel value for the specified grayscale image.
	NGrayscaleImageGetPixel [1230]	Retrieves the value of pixel at the specified coordinates in the specified grayscale image.
	NGrayscaleImageSetMaxValue [1230]	Sets the maximum pixel value for the specified grayscale image.
	NGrayscaleImageSetMinValue [1231]	Sets the minimum pixel value for the specified grayscale image.
	NGrayscaleImageSetPixel [1231]	Gets the value of pixel at the specified coordinates in the specified grayscale image.

7.1.5.6.1 Functions

7.1.5.6.1.1 NGrayscaleImageGetMaxValue Function

Retrieves maximum pixel value for the specified grayscale image.

C++

```
NResult N_API NGrayscaleImageGetMaxValue(HNGrayscaleImage hImage, NByte * pValue);
```

Parameters

Parameters	Description
HNGrayscaleImage hImage	[in] Handle to the image.
NByte * pValue	[out] Pointer to the maximum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NGrayscaleImage Unit [1228]

7.1.5.6.1.2 NGrayscaleImageGetMinValue Function

Retrieves minimum pixel value for the specified grayscale image.

C++

```
NResult N_API NGrayscaleImageGetMinValue(HNGrayscaleImage hImage, NByte * pValue);
```

Parameters

Parameters	Description
HNGrayscaleImage hImage	[in] Handle to the image.
NByte * pValue	[out] Pointer to the minimum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NGrayscaleImage Unit [1228]

7.1.5.6.1.3 NGrayscaleImageGetPixel Function

Retrieves the value of pixel at the specified coordinates in the specified grayscale image.

C++

```
NResult N_API NGrayscaleImageGetPixel(HNGrayscaleImage hImage, NUInt x, NUInt y, NByte *
pValue);
```

Parameters

Parameters	Description
HNGrayscaleImage hImage	[in] A handle to the image file.
NUInt x	[in] Specifies x-coordinate of the pixel.
NUInt y	[in] Specifies y-coordinate of the pixel.
NByte * pValue	[out] Pointer to NByte [1005] that receives pixel value.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *pValue [870]* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when x is greater than or equal to image width. - or - y is greater than or equal to image height.
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when image pixel format is not equal to *npfRgb [1385]*.

Module

NGrayscaleImage Unit [1228]

7.1.5.6.1.4 NGrayscaleImageSetMaxValue Function

Sets the maximum pixel value for the specified grayscale image.

C++

```
NResult N_API NGrayscaleImageSetMaxValue(HNGrayscaleImage hImage, NByte value);
```

Parameters

Parameters	Description
HNGrayscaleImage hImage	[in] Handle to the image.
NByte value	[in] Maximum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NGrayscaleImage Unit [1228]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.6.1.5 NGrayscaleImageSetMinValue Function

Sets the minimum pixel value for the specified grayscale image.

C++

```
NResult N_API NGrayscaleImageSetMinValue(HNGrayscaleImage hImage, NByte value);
```

Parameters

Parameters	Description
HNGrayscaleImage hImage	[in] Handle to the image.
NByte value	[in] Minimum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NGrayscaleImage Unit [1228]

7.1.5.6.1.6 NGrayscaleImageSetPixel Function

Gets the value of pixel at the specified coordinates in the specified grayscale image.

C++

```
NResult N_API NGrayscaleImageSetPixel(HNGrayscaleImage hImage, NUInt x, NUInt y, NByte
value);
```

Parameters

Parameters	Description
HNGrayscaleImage hImage	[in] A handle to the image file.
NUInt x	[in] Specifies x-coordinate of the pixel.
NUInt y	[in] Specifies y-coordinate of the pixel.
NByte value	[in] NByte [1005] that specifies new pixel value.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *pValue [870]* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when x is greater than or equal to image width. - or - y is greater than or equal to image height.
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when image pixel format is not equal to *npfRgb [1385]*.

Module

NGrayscaleImage Unit [1228]

7.1.5.6.2 Files

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.6.2.1 NGrayscaleImage.h

Header file for NGrayscaleImage [1228] module. Provides functionality for managing 8-bit grayscale images.

Functions

	Name	Description
	NGrayscaleImageGetMaxValue [1229]	Retrieves maximum pixel value for the specified grayscale image.
	NGrayscaleImageGetMinValue [1229]	Retrieves minimum pixel value for the specified grayscale image.
	NGrayscaleImageGetPixel [1230]	Retrieves the value of pixel at the specified coordinates in the specified grayscale image.
	NGrayscaleImageSetMaxValue [1230]	Sets the maximum pixel value for the specified grayscale image.
	NGrayscaleImageSetMinValue [1231]	Sets the minimum pixel value for the specified grayscale image.
	NGrayscaleImageSetPixel [1231]	Gets the value of pixel at the specified coordinates in the specified grayscale image.

Module

NGrayscaleImage Unit [1228]

7.1.5.7 NImage Unit

Provides functionality for managing images.

Files

Name	Description
NImage.h [1291]	Header file for NImage module. Provides functionality for managing images.

Functions

	Name	Description
	NImageCopy [1235]	Copies specified rectangular area of the NImage to other NImage object.
	NImageCopyTo [1236]	Copies specified rectangular area of the NImage to other NImage object.
	NImageCopyData [1236]	Copies pixels from one memory buffer to another.
	NImageCopyDataPlanes [1237]	Copies NImage planes from one memory buffer to another.
	NImageCopyDataPlanesEx [1238]	Copies NImage planes from one memory buffer to another.
	NImageCopyFromData [1239]	Copies pixels to specified NImage from memory buffer.
	NImageCopyFromDataPart [1240]	Copies part of pixels to specified NImage from memory buffer.
	NImageCopyFromDataPlanesPart [1241]	Copies part of planes of specified NImage from memory buffer.
	NImageCopyFromDataPlanesPartEx [1242]	Copies part of planes of specified NImage from memory buffer.
	NImageCreateFromFileEx [1242]	Creates (loads) an image from file with specified format.
	NImageCopyToData [1243]	Copies pixels of specified NImage to memory buffer.
	NImageCopyToDataPart [1244]	Copies part of pixels of specified NImage to memory buffer.
	NImageCopyToDataPlanesPart [1245]	Copies part of planes of specified NImage to memory buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NImageCopyToDataPlanesPartEx [1246]	Copies part of planes of specified NImage to memory buffer.
	NImageCreate [1247]	Creates an image with specified pixel format, size, stride and resolution.
	NImageCreateEx [1248]	Creates an image with specified pixel format, size, stride and resolution.
	NImageCrop [1248]	Crops an image.
	NImageCreateFromData [1249]	Creates an image with specified pixel format, size, stride and resolution and copies specified pixels to it.
	NImageFlipDiagonally [1250]	Flips image diagonally.
	NImageCreateFromDataEx [1251]	Creates an image with specified pixel format, size and stride from specified memory block.
	NImageCreateFromDataPart [1251]	Creates an image with specified pixel format, size, stride and resolution to specified source rectangular area.
	NImageCreateFromDataPartEx [1252]	Creates an image with specified pixel format, size and stride source rectangular area.
	NImageCreateFromMemory [1253]	Creates NImage from memory buffer.
	NImageFlipHorizontally [1254]	Flips image horizontally.
	NImageCreateFromDataPlanesPart [1254]	Creates an image with specified plane format, size, stride and resolution to specified source rectangular area.
	NImageCreateFromStream [1255]	Creates (loads) an image from stream with specified format.
	NImageFlipVertically [1256]	Flips image vertically.
	NImageCreateFromFileExN [1256]	Creates (loads) an image from file with specified format.
	NImageCreateFromImageEx [1257]	Creates an image from specified image with specified pixel format, stride and resolution.
	NImageGetHeight [1257]	Retrieves height of the image.
	NImageCreateFromImageEx2 [1258]	Creates an image from specified image with specified pixel format and stride.
	NImageCreateFromImagePartEx [1258]	Creates an image from other image with specified pixel format, size and stride.
	NImageGetHorzResolution [1259]	Retrieves horizontal resolution of the image.
	NImageCreateFromImagePartEx2 [1260]	Creates an image from other image with specified pixel format, size and stride.
	NImageCreateWrapperForImagePartEx2 [1260]	Creates an image wrapper for a part of specified NImage with specified size.
	NImageCreateWrapper [1261]	Creates an image wrapper for specified image pixels with specified pixel format, size, stride and resolution.
	NImageCreateWrapperEx [1262]	Creates an image wrapper for specified image pixels with specified pixel format, size, stride and resolution.
	NImageGetStride [1263]	Retrieves stride (size of one row) of the image.
	NImageCreateWrapperForPart [1264]	Creates an image wrapper for specified image pixels with specified pixel format, size, stride and resolution.
	NImageGetVertResolution [1265]	Retrieves vertical resolution of the image.
	NImageCreateWrapperForPartEx [1266]	Creates an image wrapper for specified image pixels with specified pixel format, size, stride and resolution.
	NImageGetWidth [1267]	Retrieves width of the image.
	NImageCreateWrapperForPlanesPart [1267]	Creates an image wrapper for specified image planes with specified pixel format, size, stride and resolution.
	NImageRotateFlip [1268]	Rotates image using rotation type.
	NImageExportPalette [1268]	Exports the palette from the specified image.
	NImageGetResolutionIsAspectRatio [1269]	Gets NImage resolution aspect ratio.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NImageSaveToFileEx [1269]	Saves the image to the file of specified format.
	NImageSaveToMemory [1270]	Saves the image to memory of specified format.
	NImageGetImageSize [1271]	Retrieves size of memory block containing pixels of the image.
	NImageSaveToStream [1271]	Saves the image of specified format to stream
	NImageSetHorzResolution [1272]	Sets horizontal resolution for specified NImage.
	NImageGetIndex [1272]	Retrieves the palette entry index of the specified pixel.
	NImageSetResolutionIsAspectRatio [1272]	Sets NImage resolution aspect ratio.
	NImageGetInfo [1273]	Retrieves a NImageInfo object for the specified image.
	NImageSetVertResolution [1273]	Sets vertical resolution for specified NImage.
	NImageGetMaxValue [1273]	Retrieves the maximum pixel value for the specified image.
	NImageToGrayscale [1274]	Converts NImage to NGrayscaleImage [1228].
	NImageGetMinValue [1274]	Retrieves the minimum pixel value for the specified image.
	NImageToMonochrome [1274]	Converts NImage to monochrome image.
	NImageGetPalette [1275]	Retrieves the palette of the specified image.
	NImageToRgb [1275]	Converts NImage to Rgb image.
	NImageGetPaletteEntry [1275]	Retrieves the value of the specified palette entry.
	HNImage [1276]	Handle to image.
	NImageGetPaletteLength [1276]	Retrieves the length of the specified image's palette.
	NImageGetPalettePtr [1276]	Retrieves a pointer to the specified image's palette.
	NImageGetPaletteSize [1276]	Retrieves the size of the specified image's palette.
	NImageGetPixel [1277]	Retrieves the specified pixel of the specified image.
	NImageGetPixelFormat [1277]	Retrieves pixel format of the image.
	NImageGetPixelsN [1278]	Retrieves pointer to memory block containing pixels of the image.
	NImageGetPixelsPtr [1278]	Retrieves a pointer to the pixels of the specified image.
	NImageGetPlane [1278]	Retrieves a pointer to the specified image data plane.
	NImageGetPlaneCount [1279]	Retrieves a pointer to the plane count of the specified image.
	NImageGetPlanePtr [1279]	Gets a pointer to the specified image data plane.
	NImageGetPlanePtrs [1279]	Retrieves an array of pointer to the planes of the specified image.
	NImageGetPlanes [1280]	Retrieves an array of pointer to the buffers containing planes of the specified image.
	NImageGetPlaneSize [1280]	Retrieves a pointer to the plane size of the specified image.
	NImageSetIndex [1281]	Sets the palette entry index for the specified pixel.
	NImageSetMaxValue [1281]	Sets the maximum pixel value for the specified image.
	NImageSetMinValue [1281]	Sets the minimum pixel value for the specified image.
	NImageSetPalette [1282]	Sets the specified image's palette to the specified value.
	NImageSetPaletteEntry [1282]	Sets the value of the specified palette entry to the specified value.
	NImageSetPaletteN [1282]	Sets the specified image's palette to the specified value.
	NImageSetPixel [1283]	Sets specified pixel's value to the specified value.

Macros

Name	Description
NI_ALL_DST [1284]	For internal use.
NI_ALL_DST_AND_SRC [1284]	For internal use.
NI_ALL_SRC [1284]	For internal use.
NI_DST_ALPHA_CHANNEL_FIRST [1284]	Copies Nimage using Rgb Alpha channel first method (ARGB color space).
NI_DST_BOTTOM_TO_TOP [1284]	Copies NImage changing pixels order bottom to top (bottom lines become top and vice versa).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NI_DST_SWAP_CHANNELS [1285]	Copies NImage swapping RGB channels (e.g. RGB to BRG).
NI_READ_UNKNOWN_IMAGE [1285]	Flag specifying whether unknown images should be read.
NI_SRC_ALPHA_CHANNEL_FIRST [1285]	Creates Nimage using Rgb Alpha channel first method (ARGB color space).
NI_SRC_BOTTOM_TO_TOP [1285]	Creates NImage changing pixels order bottom to top (bottom lines become top and vice versa).
NI_SRC_SWAP_CHANNELS [1285]	Creates NImage swapping RGB channels (e.g. RGB to BRG).
NI_WRITE_UNKNOWN_IMAGE [1285]	Flag specifying whether unknown images should be written.
NImageClone [1286]	Creates a copy of the specified image.
NImageCreateFromFileA [1286]	Creates (loads) an image from file with specified format.
NImageCreateFromFileEx [1286]	Picks a Unicode (W) or ANSCII (A) version of the NImageCreateFromFileEx function.
NImageCreateFromFileW [1287]	Creates (loads) an image from file with specified format.
NImageCreateFromImage [1287]	Creates an image from specified image with specified pixel format and stride.
NImageCreateFromImagePart [1287]	Creates an image from other image with specified pixel format, size and stride.
NImageCreateWrapperForImagePart [1288]	Creates an image wrapper for a part of specified NImage with specified size.
NImageFree [1289]	Frees the specified image.
NImageGetPixels [1289]	Retrieves a pointer to the pixels of the specified image.
NImageSaveToFileA [1289]	Saves the image to the file of specified format.
NImageSaveToFileEx [1290]	Picks a Unicode (W) or ANSCII (A) version of the NImageSaveToFileEx function.
NImageSaveToFileW [1290]	Saves the image to the file of specified format.

Structs, Records, Enums

	Name	Description
	NImageRotateFlipType_ [1283]	Enumerates different image rotation types.
	NImageRotateFlipType [1283]	Enumerates different image rotation types.

7.1.5.7.1 Functions

7.1.5.7.1.1 NImageCopy Function

Copies specified rectangular area of the NImage [1232] to other NImage [1232] object.

C++

```
NResult N_API NImageCopy(HNImage hSrcImage, NUInt left, NUInt top, NUInt width, NUInt
height, HNImage hDstImage, NUInt dstLeft, NUInt dstTop);
```

Parameters

Parameters	Description
HNImage hSrcImage	[in] Handle to NImage [1232] object to copy.
NUInt left	[in] The x coordinate of source rectangle.
NUInt top	[in] The y coordinate of source rectangle.
NUInt width	[in] The width of source rectangle.
NUInt height	[in] The height of source rectangle.
HNImage hDstImage	[in] Handle to a destination NImage [1232] object. Specified rectangular area will be copied to this object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt dstLeft	[in] The x coordinate in the destination NImage [1232] object. Copied NImage [1232] will be placed starting by this coordinate.
NUInt dstTop	[in] The y coordinate in the destination NImage [1232] object. Copied NImage [1232] will be placed starting by this coordinate.

Returns

If the function succeeds the return value is N_OK [433]. Otherwise, one of the following error codes are returned:

Return values	Description
N_E_ARGUMENT_NULL [427]	*hSrcImage* or *hDstImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	Specified wrong coordinates or image width or height.

Module

NImage Unit [1232]

7.1.5.7.1.2 NImageCopyTo Function

Copies specified rectangular area of the NImage [1232] to other NImage [1232] object.

C++

```
NResult N_API NImageCopyTo(HNImage hImage, HNImage hDstImage, NUInt dstLeft, NUInt dstTop);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to NImage [1232] object to copy.
HNImage hDstImage	[in] Handle to a destination NImage [1232] object. Specified rectangular area will be copied to this object.
NUInt dstLeft	[in] The x coordinate in the destination NImage [1232] object. Copied NImage [1232] will be placed starting by this coordinate.
NUInt dstTop	[in] The y coordinate in the destination NImage [1232] object. Copied NImage [1232] will be placed starting by this coordinate.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function is used when destination image size is >= to source image size.

Module

NImage Unit [1232]

7.1.5.7.1.3 NImageCopyData Function

Copies pixels from one memory buffer to another.

C++

```
NResult N_API NImageCopyData(NPixelFormat_ srcPixelFormat, NUInt srcWidth, NUInt srcHeight,
NSizeType srcStride, const void * pSrcPixels, NSizeType srcPixelsSize, NUInt srcLeft, NUInt
srcTop, NPixelFormat_ dstPixelFormat, NUInt dstWidth, NUInt dstHeight, NSizeType dstStride,
void * pDstPixels, NSizeType dstPixelsSizes, NUInt dstLeft, NUInt dstTop, NUInt width,
NUInt height, NUInt flags);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NPixelFormat_ srcPixelFormat	[in] The source pixel format.
NUInt srcWidth	[in] The width of the source image data.
NUInt srcHeight	[in] The height of the source image data.
NSizeType srcStride	[in] The stride of the source image data.
const void * pSrcPixels	[in] A pointer to the source memory block.
NSizeType srcPixelsSize	[in] The size of the source memory block.
NUInt srcLeft	[in] The left coordinate of the source image data.
NUInt srcTop	[in] The top coordinate of the source image data.
NPixelFormat_ dstPixelFormat	[in] The destination pixel format.
NUInt dstWidth	[in] The width of the destination image data.
NUInt dstHeight	[in] The height of the destination image data.
NSizeType dstStride	[in] The stride of the destination image data.
void * pDstPixels	[out] A pointer to the destination memory block.
NSizeType dstPixelsSizes	[in] The size of the destination memory block.
NUInt dstLeft	[in] The left coordinate of the destination image data.
NUInt dstTop	[in] The top coordinate of the destination image data.
NUInt width	[in] Image width.
NUInt height	[in] Image height.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_DST_ALPHA_CHANNEL_FIRST [1284], NI_DST_BOTTOM_TO_TOP [1284], NI_DST_SWAP_CHANNELS [1285].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function can copy only image pixels data. If you need to copy the entire NImage [1232] use NImageCopy [1235] function.

Module

NImage Unit [1232]

7.1.5.7.1.4 NImageCopyDataPlanes Function

Copies NImage [1232] planes from one memory buffer to another.

C++

```
NResult N_API NImageCopyDataPlanes(NPixelFormat_ srcPixelFormat, NUInt srcWidth, NUInt
srcHeight, NSizeType srcStride, const void * const * arpSrcPlanes, const NSizeType *
arSrcPlaneSizes, NInt srcPlaneCount, NUInt srcLeft, NUInt srcTop, NPixelFormat_
dstPixelFormat, NUInt dstWidth, NUInt dstHeight, NSizeType dstStride, void * const *
arpDstPlanes, const NSizeType * arDstPlaneSizes, NInt dstPlaneCount, NUInt dstLeft, NUInt
dstTop, NUInt width, NUInt height, NUInt flags);
```

Parameters

Parameters	Description
NPixelFormat_ srcPixelFormat	[in] The source pixel format.
NUInt srcWidth	[in] The width of the source image data.
NUInt srcHeight	[in] The height of the source image data.
NSizeType srcStride	[in] The stride of the source image data.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const void * const * arpSrcPlanes	[in] Auxiliary register pointer to the source memory block.
const NSizeType * arSrcPlaneSizes	[in] Auxiliary register pointer to the source memory block size.
NInt srcPlaneCount	[in] The number of source planes.
NUInt srcLeft	[in] The left coordinate of the source image data.
NUInt srcTop	[in] The top coordinate of the source image data.
NPixelFormat_ dstPixelFormat	[in] The destination pixel format.
NUInt dstWidth	[in] The width of the destination image data.
NUInt dstHeight	[in] The height of the destination image data.
NSizeType dstStride	[in] The stride of the destination image data.
void * const * arpDstPlanes	[out] Auxiliary register pointer to the destination memory block.
const NSizeType * arDstPlaneSizes	[in] Auxiliary register pointer to the destination memory block size.
NInt dstPlaneCount	[in] The number of destination planes.
NUInt dstLeft	[in] The left coordinate of the destination image data.
NUInt dstTop	[in] The top coordinate of the destination image data.
NUInt width	[in] Image width.
NUInt height	[in] Image height.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_DST_ALPHA_CHANNEL_FIRST [1284], NI_DST_BOTTOM_TO_TOP [1284], NI_DST_SWAP_CHANNELS [1285].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function can copy only image plane data. If you need to copy the entire NImage [1232] use NImageCopy [1235] function.

Module

NImage Unit [1232]

7.1.5.7.1.5 NImageCopyDataPlanesEx Function

Copies NImage [1232] planes from one memory buffer to another.

C++

```
NResult N_API NImageCopyDataPlanesEx(NPixelFormat_ srcPixelFormat, HNArray hSrcMinValue,
HNArray hSrcMaxValue, const void * pSrcPalette, NSizeType srcPaletteSize, NInt
srcPaletteLength, NUInt srcWidth, NUInt srcHeight, NSizeType srcStride, const void * const
* arpSrcPlanes, const NSizeType * arSrcPlaneSizes, NInt srcPlaneCount, NUInt srcLeft, NUInt
srcTop, NPixelFormat_ dstPixelFormat, HNArray hDstMinValue, HNArray hDstMaxValue, const
void * pDstPalette, NSizeType dstPaletteSize, NInt dstPaletteLength, NUInt dstWidth, NUInt
dstHeight, NSizeType dstStride, void * const * arpDstPlanes, const NSizeType *
arDstPlaneSizes, NInt dstPlaneCount, NUInt dstLeft, NUInt dstTop, NUInt width, NUInt
height, NUInt flags);
```

Parameters

Parameters	Description
NPixelFormat_ srcPixelFormat	[in] The source pixel format.
HNArray hSrcMinValue	[in] Handle to the minimum source pixel values.
HNArray hSrcMaxValue	[in] Handle to the maximum source pixel values.
const void * pSrcPalette	[in] Pointer to the source image palette.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSizeType srcPaletteSize	[in] Source image palette size.
NInt srcPaletteLength	[in] Source image palette length.
NUInt srcWidth	[in] The width of the source image data.
NUInt srcHeight	[in] The height of the source image data.
NSizeType srcStride	[in] The stride of the source image data.
const void * const * arpSrcPlanes	[in] Auxiliary register pointer to the source memory block.
const NSizeType * arSrcPlaneSizes	[in] Auxiliary register pointer to the source memory block size.
NInt srcPlaneCount	[in] The number of source planes.
NUInt srcLeft	[in] The left coordinate of the source image data.
NUInt srcTop	[in] The top coordinate of the source image data.
NPixelFormat_ dstPixelFormat	[in] The destination pixel format.
HNArray hDstMinValue	[in] Handle to the minimum destination pixel values.
HNArray hDstMaxValue	[in] Handle to the maximum destination pixel values.
const void * pDstPalette	[in] Pointer to the destination image palette.
NSizeType dstPaletteSize	[in] Destination image palette size.
NInt dstPaletteLength	[in] Destination image palette length.
NUInt dstWidth	[in] The width of the destination image data.
NUInt dstHeight	[in] The height of the destination image data.
NSizeType dstStride	[in] The stride of the destination image data.
void * const * arpDstPlanes	[out] Auxiliary register pointer to the destination memory block.
const NSizeType * arDstPlaneSizes	[in] Auxiliary register pointer to the destination memory block size.
NInt dstPlaneCount	[in] The number of destination planes.
NUInt dstLeft	[in] The left coordinate of the destination image data.
NUInt dstTop	[in] The top coordinate of the destination image data.
NUInt width	[in] Image width.
NUInt height	[in] Image height.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_DST_ALPHA_CHANNEL_FIRST [1284], NI_DST_BOTTOM_TO_TOP [1284], NI_DST_SWAP_CHANNELS [1285].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function can copy only image plane data. If you need to copy the entire NImage [1232] use NImageCopy [1235] function.

Module

NImage Unit [1232]

7.1.5.7.1.6 NImageCopyFromData Function

Copies pixels to specified NImage [1232] from memory buffer.

C++

```
NResult N_API NImageCopyFromData(HNImage hImage, NPixelFormat_ srcPixelFormat, NUInt
srcWidth, NUInt srcHeight, NSizeType srcStride, const void * pSrcPixels, NSizeType
srcPixelsSize, NUInt left, NUInt top, NUInt flags);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NPixelFormat_ srcPixelFormat	[in] The source pixel format.
NUInt srcWidth	[in] The width of the source image data.
NUInt srcHeight	[in] The height of the source image data.
NSizeType srcStride	[in] The stride of the source image data.
const void * pSrcPixels	[in] A pointer to the source memory block.
NSizeType srcPixelsSize	[in] The size of the source memory block.
NUInt left	[in] The left coordinate of the image data.
NUInt top	[in] The top coordinate of the image data.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_DST_ALPHA_CHANNEL_FIRST [1284], NI_DST_BOTTOM_TO_TOP [1284], NI_DST_SWAP_CHANNELS [1285].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function can copy only image pixels data. If you need to copy the entire NImage [1232] use NImageCopy [1235] function.

Module

NImage Unit [1232]

7.1.5.7.1.7 NImageCopyFromDataPart Function

Copies part of pixels to specified NImage [1232] from memory buffer.

C++

```
NResult N_API NImageCopyFromDataPart(HNImage hImage, NPixelFormat_ srcPixelFormat, NUInt
srcWidth, NUInt srcHeight, NSizeType srcStride, const void * pSrcPixels, NSizeType
srcPixelsSize, NUInt srcLeft, NUInt srcTop, NUInt left, NUInt top, NUInt width, NUInt
height, NUInt flags);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NPixelFormat_ srcPixelFormat	[in] The source pixel format.
NUInt srcWidth	[in] The width of the source image data.
NUInt srcHeight	[in] The height of the source image data.
NSizeType srcStride	[in] The stride of the source image data.
const void * pSrcPixels	[in] A pointer to the source memory block.
NSizeType srcPixelsSize	[in] The size of the source memory block.
NUInt srcLeft	[in] The left coordinate of the source image data.
NUInt srcTop	[in] The top coordinate of the source image data.
NUInt left	[in] The left coordinate of the image part.
NUInt top	[in] The top coordinate of the image part.
NUInt width	[in] Width of the image part.
NUInt height	[in] Height of the image part.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt flags	[in] Flags controlling function. These flags can be used: NI_DST_ALPHA_CHANNEL_FIRST [1284], NI_DST_BOTTOM_TO_TOP [1284], NI_DST_SWAP_CHANNELS [1285].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function can copy only image pixels data. If you need to copy the entire NImage [1232] use NImageCopy [1235] function.

Module

NImage Unit [1232]

7.1.5.7.1.8 NImageCopyFromDataPlanesPart Function

Copies part of planes of specified NImage [1232] from memory buffer.

C++

```
NResult N_API NImageCopyFromDataPlanesPart(HNImage hImage, NPixelFormat_ srcPixelFormat,
NUInt srcWidth, NUInt srcHeight, NSizeType srcStride, const void * const * arpSrcPlanes,
const NSizeType * arSrcPlaneSizes, NInt srcPlaneCount, NUInt srcLeft, NUInt srcTop, NUInt
left, NUInt top, NUInt width, NUInt height, NUInt flags);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NPixelFormat_ srcPixelFormat	[in] The source pixel format.
NUInt srcWidth	[in] The width of the source image data.
NUInt srcHeight	[in] The height of the source image data.
NSizeType srcStride	[in] The stride of the source image data.
const void * const * arpSrcPlanes	[in] Auxiliary register pointer to the source memory block.
const NSizeType * arSrcPlaneSizes	[in] Auxiliary register pointer to the source memory block size.
NInt srcPlaneCount	[in] The number of the source planes.
NUInt srcLeft	[in] The left coordinate of the source image data.
NUInt srcTop	[in] The top coordinate of the source image data.
NUInt left	[in] Left coordinate of the image part.
NUInt top	[in] Top coordinate of the image part.
NUInt width	[in] Width of the image part.
NUInt height	[in] Height of the image part.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_DST_ALPHA_CHANNEL_FIRST [1284], NI_DST_BOTTOM_TO_TOP [1284], NI_DST_SWAP_CHANNELS [1285].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function can copy only image plane data. If you need to copy the entire NImage [1232] use NImageCopy [1235] function.

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.1.9 NImageCopyFromDataPlanesPartEx Function

Copies part of planes of specified NImage [1232] from memory buffer.

C++

```
NResult N_API NImageCopyFromDataPlanesPartEx(HNImage hImage, NPixelFormat_ srcPixelFormat,
HNArray hSrcMinValue, HNArray hSrcMaxValue, const void * pSrcPalette, NSizeType
srcPaletteSize, NInt srcPaletteLength, NUInt srcWidth, NUInt srcHeight, NSizeType
srcStride, const void * const * arpSrcPlanes, const NSizeType * arSrcPlaneSizes, NInt
srcPlaneCount, NUInt srcLeft, NUInt srcTop, NUInt left, NUInt top, NUInt width, NUInt
height, NUInt flags);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NPixelFormat_ srcPixelFormat	[in] The source pixel format.
HNArray hSrcMinValue	[in] Handle to the minimum source pixel values.
HNArray hSrcMaxValue	[in] Handle to the maximum source pixel values.
const void * pSrcPalette	[in] Pointer to the source image palette.
NSizeType srcPaletteSize	[in] Source image palette size.
NInt srcPaletteLength	[in] Source image palette length.
NUInt srcWidth	[in] The width of the source image data.
NUInt srcHeight	[in] The height of the source image data.
NSizeType srcStride	[in] The stride of the source image data.
const void * const * arpSrcPlanes	[in] Auxiliary register pointer to the source memory block.
const NSizeType * arSrcPlaneSizes	[in] Auxiliary register pointer to the source memory block size.
NInt srcPlaneCount	[in] The number of the source planes.
NUInt srcLeft	[in] The left coordinate of the source image data.
NUInt srcTop	[in] The top coordinate of the source image data.
NUInt left	[in] Left coordinate of the image part.
NUInt top	[in] Top coordinate of the image part.
NUInt width	[in] Width of the image part.
NUInt height	[in] Height of the image part.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_DST_ALPHA_CHANNEL_FIRST [1284], NI_DST_BOTTOM_TO_TOP [1284], NI_DST_SWAP_CHANNELS [1285].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function can copy only image plane data. If you need to copy the entire NImage [1232] use NImageCopy [1235] function.

Module

NImage Unit [1232]

7.1.5.7.1.10 NImageCreateFromFileEx Function

Creates (loads) an image from file with specified format.

C++

```
NResult N_API NImageCreateFromFileEx(const NChar * szFileName, HNImageFormat hImageFormat,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
NUInt flags, HNImageInfo * phInfo, HNImage * phImage);
```

Parameters

Parameters	Description
const NChar * szFileName	[out] Points to string that specifies file name.
HNImageFormat hImageFormat	[in] Handle to the image format of the file. Can be NULL [1075].
NUInt flags	[in] Flags controlling function. Reserved for future use. Must be 0.
HNImageInfo * phInfo	[out] Pointer to NImageInfo.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *szFileName* or *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file specified by *szFileName* is invalid for specified image format.
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* is NULL [1075] and none of supported image formats is registered with file extension of *szFileName*. - or - *hImageFormat* is NULL [1075] and image format registered with file extension of *szFileName* does not support reading. - or - Image format specified by *hImageFormat* does not support reading.

Remarks

If *hImageFormat* is NULL [1075] image format is selected by file extension of *szFileName*.

Created image must be deleted using NObjectFree [542] function.

Module

NImage Unit [1232]

7.1.5.7.1.11 NImageCopyToData Function

Copies pixels of specified NImage [1232] to memory buffer.

C++

```
NResult N_API NImageCopyToData(HNImage hImage, NPixelFormat_ dstPixelFormat, NUInt
dstWidth, NUInt dstHeight, NSizeType dstStride, void * pDstPixels, NSizeType dstPixelsSize,
NUInt dstLeft, NUInt dstTop, NUInt flags);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to NImage [1232] object to copy.
NPixelFormat_ dstPixelFormat	[in] Destination pixel format.
NUInt dstWidth	[in] The width of destination image data.
NUInt dstHeight	[in] The height of destination image data.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSizeType dstStride	[in] The stride of destination image data.
void * pDstPixels	[in] Pointer to memory block where Nimage data is saved.
NSizeType dstPixelsSize	[in] Size of destination pixels size. dstPixelsSize >= dstHeight * dstStride
NUInt dstLeft	[in] The x coordinate in the destination NImage [1232] object. Copied data will be placed starting by this coordinate.
NUInt dstTop	[in] The y coordinate in the destination NImage [1232] object. Copied data will be placed starting by this coordinate.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_DST_ALPHA_CHANNEL_FIRST [1284], NI_DST_BOTTOM_TO_TOP [1284], NI_DST_SWAP_CHANNELS [1285].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function can copy only image pixels data. If you need to copy the entire NImage [1232] use NImageCopy [1235] function.

Module

NImage Unit [1232]

7.1.5.7.1.12 NImageCopyToDataPart Function

Copies part of pixels of specified NImage [1232] to memory buffer.

C++

```
NResult N_API NImageCopyToDataPart(HNImage hImage, NUInt left, NUInt top, NPixelFormat_
dstPixelFormat, NUInt dstWidth, NUInt dstHeight, NSizeType dstStride, void * pDstPixels,
NSizeType dstPixelsSize, NUInt dstLeft, NUInt dstTop, NUInt width, NUInt height, NUInt
flags);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to NImage [1232] object to copy.
NUInt left	[in] Left coordinate of the image part.
NUInt top	[in] Top coordinate of the image part.
NPixelFormat_ dstPixelFormat	[in] Destination pixel format.
NUInt dstWidth	[in] The width of destination image data.
NUInt dstHeight	[in] The height of destination image data.
NSizeType dstStride	[in] The stride of destination image data.
void * pDstPixels	[out] Pointer to memory block where Nimage data is saved.
NSizeType dstPixelsSize	[in] Size of destination pixels size. dstPixelsSize >= dstHeight * dstStride
NUInt dstLeft	[in] The x coordinate in the destination NImage [1232] object. Copied data will be placed starting by this coordinate.
NUInt dstTop	[in] The y coordinate in the destination NImage [1232] object. Copied data will be placed starting by this coordinate.
NUInt width	[in] Width of the image part.
NUInt height	[in] Height of the image part.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_DST_ALPHA_CHANNEL_FIRST [1284], NI_DST_BOTTOM_TO_TOP [1284], NI_DST_SWAP_CHANNELS [1285].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function can copy only image pixels data. If you need to copy the entire NImage [1232] use NImageCopy [1235] function.

Module

NImage Unit [1232]

7.1.5.7.1.13 NImageCopyToDataPlanesPart Function

Copies part of planes of specified NImage [1232] to memory buffer.

C++

```
NResult N_API NImageCopyToDataPlanesPart(HNImage hImage, NUInt left, NUInt top,
NPixelFormat_ dstPixelFormat, NUInt dstWidth, NUInt dstHeight, NSizeType dstStride, void *
const * arpDstPlanes, const NSizeType * arDstPlaneSizes, NInt dstPlaneCount, NUInt dstLeft,
NUInt dstTop, NUInt width, NUInt height, NUInt flags);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NUInt left	[in] Left coordinate of the image part.
NUInt top	[in] Top coordinate of the image part.
NPixelFormat_ dstPixelFormat	[in] Destination pixel format.
NUInt dstWidth	[in] The width of destination image data.
NUInt dstHeight	[in] The height of destination image data.
NSizeType dstStride	[in] The stride of destination image data.
void * const * arpDstPlanes	[out] Auxiliary register pointer to the destination memory block.
const NSizeType * arDstPlaneSizes	[in] Auxiliary register pointer to the destination memory block size.
NInt dstPlaneCount	[in] The number of the destination planes.
NUInt dstLeft	[in] The x coordinate in the destination NImage [1232] object. Copied data will be placed starting by this coordinate.
NUInt dstTop	[in] The y coordinate in the destination NImage [1232] object. Copied data will be placed starting by this coordinate.
NUInt width	[in] Width of the image part.
NUInt height	[in] Height of the image part.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_DST_ALPHA_CHANNEL_FIRST [1284], NI_DST_BOTTOM_TO_TOP [1284], NI_DST_SWAP_CHANNELS [1285].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function can copy only image pixels data. If you need to copy the entire NImage [1232] use NImageCopy [1235] function.

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.1.14 NImageCopyToDataPlanesPartEx Function

Copies part of planes of specified NImage [1232] to memory buffer.

C++

```
NResult N_API NImageCopyToDataPlanesPartEx(HNImage hImage, NUInt left, NUInt top,
NPixelFormat_ dstPixelFormat, HNArray hDstMinValue, HNArray hDstMaxValue, const void *
pDstPalette, NSizeType dstPaletteSize, NInt dstPaletteLength, NUInt dstWidth, NUInt
dstHeight, NSizeType dstStride, void * const * arpDstPlanes, const NSizeType *
arDstPlaneSizes, NInt dstPlaneCount, NUInt dstLeft, NUInt dstTop, NUInt width, NUInt
height, NUInt flags);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NUInt left	[in] Left coordinate of the image part.
NUInt top	[in] Top coordinate of the image part.
NPixelFormat_ dstPixelFormat	[in] Destination pixel format.
HNArray hDstMinValue	[in] Handle to the minimum destination pixel values.
HNArray hDstMaxValue	[in] Handle to the maximum destination pixel values.
const void * pDstPalette	[in] Pointer to the destination image palette.
NSizeType dstPaletteSize	[in] Destination image palette size.
NInt dstPaletteLength	[in] Destination image palette length.
NUInt dstWidth	[in] The width of destination image data.
NUInt dstHeight	[in] The height of destination image data.
NSizeType dstStride	[in] The stride of destination image data.
void * const * arpDstPlanes	[out] Auxiliary register pointer to the destination memory block.
const NSizeType * arDstPlaneSizes	[in] Auxiliary register pointer to the destination memory block size.
NInt dstPlaneCount	[in] The number of the destination planes.
NUInt dstLeft	[in] The x coordinate in the destination NImage [1232] object. Copied data will be placed starting by this coordinate.
NUInt dstTop	[in] The y coordinate in the destination NImage [1232] object. Copied data will be placed starting by this coordinate.
NUInt width	[in] Width of the image part.
NUInt height	[in] Height of the image part.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_DST_ALPHA_CHANNEL_FIRST [1284], NI_DST_BOTTOM_TO_TOP [1284], NI_DST_SWAP_CHANNELS [1285].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

This function can copy only image pixels data. If you need to copy the entire NImage [1232] use NImageCopy [1235] function.

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.1.15 NImageCreate Function

Creates an image with specified pixel format, size, stride and resolution.

C++

```
NResult N_API NImageCreate(NPixelFormat_ pixelFormat, NUInt width, NUInt height, NSizeType
stride, NFloat horzResolution, NFloat vertResolution, HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NUInt width	[in] Specifies width of the image.
NUInt height	[in] Specifies height of the image.
NSizeType stride	[in] Specifies stride of the image. Can be zero.
NFloat horzResolution	[in] Specifies horizontal resolution in pixels per inch of the image.
NFloat vertResolution	[in] Specifies vertical resolution in pixels per inch of the image.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *pixelFormat* has invalid value. - or - *stride* is not zero and is less than minimal value for specified pixel format and width.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *width* or *height* is zero. - or - *horzResolution* or *vertResolution* is less than zero.
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. This error code is returned when there was not enough memory.

Remarks

If *stride* is zero then image stride is automatically calculated. For more information on image stride see NImageGetStride [1263] function.

Created image must be deleted using NObjectFree [542] function.

horzResolution and *vertResolution* can be zero if resolution is not applicable for the image.

Notes

This function is deprecated. Use NImageCreateEx [1248], NImageSetHorzResolution [1272] and NImageSetVertResolution [1273] instead.

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.1.16 NImageCreateEx Function

Creates an image with specified pixel format, size, stride and resolution.

C++

```
NResult N_API NImageCreateEx(NPixelFormat_ pixelFormat, NUInt width, NUInt height,
NSizeType stride, NUInt flags, HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NUInt width	[in] Specifies width of the image.
NUInt height	[in] Specifies height of the image.
NSizeType stride	[in] Specifies stride of the image. Can be zero.
NUInt flags	[in] Flags controlling function. Reserved for future use. Must be 0.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *pixelFormat* has invalid value. - or - *stride* is not zero and is less than minimal value for specified pixel format and width.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *width* or *height* is zero. - or - *horzResolution* or *vertResolution* is less than zero.
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. This error code is returned when there was not enough memory.

Remarks

If *stride* is zero then image stride is automatically calculated. For more information on image stride see NImageGetStride [1263] function.

Created image must be deleted using NObjectFree [542] function.

Module

NImage Unit [1232]

7.1.5.7.1.17 NImageCrop Function

Crops an image.

C++

```
NResult N_API NImageCrop(HNImage hImage, NUInt left, NUInt top, NUInt width, NUInt height,
HNImage * phResultImage);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image that should be cropped.
NUInt left	[in] X-coordinate of the cropping rectangle.
NUInt top	[in] Y-coordinate of the cropping rectangle.
NUInt width	[in] Width of the cropping rectangle.
NUInt height	[in] Height of the cropping rectangle.
HNImage * phResultImage	[out] Pointer to HNImage [1276] that receives handle of the cropped image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *phResultImage* is NULL [1075].

Module

NImage Unit [1232]

7.1.5.7.1.18 NImageCreateFromData Function

Creates an image with specified pixel format, size, stride and resolution and copies specified pixels to it.

C++

```
NResult N_API NImageCreateFromData(NPixelFormat_ pixelFormat, NUInt width, NUInt height,
NSizeType stride, NFloat horzResolution, NFloat vertResolution, NSizeType srcStride, const
void * pSrcPixels, HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NUInt width	[in] Specifies width of the image.
NUInt height	[in] Specifies height of the image.
NSizeType stride	[in] Specifies stride of the image. Can be zero.
NFloat horzResolution	[in] Specifies horizontal resolution in pixels per inch of the image.
NFloat vertResolution	[in] Specifies vertical resolution in pixels per inch of the image.
NSizeType srcStride	[in] Specifies stride of pixels to be copied to the image.
const void * pSrcPixels	[in] Pointer to memory block containing pixels that to be copied to the image.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *pixelFormat* has invalid value. - or - *stride* is not zero and is less than minimal value for specified pixel format and width. - or - *srcStride* is less than minimal value for specified pixel format and width.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pSrcPixels* or *phImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *width* or *height* is zero. - or - *horzResolution* or *vertResolution* is less than zero.

Remarks

If *stride* is zero then image stride is automatically calculated. For more information on image stride see NImageGetStride [1263] function.

Format of memory block *pSrcPixels* points to must be the same as described in NImageGetPixels [1289] function, only stride is equal to *srcStride*.

Created image must be deleted using NObjectFree [542] function.

horzResolution and *vertResolution* can be zero if resolution is not applicable for the image.

Notes

This function is deprecated, use NImageCreateFromDataEx [1251], NImageSetHorzResolution [1272] and NImageSetVertResolution [1273] instead.

Module

NImage Unit [1232]

7.1.5.7.1.19 NImageFlipDiagonally Function

Flips image diagonally.

C++

```
NResult N_API NImageFlipDiagonally(HNImage hImage);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image that should be flipped diagonally.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* is NULL [1075].

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.1.20 NImageCreateFromDataEx Function

Creates an image with specified pixel format, size and stride from specified memory block.

C++

```
NResult N_API NImageCreateFromDataEx(NPixelFormat_ pixelFormat, NUInt width, NUInt height,
NSizeType stride, NSizeType srcStride, const void * pSrcPixels, NSizeType srcPixelsSize,
NUInt flags, HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NUInt width	[in] Specifies width of the image.
NUInt height	[in] Specifies height of the image.
NSizeType stride	[in] Specifies stride of the image. Can be zero.
NSizeType srcStride	[in] Specifies stride of pixels to be copied to the image.
const void * pSrcPixels	[in] Points to memory block containing pixels that to be copied to the image.
NSizeType srcPixelsSize	[in] Size of source pixels size. srcPixelsSize >= height * srcStride.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_SRC_ALPHA_CHANNEL_FIRST [1285], NI_DST_BOTTOM_TO_TOP [1284] and NI_SRC_SWAP_CHANNELS [1285].
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If *stride* is zero then image stride is automatically calculated. For more information on image stride see NImageGetStride [1263] function.

Format of memory block *pSrcPixels* points to must be the same as described in NImageGetPixels [1289] function, only stride is equal to *srcStride*.

Created image must be deleted using NObjectFree [542] function.

Module

NImage Unit [1232]

7.1.5.7.1.21 NImageCreateFromDataPart Function

Creates an image with specified pixel format, size, stride and resolution to specified source rectangular area.

C++

```
NResult N_API NImageCreateFromDataPart(NPixelFormat_ pixelFormat, NUInt width, NUInt
height, NSizeType stride, NFloat horzResolution, NFloat vertResolution, NSizeType
srcStride, const void * pSrcPixels, NUInt left, NUInt top, HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NUInt width	[in] Specifies width of the image.
NUInt height	[in] Specifies height of the image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSizeType stride	[in] Specifies stride of the image. Can be zero.
NFloat horzResolution	[in] Specifies horizontal resolution in pixels per inch of the image.
NFloat vertResolution	[in] Specifies vertical resolution in pixels per inch of the image.
NSizeType srcStride	[in] Specifies stride of the source image. Can be zero.
const void * pSrcPixels	[out] Pixels of source image.
NUInt left	[in] X coordinate of the source image.
NUInt top	[in] Y coordinate of the source image.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *pixelFormat* has invalid value. - or - *stride* is not zero and is less than minimal value for specified pixel format and width.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *width* or *height* is zero. - or - *horzResolution* or *vertResolution* is less than zero.
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. This error code is returned when there was not enough memory.

Remarks

If *stride* is zero then image stride is automatically calculated. For more information on image stride see NImageGetStride [1263] function.

Created image must be deleted using NObjectFree [542] function.

horzResolution and *vertResolution* can be zero if resolution is not applicable for the image.

Notes

This function is deprecated, use NImageCreateFromDataPartEx [1252], NImageSetHorzResolution [1272] and NImageSetVertResolution [1273] instead.

Module

NImage Unit [1232]

7.1.5.7.1.22 NImageCreateFromDataPartEx Function

Creates an image with specified pixel format, size and stride source rectangular area.

C++

```
NResult N_API NImageCreateFromDataPartEx(NPixelFormat_ pixelFormat, NSizeType stride, NUInt
srcWidth, NUInt srcHeight, NSizeType srcStride, const void * pSrcPixels, NSizeType
srcPixelsSize, NUInt srcLeft, NUInt srcTop, NUInt width, NUInt height, NUInt flags, HNImage
* phImage);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NSizeType stride	[in] Specifies stride of the image. Can be zero.
NUInt srcWidth	[in] Specifies width of the source image.
NUInt srcHeight	[in] Specifies height of the source image.
NSizeType srcStride	[in] Specifies stride of the source image.
const void * pSrcPixels	[in] Pixels of source image.
NSizeType srcPixelsSize	[in] Size of source pixels size. srcPixelsSize >= srcHeight * srcStride.
NUInt width	[in] Width of the new image.
NUInt height	[in] Height of the new image.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_SRC_ALPHA_CHANNEL_FIRST [1285], NI_DST_BOTTOM_TO_TOP [1284] and NI_SRC_SWAP_CHANNELS [1285].
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.
left	[in] X coordinate of the new image.
top	[in] Y coordinate of the new image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If *srcStride* is zero then image stride is automatically calculated. For more information on image stride see NImageGetStride [1263] function.

Created image must be deleted using NObjectFree [542] function.

Module

NImage Unit [1232]

7.1.5.7.1.23 NImageCreateFromMemory Function

Creates NImage [1232] from memory buffer.

C++

```
NResult N_API NImageCreateFromMemory(const void * pBuffer, NSizeType bufferSize,
HNImageFormat hImageFormat, NUInt flags, NSizeType * pSize, HNImageInfo * phInfo, HNImage *
phImage);
```

Parameters

Parameters	Description
const void * pBuffer	[out] Pointer to memory buffer where NImage [1232] is saved.
NSizeType bufferSize	[in] Size of memory buffer *pBuffer*.
HNImageFormat hImageFormat	[in] Handle to the image format of image saved in buffer. Can be NULL [1075].
NUInt flags	[in] Flags controlling function. Reserved for future use. Must be 0.
NSizeType * pSize	[out] Pointer to NSizeType [1008] that receives calculated size of created NImage [1232].
HNImageInfo * phInfo	[out] Pointer to NImageInfo.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

Created image must be deleted using NObjectFree [542] function.

Module

NImage Unit [1232]

7.1.5.7.1.24 NImageFlipHorizontally Function

Flips image horizontally.

C++

```
NResult N_API NImageFlipHorizontally(HNImage hImage);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image that should be flipped horizontally.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* is NULL [1075].

Module

NImage Unit [1232]

7.1.5.7.1.25 NImageCreateFromDataPlanesPart Function

Creates an image with specified plane format, size, stride and resolution to specified source rectangular area.

C++

```
NResult N_API NImageCreateFromDataPlanesPart(NPixelFormat_ pixelFormat, NSizeType stride,
NUInt srcWidth, NUInt srcHeight, NSizeType srcStride, const void * const * arpSrcPlanes,
const NSizeType * arSrcPlaneSizes, NInt srcPlaneCount, NUInt srcLeft, NUInt srcTop, NUInt
width, NUInt height, NUInt flags, HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NSizeType stride	[in] Specifies stride of the image. Can be zero.
NUInt srcWidth	[in] The width of the source image data.
NUInt srcHeight	[in] The height of the source image data.
NSizeType srcStride	[in] The stride of the source image data.
const void * const * arpSrcPlanes	[in] Auxiliary register pointer to the source memory block.
const NSizeType * arSrcPlaneSizes	[in] Auxiliary register pointer to the source memory block size.
NInt srcPlaneCount	[in] The number of the source image data planes.
NUInt srcLeft	[in] The left coordinate of the source image data.
NUInt srcTop	[in] The top coordinate of the source image data.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt width	[in] The width of the image.
NUInt height	[in] The height of the image.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_SRC_ALPHA_CHANNEL_FIRST [1285], NI_DST_BOTTOM_TO_TOP [1284] and NI_SRC_SWAP_CHANNELS [1285].
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.26 NImageCreateFromStream Function

Creates (loads) an image from stream with specified format.

C++

```
NResult N_API NImageCreateFromStream(HNStream hStream, HNImageFormat hImageFormat, NUInt flags, HNImageInfo * phInfo, HNImage * phImage);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to the data stream used to load the image.
HNImageFormat hImageFormat	[in] Handle to the image format of the file. Can be NULL [1075].
NUInt flags	[in] Flags controlling function. Reserved for future use. Must be 0.
HNImageInfo * phInfo	[out] Pointer to NImageInfo.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *szFileName* or *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file specified by *szFileName* is invalid for specified image format.
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* is NULL [1075] and none of supported image formats is registered with file extension of *szFileName*. - or - *hImageFormat* is NULL [1075] and image format registered with file extension of *szFileName* does not support reading. - or - Image format specified by *hImageFormat* does not support reading.

Remarks

Created image must be deleted using NObjectFree [542] function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NImage Unit [1232]

7.1.5.7.1.27 NImageFlipVertically Function

Flips image vertically.

C++

```
NResult N_API NImageFlipVertically(HNImage hImage);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image that should be flipped vertically.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* is NULL [1075].

Module

NImage Unit [1232]

7.1.5.7.1.28 NImageCreateFromFileExN Function

Creates (loads) an image from file with specified format.

C++

```
NResult N_API NImageCreateFromFileExN(HNString hFileName, HNImageFormat hImageFormat, NUInt
flags, HNImageInfo * phInfo, HNImage * phImage);
```

Parameters

Parameters	Description
HNString hFileName	[in] String that specifies file name.
HNImageFormat hImageFormat	[in] Handle to the image format of the file. Can be NULL [1075].
NUInt flags	[in] Flags controlling function. Reserved for future use. Must be 0.
HNImageInfo * phInfo	[out] Pointer to NImageInfo.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If *hImageFormat* is NULL [1075] image format is selected by file extension of *szFileName*.

Created image must be deleted using NObjectFree [542] function.

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.1.29 NImageCreateFromImageEx Function

Creates an image from specified image with specified pixel format, stride and resolution.

C++

```
NResult N_API NImageCreateFromImageEx(NPixelFormat_ pixelFormat, NSizeType stride, NFloat horzResolution, NFloat vertResolution, HNImage hSrcImage, HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NSizeType stride	[in] Specifies stride of the image. Can be zero.
NFloat horzResolution	[in] Specifies horizontal resolution in pixels per inch of the image.
NFloat vertResolution	[in] Specifies vertical resolution in pixels per inch of the image.
HNImage hSrcImage	[in] Handle to image used as source for the image.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *pixelFormat* has invalid value. - or - *stride* is not zero and is less than minimal value for specified pixel format and source image width.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hSrcImage* or *phImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *horzResolution* or *vertResolution* is less than zero.

Remarks

If *stride* is zero then image stride is automatically calculated. For more information on image stride see NImageGetStride [1263] function.

Created image must be deleted using NObjectFree [542] function.

horzResolution and *vertResolution* can be zero if resolution is not applicable for the image.

Notes

This function is deprecated, use NImageCreateFromImageEx2 [1258], NImageToGrayscale [1274], NImageToMonochrome [1274] or NImageToRgb [1275] and NImageSetHorzResolution [1272], NImageSetVertResolution [1273] and NImageSetResolutionIsAspectRatio [1272] instead.

Module

NImage Unit [1232]

7.1.5.7.1.30 NImageGetHeight Function

Retrieves height of the image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NImageGetHeight(HNImage hImage, NUInt * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NUInt * pValue	[out] Pointer to NUInt [1009] that receives height of the image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *pValue [870]* is NULL [1075].

Module

NImage Unit [1232]

7.1.5.7.1.31 NImageCreateFromImageEx2 Function

Creates an image from specified image with specified pixel format and stride.

C++

```
NResult N_API NImageCreateFromImageEx2(NPixelFormat_ pixelFormat, NSizeType stride, HNImage
hSrcImage, NUInt flags, HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NSizeType stride	[in] Specifies stride of the image. Can be zero.
HNImage hSrcImage	[in] Handle to image used as source for the image.
NUInt flags	[in] Flags controlling function. Reserved for future use. Must be 0.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If *stride* is zero then image stride is automatically calculated. For more information on image stride see NImageGetStride [1263] function.

Created image must be deleted using NObjectFree [542] function.

Module

NImage Unit [1232]

7.1.5.7.1.32 NImageCreateFromImagePartEx Function

Creates an image from other image with specified pixel format, size and stride.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NImageCreateFromImagePartEx(NPixelFormat_ pixelFormat, NSizeType stride,
NFloat horzResolution, NFloat vertResolution, HNImage hSrcImage, NUInt left, NUInt top,
NUInt width, NUInt height, HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NSizeType stride	[in] Specifies stride of the image. Can be zero.
NFloat horzResolution	[in] Specifies horizontal resolution in pixels per inch of the image.
NFloat vertResolution	[in] Specifies vertical resolution in pixels per inch of the image.
HNImage hSrcImage	[in] Handle to source NImage [1232] object.
NUInt left	[in] X coordinate of the source image.
NUInt top	[in] Y coordinate of the source image.
NUInt width	[in] Specifies width of the image.
NUInt height	[in] Specifies height of the image.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *pixelFormat* has invalid value. - or - *stride* is not zero and is less than minimal value for specified pixel format and source image width.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hSrcImage* or *phImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *horzResolution* or *vertResolution* is less than zero.

Remarks

This function is deprecated, use NImageCreateFromImageEx2 [1258], NImageToGrayscale [1274], NImageToMonochrome [1274] or NImageToRgb [1275] and NImageSetHorzResolution [1272], NImageSetVertResolution [1273] and NImageSetResolutionIsAspectRatio [1272] instead.

Module

NImage Unit [1232]

7.1.5.7.1.33 NImageGetHorzResolution Function

Retrieves horizontal resolution of the image.

C++

```
NResult N_API NImageGetHorzResolution(HNImage hImage, NFloat * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NFloat * pValue	[out] Pointer to NFloat [1005] that receives horizontal resolution in pixels per inch of the image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *pValue [870]* is NULL [1075].

Remarks

Horizontal resolution equal to zero means that it is not applicable for the image.

Module

NImage Unit [1232]

7.1.5.7.1.34 NImageCreateFromImagePartEx2 Function

Creates an image from other image with specified pixel format, size and stride.

C++

```
NResult N_API NImageCreateFromImagePartEx2(NPixelFormat_ pixelFormat, NSizeType stride,
HNImage hSrcImage, NUInt left, NUInt top, NUInt width, NUInt height, NUInt flags, HNImage *
phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NSizeType stride	[in] Specifies stride of the image. Can be zero.
HNImage hSrcImage	[in] Handle to source NImage [1232] object.
NUInt left	[in] X coordinate of the source image.
NUInt top	[in] Y coordinate of the source image.
NUInt width	[in] Specifies width of the image.
NUInt height	[in] Specifies height of the image.
NUInt flags	[in] Flags controlling function. Reserved for future use. Must be 0.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.35 NImageCreateWrapperForImagePartEx2 Function

Creates an image wrapper for a part of specified NImage [1232] with specified size.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NImageCreateWrapperForImagePartEx2(HNImage hSrcImage, NUInt left, NUInt top,
NUInt width, NUInt height, NUInt flags, HNImage * phImage);
```

Parameters

Parameters	Description
HNImage hSrcImage	[in] A handle to the source image.
NUInt left	[in] X coordinate of the image.
NUInt top	[in] Y coordinate of the image.
NUInt width	[in] Specifies width of the image.
NUInt height	[in] Specifies height of the image.
NUInt flags	[in] Flags controlling function.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *pixelFormat* has invalid value. - or - *stride* is less than minimal value for specified pixel format and width.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pixels* or *phImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *width* or *height* is zero. - or - *horzResolution* or *vertResolution* is less than zero.

Remarks

For more information on image stride see NImageGetStride [1263] function.

Format of memory block pixels points to must be the same as described in NImageGetPixels [1289] function.

Created image must be deleted using NObjectFree [542] function.

pixels must not be deleted during lifetime of the image. If ownsPixels is NTrue [1064] then pixels will be automatically deleted with the image.

horzResolution and *vertResolution* can be zero if resolution is not applicable for the image.

Module

NImage Unit [1232]

7.1.5.7.1.36 NImageCreateWrapper Function

Creates an image wrapper for specified image pixels with specified pixel format, size, stride and resolution.

C++

```
NResult N_API NImageCreateWrapper(NPixelFormat_ pixelFormat, NUInt width, NUInt height,
NSizeType stride, NFloat horzResolution, NFloat vertResolution, void * pPixels, NBool
ownsPixels, HNImage * phImage);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NUInt width	[in] Specifies width of the image.
NUInt height	[in] Specifies height of the image.
NSizeType stride	[in] Specifies stride of the image.
NFloat horzResolution	[in] Specifies horizontal resolution in pixels per inch of the image.
NFloat vertResolution	[in] Specifies vertical resolution in pixels per inch of the image.
void * pPixels	[out] Points to memory block containing pixels for the image.
NBool ownsPixels	[in] Specifies whether pixels will be automatically deleted with the image (if set to NTrue [1064]).
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *pixelFormat* has invalid value. - or - *stride* is less than minimal value for specified pixel format and width.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pixels* or *phImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *width* or *height* is zero. - or - *horzResolution* or *vertResolution* is less than zero.

Remarks

For more information on image stride see NImageGetStride [1263] function.

Format of memory block pixels points to must be the same as described in NImageGetPixels [1289] function.

Created image must be deleted using NObjectFree [542] function.

pixels must not be deleted during lifetime of the image. If ownsPixels is NTrue [1064] then pixels will be automatically deleted with the image.

horzResolution and *vertResolution* can be zero if resolution is not applicable for the image.

This function is deprecated, use NImageCreateWrapperEx [1262] instead.

Module

NImage Unit [1232]

7.1.5.7.1.37 NImageCreateWrapperEx Function

Creates an image wrapper for specified image pixels with specified pixel format, size, stride and resolution.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NImageCreateWrapperEx(NPixelFormat_ pixelFormat, NUInt width, NUInt height,
NSizeType stride, void * pPixels, NSizeType pixelsSize, NBool ownsPixels, NUInt flags,
HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NUInt width	[in] Specifies width of the image.
NUInt height	[in] Specifies height of the image.
NSizeType stride	[in] Specifies stride of the image.
void * pPixels	[out] Points to memory block containing pixels for the image.
NSizeType pixelsSize	[in] Specifies pixel size.
NBool ownsPixels	[in] Specifies whether pixels will be automatically deleted with the image (if set to NTrue [1064]).
NUInt flags	[in] Flags controlling function.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *pixelFormat* has invalid value. - or - *stride* is less than minimal value for specified pixel format and width.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pixels* or *phImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *width* or *height* is zero. - or - *horzResolution* or *vertResolution* is less than zero.

Remarks

For more information on image stride see NImageGetStride [1263] function.

Format of memory block pixels points to must be the same as described in NImageGetPixels [1289] function.

Created image must be deleted using NObjectFree [542] function.

pixels must not be deleted during lifetime of the image. If ownsPixels is NTrue [1064] then pixels will be automatically deleted with the image.

horzResolution and *vertResolution* can be zero if resolution is not applicable for the image.

Module

NImage Unit [1232]

7.1.5.7.1.38 NImageGetStride Function

Retrieves stride (size of one row) of the image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NImageGetStride(HNImage hImage, NSizeType * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NSizeType * pValue	[out] Pointer to NSizeType [1008] that receives stride of the image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *pValue [870]* is NULL [1075].

Remarks

Stride (size of one row) of the image depends on image pixel format and width. It cannot be less than value obtained with NPixelFormatGetRowSize [1367] macro with arguments obtained with NImageGetPixelFormat [1277] and NImageGetWidth [1267] functions.

Module

NImage Unit [1232]

7.1.5.7.1.39 NImageCreateWrapperForPart Function

Creates an image wrapper for specified image pixels with specified pixel format, size, stride and resolution.

C++

```
NResult N_API NImageCreateWrapperForPart(NPixelFormat_ pixelFormat, NUInt width, NUInt
height, NSizeType stride, NFloat horzResolution, NFloat vertResolution, void * pPixels,
NUInt left, NUInt top, HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NUInt width	[in] Specifies width of the image.
NUInt height	[in] Specifies height of the image.
NSizeType stride	[in] Specifies stride of the image.
NFloat horzResolution	[in] Specifies horizontal resolution in pixels per inch of the image.
NFloat vertResolution	[in] Specifies vertical resolution in pixels per inch of the image.
void * pPixels	[out] Points to memory block containing pixels for the image.
NUInt left	[in] X coordinate of the source image.
NUInt top	[in] Y coordinate of the source image.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *pixelFormat* has invalid value. - or - *stride* is less than minimal value for specified pixel format and width.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pixels* or *phImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *width* or *height* is zero. - or - *horzResolution* or *vertResolution* is less than zero.

Remarks

For more information on image stride see NImageGetStride [1263] function.

Format of memory block pixels points to must be the same as described in NImageGetPixels [1289] function.

Created image must be deleted using NObjectFree [542] function.

pixels must not be deleted during lifetime of the image. If ownsPixels is NTrue [1064] then pixels will be automatically deleted with the image.

horzResolution and *vertResolution* can be zero if resolution is not applicable for the image.

This function is deprecated, use NImageCreateWrapperForPartEx [1266] instead.

Module

NImage Unit [1232]

7.1.5.7.1.40 NImageGetVertResolution Function

Retrieves vertical resolution of the image.

C++

```
NResult N_API NImageGetVertResolution(HNImage hImage, NFloat * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NFloat * pValue	[out] Pointer to NFloat [1005] that receives vertical resolution in pixels per inch of the image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *pValue [870]* is NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

Vertical resolution equal to zero means that it is not applicable for the image.

Module

NImage Unit [1232]

7.1.5.7.1.41 NImageCreateWrapperForPartEx Function

Creates an image wrapper for specified image pixels with specified pixel format, size, stride and resolution.

C++

```
NResult N_API NImageCreateWrapperForPartEx(NPixelFormat_ pixelFormat, NUInt srcWidth, NUInt
srcHeight, NSizeType stride, void * pPixels, NSizeType pixelsSize, NUInt left, NUInt top,
NUInt width, NUInt height, NUInt flags, HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NUInt srcWidth	[in] Width of the source image.
NUInt srcHeight	[in] Height of the source image.
NSizeType stride	[in] Specifies stride of the image.
void * pPixels	[out] Points to memory block containing pixels for the image.
NSizeType pixelsSize	[in] Specifies pixel size.
NUInt left	[in] X coordinate of the image.
NUInt top	[in] Y coordinate of the image.
NUInt width	[in] Specifies width of the image.
NUInt height	[in] Specifies height of the image.
NUInt flags	[in] Flags controlling function.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *pixelFormat* has invalid value. - or - *stride* is less than minimal value for specified pixel format and width.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pixels* or *phImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *width* or *height* is zero. - or - *horzResolution* or *vertResolution* is less than zero.

Remarks

For more information on image stride see NImageGetStride [1263] function.

Format of memory block pixels points to must be the same as described in NImageGetPixels [1289] function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Created image must be deleted using NObjectFree [542] function.

pixels must not be deleted during lifetime of the image. If ownsPixels is NTrue [1064] then pixels will be automatically deleted with the image.

horzResolution and *vertResolution* can be zero if resolution is not applicable for the image.

Module

NImage Unit [1232]

7.1.5.7.1.42 NImageGetWidth Function

Retrieves width of the image.

C++

```
NResult N_API NImageGetWidth(HNImage hImage, NUInt * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NUInt * pValue	[out] Pointer to NUInt [1009] that receives width of the image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *pValue [870]* is NULL [1075].

Module

NImage Unit [1232]

7.1.5.7.1.43 NImageCreateWrapperForPlanesPart Function

Creates an image wrapper for specified image planes with specified pixel format, size, stride and resolution.

C++

```
NResult N_API NImageCreateWrapperForPlanesPart(NPixelFormat_ pixelFormat, NUInt srcWidth,
NUInt srcHeight, NSizeType stride, void * const * arpPlanes, const NSizeType *
arPlaneSizes, NInt planeCount, NUInt left, NUInt top, NUInt width, NUInt height, NUInt
flags, HNImage * phImage);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NUInt srcWidth	[in] The width of the source image data.
NUInt srcHeight	[in] The height of the source image data.
NSizeType stride	[in] The stride of the image.
void * const * arpPlanes	[in] Auxiliary register pointer to the memory block containing planes for the image.
const NSizeType * arPlaneSizes	[in] Auxiliary register pointer to the memory block size.
NInt planeCount	[in] The number of the image planes.
NUInt left	[in] X coordinate of the source image.
NUInt top	[in] Y coordinate of the source image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt width	[in] Specifies the width of the image.
NUInt height	[in] Specifies the height of the image.
NUInt flags	[in] Flags controlling function. These flags can be used: NI_SRC_ALPHA_CHANNEL_FIRST [1285], NI_DST_BOTTOM_TO_TOP [1284] and NI_SRC_SWAP_CHANNELS [1285].
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.44 NImageRotateFlip Function

Rotates image using rotation type.

C++

```
NResult N_API NImageRotateFlip(HNImage hImage, NImageRotateFlipType rotateFlipType, HNImage
* phResultImage);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image that should be rotated.
NImageRotateFlipType rotateFlipType	[in] One of the rotation type value.
HNImage * phResultImage	[out] Pointer to rotated image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *phResultImage* is NULL [1075].
N_E_ARGUMENT [427]	This error is returned when *rotateFlipType* has an invalid value.

Module

NImage Unit [1232]

7.1.5.7.1.45 NImageExportPalette Function

Exports the palette from the specified image.

C++

```
NResult N_API NImageExportPalette(HNImage hImage, NPixelFormat_ dstPixelFormat, HNArray
hDstMinValue, HNArray hDstMaxValue, NUInt flags, NInt * pPaletteLength, HNBuffer *
phPalette);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NPixelFormat_ dstPixelFormat	[in] The destination pixel format.
HNArray hDstMinValue	[in] Handle to the minimum destination pixel values.
HNArray hDstMaxValue	[in] Handle to the maximum destination pixel values.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt flags	[in] Flags controlling function. These flags can be used: NI_DST_ALPHA_CHANNEL_FIRST [1284], NI_DST_BOTTOM_TO_TOP [1284], NI_DST_SWAP_CHANNELS [1285].
NInt * pPaletteLength	[out] Pointer to the palette length.
HNBuffer * phPalette	[out] Pointer to a handle to NBuffer [313] containing the palette.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.46 NImageGetResolutionIsAspectRatio Function

Gets NImage [1232] resolution aspect ratio.

C++

```
NResult N_API NImageGetResolutionIsAspectRatio(HNImage hImage, NBool * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NBool * pValue	[out] Resolution aspect ratio. NTrue [1064], if resolution aspect ratio is used. NFalse [1041], otherwise.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

NImage [1232] resolution can have 2 modes: physical height and width in pixels or aspect ratio. Picture aspect ratio defines the shape of displayed image. Picture aspect ratio is notated X:Y, where X:Y is the ration of picture width to height. If this function returns NTrue [1064] value it means that NImage [1232] shape is defined using width and height ratio (aspect ratio).

Module

NImage Unit [1232]

7.1.5.7.1.47 NImageSaveToFileEx Function

Saves the image to the file of specified format.

C++

```
NResult N_API NImageSaveToFileEx(HNImage hImage, const NChar * szFileName, HNImageFormat
hImageFormat, HNImageInfo hInfo, NUInt flags);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to NImage [1232] object.
const NChar * szFileName	[in] Points to string that specifies file name.
HNImageFormat hImageFormat	[in] Handle to the image format of the file. Can be NULL [1075].
HNImageInfo hInfo	[in] Handle to NImageInfo.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt flags	[in] Flags controlling function. Reserved for future use. Must be 0.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *szFileName* is NULL [1075].
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* is NULL [1075] and none of supported image formats is registered with file extension of *szFileName*. - or - *hImageFormat* is NULL [1075] and image format registered with file extension of *szFileName* does not support writing. - or - Image format specified by *hImageFormat* does not support writing.

Remarks

If *hImageFormat* is NULL [1075] image format is selected by file extension of *szFileName*.

Module

NImage Unit [1232]

7.1.5.7.1.48 NImageSaveToMemory Function

Saves the image to memory of specified format.

C++

```
NResult N_API NImageSaveToMemory(HNImage hImage, HNImageFormat hImageFormat, HNImageInfo
hInfo, NUInt flags, void * * ppBuffer, NSizeType * pBufferSize);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to NImage [1232] object.
HNImageFormat hImageFormat	[in] Handle to the image format of the file. Can be NULL [1075].
HNImageInfo hInfo	[in] Pointer to NImageInfo.
NUInt flags	[in] Flags controlling function. Reserved for future use. Must be 0.
void * * ppBuffer	[out] Pointer to void that receives pointer to memory block containing saved NImage [1232] object.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.1.49 NImageGetImageSize Function

Retrieves size of memory block containing pixels of the image.

C++

```
NResult N_API NImageGetImageSize(HNImage hImage, NSizeType * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NSizeType * pValue	[out] Pointer to NSizeType [1008] that receives size of memory block containing pixels of the image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *pValue [870]* is NULL [1075].

Remarks

Size of memory block containing image pixels is equal to image height multiplied by image stride. For more information see NImageGetHeight [1257] and NImageGetStride [1263] functions.

Module

NImage Unit [1232]

7.1.5.7.1.50 NImageSaveToStream Function

Saves the image of specified format to stream

C++

```
NResult N_API NImageSaveToStream(HNImage hImage, HNStream hStream, HNImageFormat
hImageFormat, HNImageInfo hInfo, NUInt flags);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to NImage [1232] object.
HNStream hStream	[in] Handle to NStream [660] which will be used to save NImage [1232].
HNImageFormat hImageFormat	[in] Handle to the image format of the file. Can be NULL [1075].
HNImageInfo hInfo	[in] Pointer to NImageInfo.
NUInt flags	[in] Flags controlling function. Reserved for future use. Must be 0.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.1.51 NImageSetHorzResolution Function

Sets horizontal resolution for specified NImage [1232].

C++

```
NResult N_API NImageSetHorzResolution(HNImage hImage, NFloat value);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to NImage [1232].
NFloat value	[in] Horizontal resolution (pixels per inch).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.52 NImageGetIndex Function

Retrieves the palette entry index of the specified pixel.

C++

```
NResult N_API NImageGetIndex(HNImage hImage, NUInt x, NUInt y, NInt * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NUInt x	[in] X-coordinate of the pixel.
NUInt y	[in] Y-coordinate of the pixel.
NInt * pValue	[out] Pointer to the palette entry index.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.53 NImageSetResolutionIsAspectRatio Function

Sets NImage [1232] resolution aspect ratio.

C++

```
NResult N_API NImageSetResolutionIsAspectRatio(HNImage hImage, NBool value);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image to set its resolution aspect ratio .
NBool value	[in] Resolution aspect ratio. NTrue [1064], if resolution aspect ratio is used. NFalse [1041], otherwise.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

NImage [1232] resolution can have 2 modes: physical height and width in pixels or aspect ratio. Picture aspect ratio defines the shape of displayed image. Picture aspect ratio is notated X:Y, where X:Y is the ration of picture width to height. If NTrue [1064] value is used it means that NImage [1232] shape will be defined using width and height ratio (aspect ratio).

Module

NImage Unit [1232]

7.1.5.7.1.54 NImageGetInfo Function

Retrieves a NImageInfo object for the specified image.

C++

```
NResult N_API NImageGetInfo(HNImage hImage, HNImageInfo * phValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image. `phValue : [out] Pointer to a handle to the NImageInfo.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.55 NImageSetVertResolution Function

Sets vertical resolution for specified NImage [1232].

C++

```
NResult N_API NImageSetVertResolution(HNImage hImage, NFloat value);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to NImage [1232].
NFloat value	[in] Vertical resolution (pixels per inch).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.56 NImageGetMaxValue Function

Retrieves the maximum pixel value for the specified image.

C++

```
NResult N_API NImageGetMaxValue(HNImage hImage, HNArray * phValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
HNArray * phValue	[out] Pointer to a handle to the maximum pixel value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.57 NImageToGrayscale Function

Converts NImage [1232] to NGrayscaleImage [1228].

C++

```
NResult N_API NImageToGrayscale(HNImage hImage, HNGrayscaleImage * phResultImage);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to NImage [1232].
HNGrayscaleImage * phResultImage	[out] Pointer to NGrayscaleImage [1228].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.58 NImageGetMinValue Function

Retrieves the minimum pixel value for the specified image.

C++

```
NResult N_API NImageGetMinValue(HNImage hImage, HNArray * phValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
HNArray * phValue	[out] Pointer to a handle to the minimum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.59 NImageToMonochrome Function

Converts NImage [1232] to monochrome image.

C++

```
NResult N_API NImageToMonochrome(HNImage hImage, HNMonochromeImage * phResultImage);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to NImage [1232].
HNMonochromeImage * phResultImage	[out] Pointer to NMonochromeImage [1355].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.60 NImageGetPalette Function

Retrieves the palette of the specified image.

C++

```
NResult N_API NImageGetPalette(HNImage hImage, HNBuffer * phValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
HNBuffer * phValue	[out] Pointer to a handle to NBuffer [313] object containing the palette.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.61 NImageToRgb Function

Converts NImage [1232] to Rgb image.

C++

```
NResult N_API NImageToRgb(HNImage hImage, HNRgbImage * phResultImage);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to NImage [1232].
HNRgbImage * phResultImage	[out] Pointer to Rgb image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.62 NImageGetPaletteEntry Function

Retrieves the value of the specified palette entry.

C++

```
NResult N_API NImageGetPaletteEntry(HNImage hImage, NInt index, HNArray * phValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NInt index	[in] Index of the palette entry.
HNArray * phValue	[out] Pointer to a handle to the palette entry value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.63 HNImage

Handle to image.

Module

NImage Unit [1232]

7.1.5.7.1.64 NImageGetPaletteLength Function

Retrieves the length of the specified image's palette.

C++

```
NResult N_API NImageGetPaletteLength(HNImage hImage, NInt * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NInt * pValue	[out] Pointer to the palette length value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.65 NImageGetPalettePtr Function

Retrieves a pointer to the specified image's palette.

C++

```
NResult N_API NImageGetPalettePtr(HNImage hImage, void * * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
void * * pValue	[out] Pointer to the palette.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.66 NImageGetPaletteSize Function

Retrieves the size of the specified image's palette.

C++

```
NResult N_API NImageGetPaletteSize(HNImage hImage, NSizeType * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NSizeType * pValue	[out] Pointer to the palette size value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.67 NImageGetPixel Function

Retrieves the specified pixel of the specified image.

C++

```
NResult N_API NImageGetPixel(HNImage hImage, NUInt x, NUInt y, HNArray * phValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NUInt x	[in] X-coordinate of the pixel.
NUInt y	[in] Y-coordinate of the pixel.
HNArray * phValue	[out] Pointer to a handle to an array representing the pixel.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.68 NImageGetPixelFormat Function

Retrieves pixel format of the image.

C++

```
NResult N_API NImageGetPixelFormat(HNImage hImage, NPixelFormat_ * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NPixelFormat_ * pValue	[out] Pointer to NPixelFormat [1369] that receives pixel format of the image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error code is returned when *hImage* or *pValue [870]* is NULL [1075].

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.1.69 NImageGetPixelsN Function

Retrieves pointer to memory block containing pixels of the image.

C++

```
NResult N_API NImageGetPixelsN(HNImage hImage, HNBuffer * phValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
HNBuffer * phValue	[out] Pointer toNBuffer that receives pointer to memory block containing pixels of the image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *ppValue* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *ppValue* is NULL [1075].

Remarks

Memory block containing image pixels is organized as image height rows following each other in top-to-bottom order. Each row occupies image stride bytes and is organized as image width pixels following each other in right-to-left order. Each pixel is described by image pixel format.

For more information see NImageGetPixelFormat [1277], NImageGetWidth [1267], NImageGetHeight [1257], NImageGetStride [1263], and NImageGetSize functions.

Module

NImage Unit [1232]

7.1.5.7.1.70 NImageGetPixelsPtr Function

Retrieves a pointer to the pixels of the specified image.

C++

```
NResult N_API NImageGetPixelsPtr(HNImage hImage, void * * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the NImage [1232].
void * * pValue	[out] A pointer to the pointer to the image pixels.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.71 NImageGetPlane Function

Retrieves a pointer to the specified image data plane.

C++

```
NResult N_API NImageGetPlane(HNImage hImage, NInt index, HNBuffer * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the NImage [1232].
NInt index	[in] Plane index.
HNBuffer * phValue	[out] A pointer to the buffer containing the specified plane.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.72 NImageGetPlaneCount Function

Retrieves a pointer to the plane count of the specified image.

C++

```
NResult N_API NImageGetPlaneCount(HNImage hImage, NInt * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the NImage [1232].
NInt * pValue	[out] A pointer to the number of the planes.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.73 NImageGetPlanePtr Function

Gets a pointer to the specified image data plane.

C++

```
NResult N_API NImageGetPlanePtr(HNImage hImage, NInt index, void * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the NImage [1232].
NInt index	[in] Plane index.
void * pValue	[out] A pointer to the plane.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.74 NImageGetPlanePtrs Function

Retrieves an array of pointer to the planes of the specified image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NImageGetPlanePtrs(HNImage hImage, void * * arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the NImage [1232].
void * * arValues	[out] An array of pointers to pointers to the planes.
NInt valuesLength	[in] Number of pointers in the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.75 NImageGetPlanes Function

Retrieves an array of pointer to the buffers containing planes of the specified image.

C++

```
NResult N_API NImageGetPlanes(HNImage hImage, HNBuffer * arhValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the NImage [1232].
HNBuffer * arhValues	[out] An array of pointers to handles to the buffers containing the planes.
NInt valuesLength	[in] Number of pointers in the array.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.76 NImageGetPlaneSize Function

Retrieves a pointer to the plane size of the specified image.

C++

```
NResult N_API NImageGetPlaneSize(HNImage hImage, NSizeType * pValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the NImage [1232].
NSizeType * pValue	[out] A pointer to the plane size value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.1.77 NImageSetIndex Function

Sets the palette entry index for the specified pixel.

C++

```
NResult N_API NImageSetIndex(HNImage hImage, NUInt x, NUInt y, NInt value);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NUInt x	[in] X-coordinate of the pixel.
NUInt y	[in] Y-coordinate of the pixel.
NInt value	[in] Index of the palette entry.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.78 NImageSetMaxValue Function

Sets the maximum pixel value for the specified image.

C++

```
NResult N_API NImageSetMaxValue(HNImage hImage, HNArray hValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
HNArray hValue	[in] Handle to the maximum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.79 NImageSetMinValue Function

Sets the minimum pixel value for the specified image.

C++

```
NResult N_API NImageSetMinValue(HNImage hImage, HNArray hValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
HNArray hValue	[in] Handle to the maximum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.1.80 NImageSetPalette Function

Sets the specified image's palette to the specified value.

C++

```
NResult N_API NImageSetPalette(HNImage hImage, const void * pPalette, NSizeType
paletteSize, NInt paletteLength);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
const void * pPalette	[in] Pointer to a memory buffer containing new palette value.
NSizeType paletteSize	[in] Size of the new palette.
NInt paletteLength	[in] Length of the new palette.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.81 NImageSetPaletteEntry Function

Sets the value of the specified palette entry to the specified value.

C++

```
NResult N_API NImageSetPaletteEntry(HNImage hImage, NInt index, HNArray hValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NInt index	[in] Palette entry index.
HNArray hValue	[in] Handle to the new palette entry value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.82 NImageSetPaletteN Function

Sets the specified image's palette to the specified value.

C++

```
NResult N_API NImageSetPaletteN(HNImage hImage, HNBuffer hPalette, NInt paletteLength);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
HNBuffer hPalette	[in] Handle to NBuffer [313] object containing new palette value.
NInt paletteLength	[in] Length of the new palette.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.1.83 NImageSetPixel Function

Sets specified pixel's value to the specified value.

C++

```
NResult N_API NImageSetPixel(HNImage hImage, NUInt x, NUInt y, HNArray hValue);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NUInt x	[in] X-coordinate of the pixel.
NUInt y	[in] Y-coordinate of the pixel.
HNArray hValue	[in] Handle to the new pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.2 Structs, Records, Enums

7.1.5.7.2.1 NImageRotateFlipType Enumeration

Enumerates different image rotation types.

C++

```
typedef enum NImageRotateFlipType_ {
  nirftRotateNone = 0,
  nirftRotate90 = 1,
  nirftRotate180 = 2,
  nirftRotate270 = 3,
  nirftFlipNone = 0,
  nirftFlipX = 4,
  nirftFlipY = 8,
  nirftFlipXY = nirftFlipX|nirftFlipY,
  nirftNone = nirftRotateNone|nirftFlipNone
} NImageRotateFlipType;
```

Members

Members	Description
nirftRotateNone = 0	Indicates that image should not be rotated.
nirftRotate90 = 1	Rotates image by 90 degrees.
nirftRotate180 = 2	Rotates image by 180 degrees.
nirftRotate270 = 3	Rotates image by 270 degrees.
nirftFlipNone = 0	Indicates that image should not be flipped.
nirftFlipX = 4	Flips the image's orientation of the X coordinates.
nirftFlipY = 8	Flips the image's orientation of the Y coordinates.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

nirftFlipXY = nirftFlipX|nirftFlipY	Flips the image's orientation of the X or Y coordinates.
nirftNone = nirftRotateNone|nirftFlipNone	Indicates that image should not be rotated or flipped.

Module

NImage Unit [1232]

7.1.5.7.3 Macros

7.1.5.7.3.1 NI_ALL_DST Macro

For internal use.

C++

```
#define NI_ALL_DST 0x00000F00
```

Module

NImage Unit [1232]

7.1.5.7.3.2 NI_ALL_DST_AND_SRC Macro

For internal use.

C++

```
#define NI_ALL_DST_AND_SRC (NI_ALL_DST | NI_ALL_SRC)
```

Module

NImage Unit [1232]

7.1.5.7.3.3 NI_ALL_SRC Macro

For internal use.

C++

```
#define NI_ALL_SRC 0x0000F000
```

Module

NImage Unit [1232]

7.1.5.7.3.4 NI_DST_ALPHA_CHANNEL_FIRST Macro

Copies Nimage using Rgb Alpha channel first method (ARGB color space).

C++

```
#define NI_DST_ALPHA_CHANNEL_FIRST 0x00000200
```

Module

NImage Unit [1232]

7.1.5.7.3.5 NI_DST_BOTTOM_TO_TOP Macro

Copies NImage [1232] changing pixels order bottom to top (bottom lines become top and vice versa).

C++

```
#define NI_DST_BOTTOM_TO_TOP 0x00000400
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NImage Unit [1232]

7.1.5.7.3.6 NI_DST_SWAP_CHANNELS Macro

Copies NImage [1232] swapping RGB channels (e.g. RGB to BRG).

C++

```
#define NI_DST_SWAP_CHANNELS 0x00000100
```

Module

NImage Unit [1232]

7.1.5.7.3.7 NI_READ_UNKNOWN_IMAGE Macro

Flag specifying whether unknown images should be read.

C++

```
#define NI_READ_UNKNOWN_IMAGE 0x00000001
```

Module

NImage Unit [1232]

7.1.5.7.3.8 NI_SRC_ALPHA_CHANNEL_FIRST Macro

Creates Nimage using Rgb Alpha channel first method (ARGB color space).

C++

```
#define NI_SRC_ALPHA_CHANNEL_FIRST 0x00002000
```

Module

NImage Unit [1232]

7.1.5.7.3.9 NI_SRC_BOTTOM_TO_TOP Macro

Creates NImage [1232] changing pixels order bottom to top (bottom lines become top and vice versa).

C++

```
#define NI_SRC_BOTTOM_TO_TOP 0x00004000
```

Module

NImage Unit [1232]

7.1.5.7.3.10 NI_SRC_SWAP_CHANNELS Macro

Creates NImage [1232] swapping RGB channels (e.g. RGB to BRG).

C++

```
#define NI_SRC_SWAP_CHANNELS 0x00001000
```

Module

NImage Unit [1232]

7.1.5.7.3.11 NI_WRITE_UNKNOWN_IMAGE Macro

Flag specifying whether unknown images should be written.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NI_WRITE_UNKNOWN_IMAGE 0x00000001
```

Module

NImage Unit [1232]

7.1.5.7.3.12 NImageClone Macro

Creates a copy of the specified image.

C++

```
#define NImageClone(hImage, phClonedImage) NObjectClone(hImage, (HNObject *)phClonedImage)
```

Parameters

Parameters	Description
hImage	[in] Handle to the original image.
phClonedImage	[out] Pointer to a handle to the copied image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.3.13 NImageCreateFromFileA Macro

Creates (loads) an image from file with specified format.

C++

```
#define NImageCreateFromFileA(szFileName, hImageFormat, phImage)
NImageCreateFromFileExA(szFileName, hImageFormat, 0, NULL, phImage)
```

Parameters

Parameters	Description
szFileName	[in] Pointer to a string that specifies file name.
hImageFormat	[in] Handle to the image format of the file. Can be NULL [1075].
phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.3.14 NImageCreateFromFileEx Macro

Picks a Unicode (W) or ANSCII (A) version of the NImageCreateFromFileEx function.

C++

```
#define NImageCreateFromFileEx N_FUNC_AW(NImageCreateFromFileEx)
```

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.3.15 NImageCreateFromFileW Macro

Creates (loads) an image from file with specified format.

C++

```
#define NImageCreateFromFileW(szFileName, hImageFormat, phImage)
NImageCreateFromFileExW(szFileName, hImageFormat, 0, NULL, phImage)
```

Parameters

Parameters	Description
szFileName	[in] Pointer to a string that specifies file name.
hImageFormat	[in] Handle to the image format of the file. Can be NULL [1075].
phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.3.16 NImageCreateFromImage Macro

Creates an image from specified image with specified pixel format and stride.

C++

```
#define NImageCreateFromImage(pixelFormat, stride, hSrcImage, phImage)
NImageCreateFromImageEx2(pixelFormat, stride, hSrcImage, 0, phImage)
```

Parameters

Parameters	Description
pixelFormat	[in] Specifies pixel format of the image.
stride	[in] Specifies stride of the image. Can be zero.
hSrcImage	[in] Handle to image used as source for the image.
phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.3.17 NImageCreateFromImagePart Macro

Creates an image from other image with specified pixel format, size and stride.

C++

```
#define NImageCreateFromImagePart(pixelFormat, stride, hSrcImage, left, top, width, height,
phImage) NImageCreateFromImagePartEx2(pixelFormat, stride, hSrcImage, left, top, width,
height, 0, phImage)
```

Parameters

Parameters	Description
pixelFormat	[in] Specifies pixel format of the image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

stride	[in] Specifies stride of the image. Can be zero.
hSrcImage	[in] Handle to source NImage [1232] object.
left	[in] X coordinate of the source image.
top	[in] Y coordinate of the source image.
width	[in] Specifies width of the image.
height	[in] Specifies height of the image.
phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.3.18 NImageCreateWrapperForImagePart Macro

Creates an image wrapper for a part of specified NImage [1232] with specified size.

C++

```
#define NImageCreateWrapperForImagePart(hSrcImage, left, top, width, height, phImage)
NImageCreateWrapperForImagePartEx2(hSrcImage, left, top, width, height, 0, phImage)
```

Parameters

Parameters	Description
hSrcImage	[in] A handle to the source image.
left	[in] X coordinate of the image.
top	[in] Y coordinate of the image.
width	[in] Specifies width of the image.
height	[in] Specifies height of the image.
phImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds and *phImage* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *pixelFormat* has invalid value. - or - *stride* is less than minimal value for specified pixel format and width.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pixels* or *phImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *width* or *height* is zero. - or - *horzResolution* or *vertResolution* is less than zero.

Module

NImage Unit [1232]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.7.3.19 NImageFree Macro

Frees the specified image.

C++

```
#define NImageFree(hImage) NObjectFree(hImage)
```

Parameters

Parameters	Description
hImage	[in] Handle to the image to be freed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.3.20 NImageGetPixels Macro

Retrieves a pointer to the pixels of the specified image.

C++

```
#define NImageGetPixels(hImage, pValue) NImageGetPixelsPtr(hImage, pValue)
```

Parameters

Parameters	Description
hImage	[in] Handle to the NImage [1232].
pValue	[out] A pointer to the image pixels.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImage Unit [1232]

7.1.5.7.3.21 NImageSaveToFileA Macro

Saves the image to the file of specified format.

C++

```
#define NImageSaveToFileA(hImage, szFileName, hImageFormat) NImageSaveToFileExA(hImage,
szFileName, hImageFormat, NULL, 0)
```

Parameters

Parameters	Description
hImage	[in] Handle to NImage [1232] object.
szFileName	[in] Points to string that specifies file name.
hImageFormat	[in] Handle to the image format of the file. Can be NULL [1075].

Returns

Return values	Description
N_OK [433]	If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *szFileName* is NULL [1075].
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* is NULL [1075] and none of supported image formats is registered with file extension of *szFileName*. - or - *hImageFormat* is NULL [1075] and image format registered with file extension of *szFileName* does not support writing. - or - Image format specified by *hImageFormat* does not support writing.

Remarks

If *hImageFormat* is NULL [1075] image format is selected by file extension of *szFileName*.

Module

NImage Unit [1232]

7.1.5.7.3.22 NImageSaveToFileEx Macro

Picks a Unicode (W) or ANSCII (A) version of the NImageSaveToFileEx function.

C++

```
#define NImageSaveToFileEx N_FUNC_AW(NImageSaveToFileEx)
```

Module

NImage Unit [1232]

7.1.5.7.3.23 NImageSaveToFileW Macro

Saves the image to the file of specified format.

C++

```
#define NImageSaveToFileW(hImage, szFileName, hImageFormat) NImageSaveToFileExW(hImage,
szFileName, hImageFormat, NULL, 0)
```

Parameters

Parameters	Description
hImage	[in] Handle to NImage [1232] object.
szFileName	[in] Points to string that specifies file name.
hImageFormat	[in] Handle to the image format of the file. Can be NULL [1075].

Returns

Return values	Description
N_OK [433]	If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *szFileName* is NULL [1075].

7

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* is NULL [1075] and none of supported image formats is registered with file extension of *szFileName*. - or - *hImageFormat* is NULL [1075] and image format registered with file extension of *szFileName* does not support writing. - or - Image format specified by *hImageFormat* does not support writing.

Remarks

If *hImageFormat* is NULL [1075] image format is selected by file extension of *szFileName*.

Module

NImage Unit [1232]

7.1.5.7.4 Files

7.1.5.7.4.1 NImage.h

Header file for NImage [1232] module. Provides functionality for managing images.

Enumerations

	Name	Description
	NImageRotateFlipType_ [1283]	Enumerates different image rotation types.
	NImageRotateFlipType [1283]	Enumerates different image rotation types.

Functions

	Name	Description
	NImageCopy [1235]	Copies specified rectangular area of the NImage [1232] to other NImage [1232] object.
	NImageCopyData [1236]	Copies pixels from one memory buffer to another.
	NImageCopyDataPlanes [1237]	Copies NImage [1232] planes from one memory buffer to another.
	NImageCopyDataPlanesEx [1238]	Copies NImage [1232] planes from one memory buffer to another.
	NImageCopyFromData [1239]	Copies pixels to specified NImage [1232] from memory buffer.
	NImageCopyFromDataPart [1240]	Copies part of pixels to specified NImage [1232] from memory buffer.
	NImageCopyFromDataPlanesPart [1241]	Copies part of planes of specified NImage [1232] from memory buffer.
	NImageCopyFromDataPlanesPartEx [1242]	Copies part of planes of specified NImage [1232] from memory buffer.
	NImageCopyTo [1236]	Copies specified rectangular area of the NImage [1232] to other NImage [1232] object.
	NImageCopyToData [1243]	Copies pixels of specified NImage [1232] to memory buffer.
	NImageCopyToDataPart [1244]	Copies part of pixels of specified NImage [1232] to memory buffer.
	NImageCopyToDataPlanesPart [1245]	Copies part of planes of specified NImage [1232] to memory buffer.
	NImageCopyToDataPlanesPartEx [1246]	Copies part of planes of specified NImage [1232] to memory buffer.
	NImageCreate [1247]	Creates an image with specified pixel format, size, stride and resolution.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NImageCreateEx [1248]	Creates an image with specified pixel format, size, stride and resolution.
	NImageCreateFromData [1249]	Creates an image with specified pixel format, size, stride and resolution and copies specified pixels to it.
	NImageCreateFromDataEx [1251]	Creates an image with specified pixel format, size and stride from specified memory block.
	NImageCreateFromDataPart [1251]	Creates an image with specified pixel format, size, stride and resolution to specified source rectangular area.
	NImageCreateFromDataPartEx [1252]	Creates an image with specified pixel format, size and stride source rectangular area.
	NImageCreateFromDataPlanesPart [1254]	Creates an image with specified plane format, size, stride and resolution to specified source rectangular area.
	NImageCreateFromFileEx [1242]	Creates (loads) an image from file with specified format.
	NImageCreateFromFileExN [1256]	Creates (loads) an image from file with specified format.
	NImageCreateFromImageEx [1257]	Creates an image from specified image with specified pixel format, stride and resolution.
	NImageCreateFromImageEx2 [1258]	Creates an image from specified image with specified pixel format and stride.
	NImageCreateFromImagePartEx [1258]	Creates an image from other image with specified pixel format, size and stride.
	NImageCreateFromImagePartEx2 [1260]	Creates an image from other image with specified pixel format, size and stride.
	NImageCreateFromMemory [1253]	Creates NImage [1232] from memory buffer.
	NImageCreateFromStream [1255]	Creates (loads) an image from stream with specified format.
	NImageCreateWrapper [1261]	Creates an image wrapper for specified image pixels with specified pixel format, size, stride and resolution.
	NImageCreateWrapperEx [1262]	Creates an image wrapper for specified image pixels with specified pixel format, size, stride and resolution.
	NImageCreateWrapperForImagePartEx2 [1260]	Creates an image wrapper for a part of specified NImage [1232] with specified size.
	NImageCreateWrapperForPart [1264]	Creates an image wrapper for specified image pixels with specified pixel format, size, stride and resolution.
	NImageCreateWrapperForPartEx [1266]	Creates an image wrapper for specified image pixels with specified pixel format, size, stride and resolution.
	NImageCreateWrapperForPlanesPart [1267]	Creates an image wrapper for specified image planes with specified pixel format, size, stride and resolution.
	NImageCrop [1248]	Crops an image.
	NImageExportPalette [1268]	Exports the palette from the specified image.
	NImageFlipDiagonally [1250]	Flips image diagonally.
	NImageFlipHorizontally [1254]	Flips image horizontally.
	NImageFlipVertically [1256]	Flips image vertically.
	NImageGetHeight [1257]	Retrieves height of the image.
	NImageGetHorzResolution [1259]	Retrieves horizontal resolution of the image.
	NImageGetImageSize [1271]	Retrieves size of memory block containing pixels of the image.
	NImageGetIndex [1272]	Retrieves the palette entry index of the specified pixel.
	NImageGetInfo [1273]	Retrieves a NImageInfo object for the specified image.
	NImageGetMaxValue [1273]	Retrieves the maximum pixel value for the specified image.
	NImageGetMinValue [1274]	Retrieves the minimum pixel value for the specified image.
	NImageGetPalette [1275]	Retrieves the palette of the specified image.
	NImageGetPaletteEntry [1275]	Retrieves the value of the specified palette entry.
	NImageGetPaletteLength [1276]	Retrieves the length of the specified image's palette.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NImageGetPalettePtr [1276]	Retrieves a pointer to the specified image's palette.
	NImageGetPaletteSize [1276]	Retrieves the size of the specified image's palette.
	NImageGetPixel [1277]	Retrieves the specified pixel of the specified image.
	NImageGetPixelFormat [1277]	Retrieves pixel format of the image.
	NImageGetPixelsN [1278]	Retrieves pointer to memory block containing pixels of the image.
	NImageGetPixelsPtr [1278]	Retrieves a pointer to the pixels of the specified image.
	NImageGetPlane [1278]	Retrieves a pointer to the specified image data plane.
	NImageGetPlaneCount [1279]	Retrieves a pointer to the plane count of the specified image.
	NImageGetPlanePtr [1279]	Gets a pointer to the specified image data plane.
	NImageGetPlanePtrs [1279]	Retrieves an array of pointer to the planes of the specified image.
	NImageGetPlanes [1280]	Retrieves an array of pointer to the buffers containing planes of the specified image.
	NImageGetPlaneSize [1280]	Retrieves a pointer to the plane size of the specified image.
	NImageGetResolutionIsAspectRatio [1269]	Gets NImage [1232] resolution aspect ratio.
	NImageGetStride [1263]	Retrieves stride (size of one row) of the image.
	NImageGetVertResolution [1265]	Retrieves vertical resolution of the image.
	NImageGetWidth [1267]	Retrieves width of the image.
	NImageRotateFlip [1268]	Rotates image using rotation type.
	NImageSaveToFileEx [1269]	Saves the image to the file of specified format.
	NImageSaveToMemory [1270]	Saves the image to memory of specified format.
	NImageSaveToStream [1271]	Saves the image of specified format to stream
	NImageSetHorzResolution [1272]	Sets horizontal resolution for specified NImage [1232].
	NImageSetIndex [1281]	Sets the palette entry index for the specified pixel.
	NImageSetMaxValue [1281]	Sets the maximum pixel value for the specified image.
	NImageSetMinValue [1281]	Sets the minimum pixel value for the specified image.
	NImageSetPalette [1282]	Sets the specified image's palette to the specified value.
	NImageSetPaletteEntry [1282]	Sets the value of the specified palette entry to the specified value.
	NImageSetPaletteN [1282]	Sets the specified image's palette to the specified value.
	NImageSetPixel [1283]	Sets specified pixel's value to the specified value.
	NImageSetResolutionIsAspectRatio [1272]	Sets NImage [1232] resolution aspect ratio.
	NImageSetVertResolution [1273]	Sets vertical resolution for specified NImage [1232].
	NImageToGrayscale [1274]	Converts NImage [1232] to NGrayscaleImage [1228].
	NImageToMonochrome [1274]	Converts NImage [1232] to monochrome image.
	NImageToRgb [1275]	Converts NImage [1232] to Rgb image.

Macros

Name	Description
NI_ALL_DST [1284]	For internal use.
NI_ALL_DST_AND_SRC [1284]	For internal use.
NI_ALL_SRC [1284]	For internal use.
NI_DST_ALPHA_CHANNEL_FIRST [1284]	Copies Nimage using Rgb Alpha channel first method (ARGB color space).
NI_DST_BOTTOM_TO_TOP [1284]	Copies NImage [1232] changing pixels order bottom to top (bottom lines become top and vice versa).
NI_DST_SWAP_CHANNELS [1285]	Copies NImage [1232] swapping RGB channels (e.g. RGB to BRG).
NI_READ_UNKNOWN_IMAGE [1285]	Flag specifying whether unknown images should be read.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NI_SRC_ALPHA_CHANNEL_FIRST [1285]	Creates Nimage using Rgb Alpha channel first method (ARGB color space).
NI_SRC_BOTTOM_TO_TOP [1285]	Creates NImage [1232] changing pixels order bottom to top (bottom lines become top and vice versa).
NI_SRC_SWAP_CHANNELS [1285]	Creates NImage [1232] swapping RGB channels (e.g. RGB to BRG).
NI_WRITE_UNKNOWN_IMAGE [1285]	Flag specifying whether unknown images should be written.
NImageClone [1286]	Creates a copy of the specified image.
NImageCreateFromFileA [1286]	Creates (loads) an image from file with specified format.
NImageCreateFromFileEx [1286]	Picks a Unicode (W) or ANSCII (A) version of the NImageCreateFromFileEx function.
NImageCreateFromFileW [1287]	Creates (loads) an image from file with specified format.
NImageCreateFromImage [1287]	Creates an image from specified image with specified pixel format and stride.
NImageCreateFromImagePart [1287]	Creates an image from other image with specified pixel format, size and stride.
NImageCreateWrapperForImagePart [1288]	Creates an image wrapper for a part of specified NImage [1232] with specified size.
NImageFree [1289]	Frees the specified image.
NImageGetPixels [1289]	Retrieves a pointer to the pixels of the specified image.
NImageSaveToFileA [1289]	Saves the image to the file of specified format.
NImageSaveToFileEx [1290]	Picks a Unicode (W) or ANSCII (A) version of the NImageSaveToFileEx function.
NImageSaveToFileW [1290]	Saves the image to the file of specified format.

Module

NImage Unit [1232]

7.1.5.8 NImageFile Unit

Provides functionality for reading image files in format-neutral way.

Files

Name	Description
NImageFile.h [1298]	Header file for NImageFile module. Provides functionality for reading image files in format-neutral way.

Functions

	Name	Description
	NImageFileClose [1295]	Closes the file associated with the image file.
	NImageFileCreate [1295]	Opens image file of specified format.
	NImageFileGetFormat [1296]	Retrieves image format of the image file.
	NImageFileGetFormatEx [1296]	Retrieves image format of the image file.
	NImageFileIsOpened [1297]	Retrieves a value indicating whether the file associated with the image file is opened.
	NImageFileReadImage [1297]	Reads image from the image file.
	HNImageFile [1298]	Handle to opened read-only image file.

Macros

Name	Description
NImageFileFree [1298]	Frees the specified image file.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.8.1 Functions

7.1.5.8.1.1 NImageFileClose Function

Closes the file associated with the image file.

C++

```
NResult N_API NImageFileClose(HNImageFile hImageFile);
```

Parameters

Parameters	Description
HNImageFile hImageFile	[in] Handle to image file.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFile* is NULL [1075].

Remarks

If the file associated with the image file is already closed the function does nothing.

Module

NImageFile Unit [1294]

7.1.5.8.1.2 NImageFileCreate Function

Opens image file of specified format.

C++

```
NResult N_API NImageFileCreate(const NChar * szFileName, HNImageFormat hImageFormat,
HNImageFile * phImageFile);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Points to string that specifies file name.
HNImageFormat hImageFormat	[in] Handle to the image format of the file. Can be NULL [1075].
HNImageFile * phImageFile	[out] Pointer to HNImageFile [1298] that receives handle to opened image file.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *szFileName* or *phImageFile* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file specified by szFileName is invalid for specified image format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* is NULL [1075] and none of supported image formats is registered with file extension of *szFileName*. - or - *hImageFormat* is NULL [1075] and image format registered with file extension of *szFileName* does not support reading. - or - Image format specified by *hImageFormat* does not support reading.

Remarks

If *hImageFormat* is NULL [1075] image format is selected by file extension of *szFileName*.

Opened image file must be closed using NObjectFree [542] function.

This function does not check format of the file for all image formats. However format of the file is always checked in NImageFileReadImage [1297] function.

Module

NImageFile Unit [1294]

7.1.5.8.1.3 NImageFileGetFormat Function

Retrieves image format of the image file.

C++

```
NResult N_API NImageFileGetFormat(HNImageFile hImageFile, HNImageFormat * phValue);
```

Parameters

Parameters	Description
HNImageFile hImageFile	[in] Handle to image file.
HNImageFormat * phValue	[out] Pointer to NImageFormat [1299] that receives image format of the image file.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFile* or *phValue* is NULL [1075].

Remarks

This function is deprecated. Use NImageGetFormatEx instead.

Module

NImageFile Unit [1294]

7.1.5.8.1.4 NImageFileGetFormatEx Function

Retrieves image format of the image file.

C++

```
NResult N_API NImageFileGetFormatEx(HNImageFile hImageFile, HNImageFormat * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImageFile hImageFile	[in] Handle to image file.
HNImageFormat * phValue	[out] Pointer to NImageFormat [1299] that receives image format of the image file.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFile* or *phValue* is NULL [1075].

Module

NImageFile Unit [1294]

7.1.5.8.1.5 NImageFileIsOpened Function

Retrieves a value indicating whether the file associated with the image file is opened.

C++

```
NResult N_API NImageFileIsOpened(HNImageFile hImageFile, NBool * pValue);
```

Parameters

Parameters	Description
HNImageFile hImageFile	[in] Handle to image file.
NBool * pValue	[out] Pointer to NBool [1004] that receives value indicating whether the file associated with the image file is opened.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFile* or *pValue [870]* is NULL [1075].

Module

NImageFile Unit [1294]

7.1.5.8.1.6 NImageFileReadImage Function

Reads image from the image file.

C++

```
NResult N_API NImageFileReadImage(HNImageFile hImageFile, HNImage * phImage);
```

Parameters

Parameters	Description
HNImageFile hImageFile	[in] Handle to image file.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to image read from the image file.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFile* or *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file associated with the image file is invalid for the image format of the image file.
N_E_INVALID_OPERATION [430]	If the function fails, this error can be returned. This error code is returned when file associated with the image file is closed.

Remarks

If all images are already read from the image file then handle to image returned via *phImage* is NULL [1075].

Module

NImageFile Unit [1294]

7.1.5.8.1.7 HNImageFile

Handle to opened read-only image file.

Module

NImageFile Unit [1294]

7.1.5.8.2 Macros

7.1.5.8.2.1 NImageFileFree Macro

Frees the specified image file.

C++

```
#define NImageFileFree(hImageFile) NObjectFree(hImageFile)
```

Parameters

Parameters	Description
hImageFile	[in] Handle to the image file to be freed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFile Unit [1294]

7.1.5.8.3 Files

7.1.5.8.3.1 NImageFile.h

Header file for NImageFile [1294] module. Provides functionality for reading image files in format-neutral way.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NImageFileClose [1295]	Closes the file associated with the image file.
	NImageFileCreate [1295]	Opens image file of specified format.
	NImageFileGetFormat [1296]	Retrieves image format of the image file.
	NImageFileGetFormatEx [1296]	Retrieves image format of the image file.
	NImageFileIsOpened [1297]	Retrieves a value indicating whether the file associated with the image file is opened.
	NImageFileReadImage [1297]	Reads image from the image file.

Macros

Name	Description
NImageFileFree [1298]	Frees the specified image file.

Module

NImageFile Unit [1294]

7.1.5.9 NImageFormat Unit

Provides functionality for loading and saving images in format-neutral way.

Files

Name	Description
NImageFormat.h [1329]	Header file for NImageFormat module. Provides functionality for loading and saving images in format-neutral way.

Functions

	Name	Description
	NImageFormatCanRead [1301]	Retrieves a value indicating whether the image format supports reading.
	NImageFormatCanWrite [1302]	Retrieves a value indicating whether the image format supports writing.
	NImageFormatCanWriteMultiple [1302]	Retrieves a value indicating whether the image format supports writing of multiple images.
	NImageFormatCreateInfo [1303]	Creates NImageInfo for the specified image with a specified format.
	NImageFormatGetBmp [1303]	Retrieves BMP image format.
	NImageFormatGetDefaultFileExtensionEx [1303]	Retrieves default extension of the image format.
	NImageFormatGetBmpEx [1304]	Retrieves BMP image format.
	NImageFormatGetFormat [1304]	Retrieves supported image format with the specified index.
	NImageFormatGetDefaultInternetMediaType [1305]	Gets the default internet media type of the specified image format.
	NImageFormatGetFileFilterEx [1305]	Retrieves file filter of the image format.
	NImageFormatGetFormatCount [1306]	Retrieves number of supported image formats.
	NImageFormatGetFormatEx [1306]	Retrieves supported image format with the specified index.
	NImageFormatGetIHead [1306]	Retrieves NIST IHead [1201] image format.
	NImageFormatGetJpeg [1307]	Retrieves JPEG image format.
	NImageFormatGetIHeadEx [1307]	Retrieves NIST IHead [1201] image format.
	NImageFormatGetJpeg2K [1308]	Retrieves JPEG2K image format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NImageFormatGetInternetMediaType [1308]	Retrieves Internet media type.
	NImageFormatGetNameEx [1309]	Retrieves name of the image format.
	NImageFormatGetPng [1309]	Retrieves PNG image format.
	NImageFormatGetJpeg2KEx [1310]	Retrieves JPEG2K image format.
	NImageFormatOpenFile [1310]	Opens image file with specified file name and image format.
	NImageFormatGetJpegEx [1311]	Retrieves JPEG image format.
	NImageFormatGetTiff [1311]	Retrieves TIFF image format.
	NImageFormatGetWsq [1311]	Retrieves WSQ image format.
	NImageFormatGetPngEx [1312]	Retrieves PNG image format.
	NImageFormatLoadImageFromFile [1312]	Loads image from file of specified image format.
	NImageFormatSaveImagesToFile [1313]	Saves specified number of images to the file in the specified format.
	NImageFormatGetTiffEx [1313]	Retrieves TIFF image format.
	NImageFormatSaveImagesToMemory [1314]	Saves specified number of images to a memory buffer in the specified format.
	NImageFormatLoadImageFromMemory [1314]	Loads image from the memory buffer containing file of the specified format.
	NImageFormatGetWsqEx [1315]	Retrieves WSQ image format.
	NImageFormatLoadImageFromStream [1315]	Loads image from the memory stream containing file of the specified format.
	NImageFormatSaveImagesToStream [1316]	Saves specified number of images to a stream in the specified format.
	NImageFormatOpenFileFromMemory [1317]	Opens image file from the memory buffer containing file of specified format.
	NImageFormatOpenFileFromStream [1317]	Opens image file from the memory stream containing file of specified format.
	NImageFormatOpenReaderFromFile [1318]	Opens NImageReader from a specified file.
	NImageFormatOpenReaderFromStream [1318]	Opens NImageReader from a specified stream.
	NImageFormatSaveImageToFile [1319]	Saves image to the file in the specified format.
	NImageFormatOpenWriterToStream [1319]	Opens NImageWriter to a specified stream.
	NImageFormatOpenReaderFromMemory [1319]	Opens NImageReader from a specified memory buffer.
	NImageFormatSaveImageToMemory [1320]	Saves image to the memory buffer in the specified format.
	NImageFormatSaveImageToStream [1320]	Saves image to the memory stream in the specified format.
	NImageFormatOpenWriterToFile [1321]	Opens NImageWriter to a specified file.
	NImageFormatSelect [1321]	Retrieves supported image format registered with file extension of specified file name and supporting reading/writing as specified.
	NImageFormatSelectByInternetMediaType [1322]	Retrieves supported image format registered with Internet media type of specified file name and supporting reading/writing as specified.
	NImageFormatSelectEx [1322]	Retrieves supported image format registered with file extension of specified file name and supporting reading/writing as specified.
	NImageFormatOpenWriterToMemory [1323]	Opens NImageWriter to a memory buffer.
	NImageFormatSelectReaderFromStream [1323]	Selects a NImageReader from a specified stream.
	NImageFormatSelectReaderFromFile [1324]	Selects a NImageReader from a specified file.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NImageFormatSelectReaderFromMemory [1324]	Selects a NImageReader from a specified memory buffer.
	HNImageFormat [1324]	Handle to image format.

Macros

Name	Description
NImageFormatGetDefaultFileExtensionA [1325]	Gets the default file extension for the specified format.
NImageFormatGetDefaultFileExtensionEx [1325]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetDefaultFileExtensionEx function.
NImageFormatGetDefaultFileExtensionW [1325]	Gets the default file extension for the specified format.
NImageFormatGetDefaultInternetMediaType [1326]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetDefaultInternetMediaType function.
NImageFormatGetFileFilterA [1326]	Gets the default file filter for the specified format.
NImageFormatGetFileFilterEx [1326]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetFileFilterEx function.
NImageFormatGetFileFilterW [1326]	Gets the default file filter for the specified format.
NImageFormatGetInternetMediaType [1327]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetInternetMediaType function.
NImageFormatGetNameA [1327]	Gets the name of the specified image format.
NImageFormatGetNameEx [1327]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetNameEx function.
NImageFormatGetNameW [1327]	Gets the name of the specified image format.
NImageFormatOpenReaderFromFile [1328]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatOpenReaderFromFile function.
NImageFormatOpenWriterToFile [1328]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatOpenWriterToFile function.
NImageFormatSelectByInternetMediaType [1328]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatSelectByInternetMediaType function.
NImageFormatSelectEx [1328]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatSelectEx function.
NImageFormatSelectReaderFromFile [1328]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatSelectReaderFromFile function.

7.1.5.9.1 Functions

7.1.5.9.1.1 NImageFormatCanRead Function

Retrieves a value indicating whether the image format supports reading.

C++

```
NResult N_API NImageFormatCanRead(HNImageFormat hImageFormat, NBool * pValue);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
NBool * pValue	[out] Pointer to NBool [1004] that receives value indicating whether the image format supports reading.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when hImageFormat or pValue [870] is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.2 NImageFormatCanWrite Function

Retrieves a value indicating whether the image format supports writing.

C++

```
NResult N_API NImageFormatCanWrite(HNImageFormat hImageFormat, NBool * pValue);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
NBool * pValue	[out] Pointer to NBool [1004] that receives value indicating whether the image format supports writing.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* or *pValue [870]* is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.3 NImageFormatCanWriteMultiple Function

Retrieves a value indicating whether the image format supports writing of multiple images.

C++

```
NResult N_API NImageFormatCanWriteMultiple(HNImageFormat hImageFormat, NBool * pValue);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
NBool * pValue	[out] Pointer to NBool [1004] that receives value indicating whether the image format supports writing of multiple images.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* or *pValue [870]* is NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NImageFormat Unit [1299]

7.1.5.9.1.4 NImageFormatCreateInfo Function

Creates NImageInfo for the specified image with a specified format.

C++

```
NResult N_API NImageFormatCreateInfo(HNImageFormat hImageFormat, HNImage hImage, NUInt
flags, HNImageInfo * phInfo);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] A handle to HNImageFormat [1324].
HNImage hImage	[in] A handle to HNImage [1276].
NUInt flags	[in] Flags controlling function.
HNImageInfo * phInfo	[out] A pointer to the created HNImageInfo.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.5 NImageFormatGetBmp Function

Retrieves BMP image format.

C++

```
NResult N_API NImageFormatGetBmp(HNImageFormat * phValue);
```

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when phValue is NULL [1075].

Remarks

This function is deprecated. Use NImageFormatGetBmpEx [1304] instead.

Module

NImageFormat Unit [1299]

7.1.5.9.1.6 NImageFormatGetDefaultFileExtensionEx Function

Retrieves default extension of the image format.

C++

```
NResult N_API NImageFormatGetDefaultFileExtensionEx(HNImageFormat hImageFormat, NChar *
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
NChar * szValue	[out] Pointer to string that receives default file extension of the image format. Can be NULL [1075].
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.7 NImageFormatGetBmpEx Function

Retrieves BMP image format.

C++

```
NResult N_API NImageFormatGetBmpEx(HNImageFormat * phValue);
```

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when phValue is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.8 NImageFormatGetFormat Function

Retrieves supported image format with the specified index.

C++

```
NResult N_API NImageFormatGetFormat(NInt index, HNImageFormat * phValue);
```

Parameters

Parameters	Description
NInt index	[in] Specifies zero-based supported image format index to retrieve.
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *index* is less that zero or greater than or equal to supported image format count.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Remarks

This function is deprecated. Use NImageFormatGetFormatEx [1306] instead.

Module

NImageFormat Unit [1299]

7.1.5.9.1.9 NImageFormatGetDefaultInternetMediaType Function

Gets the default internet media type of the specified image format.

C++

```
NResult N_API NImageFormatGetDefaultInternetMediaType(HNImageFormat hImageFormat, NChar *
szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to the image format.
NChar * szValue	[out] Pointer to the default internet media type.
NInt valueSize	[in] Size of the default internet media type value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.10 NImageFormatGetFileFilterEx Function

Retrieves file filter of the image format.

C++

```
NResult N_API NImageFormatGetFileFilterEx(HNImageFormat hImageFormat, NChar * szValue, NInt
valueSize);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
NChar * szValue	[out] Pointer to the string that receives file filter of the image format. Can be NULL [1075].
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

Return value	Description
N_OK [433]	If the function succeeds and *szValue* is not NULL [1075], the return value is N_OK [433].
Length of a string	If the function succeeds and *szValue* is NULL [1075], the return value is length of the string (not including the NULL [1075]-terminator) *szValue* should point to.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* is NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NImageFormat Unit [1299]

7.1.5.9.1.11 NImageFormatGetFormatCount Function

Retrieves number of supported image formats.

C++

```
NResult N_API NImageFormatGetFormatCount(NInt * pValue);
```

Parameters

Parameters	Description
NInt * pValue	[out] Pointer to NInt [1006] that receives number of supported image formats.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pValue [870]* is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.12 NImageFormatGetFormatEx Function

Retrieves supported image format with the specified index.

C++

```
NResult N_API NImageFormatGetFormatEx(NInt index, HNImageFormat * phValue);
```

Parameters

Parameters	Description
NInt index	[in] Specifies zero-based supported image format index to retrieve.
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *index* is less that zero or greater than or equal to supported image format count.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.13 NImageFormatGetIHead Function

Retrieves NIST IHead [1201] image format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NImageFormatGetIHead(HNImageFormat * phValue);
```

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Remarks

This function is deprecated. Use NImageFormatGetIHeadEx [1307] instead.

Module

NImageFormat Unit [1299]

7.1.5.9.1.14 NImageFormatGetJpeg Function

Retrieves JPEG image format.

C++

```
NResult N_API NImageFormatGetJpeg(HNImageFormat * phValue);
```

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Remarks

This function is deprecated. Use NImageFormatGetJpegEx [1311] instead.

Module

NImageFormat Unit [1299]

7.1.5.9.1.15 NImageFormatGetIHeadEx Function

Retrieves NIST IHead [1201] image format.

C++

```
NResult N_API NImageFormatGetIHeadEx(HNImageFormat * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.16 NImageFormatGetJpeg2K Function

Retrieves JPEG2K image format.

C++

```
NResult N_API NImageFormatGetJpeg2K(HNImageFormat * phValue);
```

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Remarks

This function is deprecated. Use NImageFormatGetJpeg2KEx [1310] instead.

Module

NImageFormat Unit [1299]

7.1.5.9.1.17 NImageFormatGetInternetMediaType Function

Retrieves Internet media type.

C++

```
NResult N_API NImageFormatGetInternetMediaType(HNImageFormat hImageFormat, NChar * szValue,
NInt valueSize);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] A handle to the HNImageFormat [1324].
NChar * szValue	[out] A pointer to NChar [1005] containing the name the Internet media type.
NInt valueSize	[in] Size of NChar [1005].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.18 NImageFormatGetNameEx Function

Retrieves name of the image format.

C++

```
NResult N_API NImageFormatGetNameEx(HNImageFormat hImageFormat, NChar * szValue, NInt
valueSize);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
NChar * szValue	[out] Pointer to the string that receives name of the image format. Can be NULL [1075].
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

Return value	Description
N_OK [433]	If the function succeeds and *szValue* is not NULL [1075], the return value is N_OK [433].
Length of a string	If the function succeeds and *szValue* is NULL [1075], the return value is length of the string (not including the NULL [1075]-terminator) *szValue* should point to.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.19 NImageFormatGetPng Function

Retrieves PNG image format.

C++

```
NResult N_API NImageFormatGetPng(HNImageFormat * phValue);
```

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Remarks

This function is deprecated. Use NImageFormatGetPngEx [1312] instead.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NImageFormat Unit [1299]

7.1.5.9.1.20 NImageFormatGetJpeg2KEx Function

Retrieves JPEG2K image format.

C++

```
NResult N_API NImageFormatGetJpeg2KEx(HNImageFormat * phValue);
```

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.21 NImageFormatOpenFile Function

Opens image file with specified file name and image format.

C++

```
NResult N_API NImageFormatOpenFile(HNImageFormat hImageFormat, const NChar * szFileName,
HNImageFile * phImageFile);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
const NChar * szFileName	[in] File name to open.
HNImageFile * phImageFile	[out] Pointer to HNImageFile [1298] that receives handle to opened image file.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* or *phImageFile* is NULL [1075].
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when image format specified by *hImageFormat* does not support reading.

Module

NImageFormat Unit [1299]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.9.1.22 NImageFormatGetJpegEx Function

Retrieves JPEG image format.

C++

```
NResult N_API NImageFormatGetJpegEx(HNImageFormat * phValue);
```

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.23 NImageFormatGetTiff Function

Retrieves TIFF image format.

C++

```
NResult N_API NImageFormatGetTiff(HNImageFormat * phValue);
```

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Remarks

This function is deprecated. Use NImageFormatGetTiffEx [1313] instead.

Module

NImageFormat Unit [1299]

7.1.5.9.1.24 NImageFormatGetWsq Function

Retrieves WSQ image format.

C++

```
NResult N_API NImageFormatGetWsq(HNImageFormat * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Remarks

This function is deprecated. Use NImageFormatGetWsqEx [1315] instead.

Module

NImageFormat Unit [1299]

7.1.5.9.1.25 NImageFormatGetPngEx Function

Retrieves PNG image format.

C++

```
NResult N_API NImageFormatGetPngEx(HNImageFormat * phValue);
```

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.26 NImageFormatLoadImageFromFile Function

Loads image from file of specified image format.

C++

```
NResult N_API NImageFormatLoadImageFromFile(HNImageFormat hImageFormat, const NChar *
szFileName, HNImage * phImage);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
const NChar * szFileName	[in] Points to string that specifies file name.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat*, *szFileName* or *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file specified by *szFileName* is invalid.
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when image format specified by *hImageFormat* does not support reading.

Module

NImageFormat Unit [1299]

7.1.5.9.1.27 NImageFormatSaveImagesToFile Function

Saves specified number of images to the file in the specified format.

C++

```
NResult N_API NImageFormatSaveImagesToFile(HNImageFormat hImageFormat, NInt imageCount,
HNImage * arhImages, const NAChar * szFileName);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
NInt imageCount	[in] Number of images to save.
HNImage * arhImages	[out] Handles to all images that should be saved.
const NAChar * szFileName	[in] Points to memory buffer that contains file names for all images to save.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* or *arhImages* is NULL [1075].
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when image format specified by *hImageFormat* does not support writing.
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when image *imageCount* value is equal or less than zero.

Module

NImageFormat Unit [1299]

7.1.5.9.1.28 NImageFormatGetTiffEx Function

Retrieves TIFF image format.

C++

```
NResult N_API NImageFormatGetTiffEx(HNImageFormat * phValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.29 NImageFormatSaveImagesToMemory Function

Saves specified number of images to a memory buffer in the specified format.

C++

```
NResult N_API NImageFormatSaveImagesToMemory(HNImageFormat hImageFormat, NInt imageCount,
HNImage * arhImages, void * * ppBuffer, NSizeType * pBufferSize);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
NInt imageCount	[in] Number of images to save.
HNImage * arhImages	[out] Handles to all images that should be saved.
void * * ppBuffer	[out] Pointer to void * that receives pointer to allocated memory buffer.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* or *arhImages* is NULL [1075].
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when image format specified by *hImageFormat* does not support writing.
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when image *imageCount* value is equal or less than zero.

Module

NImageFormat Unit [1299]

7.1.5.9.1.30 NImageFormatLoadImageFromMemory Function

Loads image from the memory buffer containing file of the specified format.

C++

```
NResult N_API NImageFormatLoadImageFromMemory(HNImageFormat hImageFormat, const void *
pBuffer, NSizeType bufferSize, HNImage * phImage);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
const void * pBuffer	[out] Pointer to memory buffer.
NSizeType bufferSize	[in] Size of memory buffer.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* or *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of the file contained in memory buffer specified by *pBuffer* is invalid for the specified image format.
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when image format specified by *hImageFormat* does not support reading.

Module

NImageFormat Unit [1299]

7.1.5.9.1.31 NImageFormatGetWsqEx Function

Retrieves WSQ image format.

C++

```
NResult N_API NImageFormatGetWsqEx(HNImageFormat * phValue);
```

Parameters

Parameters	Description
HNImageFormat * phValue	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *phValue* is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.32 NImageFormatLoadImageFromStream Function

Loads image from the memory stream containing file of the specified format.

C++

```
NResult N_API NImageFormatLoadImageFromStream(HNImageFormat hImageFormat, HNStream hStream,
HNImage * phImage);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
HNStream hStream	[in] Pointer to memory stream.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of the file contained in memory stream specified by buffer is invalid for the specified image format.
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when image format specified by *hImageFormat* does not support reading.

Module

NImageFormat Unit [1299]

7.1.5.9.1.33 NImageFormatSaveImagesToStream Function

Saves specified number of images to a stream in the specified format.

C++

```
NResult N_API NImageFormatSaveImagesToStream(HNImageFormat hImageFormat, NInt imageCount,
HNImage * arhImages, HNStream hStream);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
NInt imageCount	[in] Number of images to save.
HNImage * arhImages	[out] Handles to all images that should be saved.
HNStream hStream	[in] Handle to the stream to which the images are to be saved.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* or *arhImages* is NULL [1075].
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when image format specified by *hImageFormat* does not support writing.
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when image *imageCount* value is equal or less than zero.

Module

NImageFormat Unit [1299]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.9.1.34 NImageFormatOpenFileFromMemory Function

Opens image file from the memory buffer containing file of specified format.

C++

```
NResult N_API NImageFormatOpenFileFromMemory(HNImageFormat hImageFormat, const void *
pBuffer, NSizeType bufferSize, HNImageFile * phImageFile);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
const void * pBuffer	[out] Pointer to memory buffer.
NSizeType bufferSize	[in] Size of memory buffer.
HNImageFile * phImageFile	[out] Pointer to HNImageFile [1298] that receives handle to opened image file.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* or *phImageFile* is NULL [1075] - or - *buffer* is NULL [1075] and *bufferSize* is not equal to zero.
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file contained in buffer specified by *pBuffer* is invalid for specified image format.
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when image format specified by *hImageFormat* does not support reading.

Module

NImageFormat Unit [1299]

7.1.5.9.1.35 NImageFormatOpenFileFromStream Function

Opens image file from the memory stream containing file of specified format.

C++

```
NResult N_API NImageFormatOpenFileFromStream(HNImageFormat hImageFormat, HNStream hStream,
HNImageFile * phImageFile);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
HNStream hStream	[in] Handle to memory stream.
HNImageFile * phImageFile	[out] Pointer to HNImageFile [1298] that receives handle to opened image file.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat* or *pHImageFile* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file contained in buffer specified by buffer is invalid for specified image format.
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when image format specified by *hImageFormat* does not support reading.

Module

NImageFormat Unit [1299]

7.1.5.9.1.36 NImageFormatOpenReaderFromFile Function

Opens NImageReader from a specified file.

C++

```
NResult N_API NImageFormatOpenReaderFromFile(HNImageFormat hImageFormat, const NChar *
szFileName, NUInt flags, HNImageReader * phReader);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] A handle to the image format.
const NChar * szFileName	[in] A pointer to the file name.
NUInt flags	[in] Flags controlling function.
HNImageReader * phReader	[out] A pointer to the HNImageReader.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.37 NImageFormatOpenReaderFromStream Function

Opens NImageReader from a specified stream.

C++

```
NResult N_API NImageFormatOpenReaderFromStream(HNImageFormat hImageFormat, HNStream
hStream, NUInt flags, HNImageReader * phReader);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] A handle to the image format.
HNStream hStream	[in] A handle to the specified stream.
NUInt flags	[in] Flags controlling function.
HNImageReader * phReader	[out] A pointer to the HNImageReader.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.9.1.38 NImageFormatSaveImageToFile Function

Saves image to the file in the specified format.

C++

```
NResult N_API NImageFormatSaveImageToFile(HNImageFormat hImageFormat, HNImage hImage, const NChar * szFileName);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
HNImage hImage	[in] Handle to image.
const NChar * szFileName	[in] Points to string that specifies file name.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat*, *hImage* or *szFileName* is NULL [1075].
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when image format specified by *hImageFormat* does not support writing.

Module

NImageFormat Unit [1299]

7.1.5.9.1.39 NImageFormatOpenWriterToStream Function

Opens NImageWriter to a specified stream.

C++

```
NResult N_API NImageFormatOpenWriterToStream(HNImageFormat hImageFormat, HNStream hStream, NUInt flags, HNImageWriter * phWriter);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] A handle to the image format.
HNStream hStream	[in] A handle to the specified stream.
NUInt flags	[in] Flags controlling function.
HNImageWriter * phWriter	[out] A pointer to the HNImageWriter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.40 NImageFormatOpenReaderFromMemory Function

Opens NImageReader from a specified memory buffer.

C++

```
NResult N_API NImageFormatOpenReaderFromMemory(HNImageFormat hImageFormat, const void * pBuffer, NSizeType bufferSize, NUInt flags, HNImageReader * phReader);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] A handle to the image format.
const void * pBuffer	[in] A pointer to the specified memory buffer.
NSizeType bufferSize	[in] Size of the specified memory buffer.
NUInt flags	[in] Flags controlling function.
HNImageReader * phReader	[out] A pointer to the HNImageReader.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.41 NImageFormatSaveImageToMemory Function

Saves image to the memory buffer in the specified format.

C++

```
NResult N_API NImageFormatSaveImageToMemory(HNImageFormat hImageFormat, HNImage hImage,
void * * ppBuffer, NSizeType * pBufferSize);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
HNImage hImage	[in] Handle to image.
void * * ppBuffer	[out] Pointer to void * that receives pointer to allocated memory buffer.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat*, *hImage*, *ppBuffer* or *pBufferSize* is NULL [1075].
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. This error code is returned when there was not enough memory to allocate memory buffer.
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when image format specified by *hImageFormat* does not support writing.

Remarks

Memory buffer allocated by this function must be deallocated by NFree [506] function when it is no longer needed.

Module

NImageFormat Unit [1299]

7.1.5.9.1.42 NImageFormatSaveImageToStream Function

Saves image to the memory stream in the specified format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NImageFormatSaveImageToStream(HNImageFormat hImageFormat, HNImage hImage,
HNStream hStream);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] Handle to image format.
HNImage hImage	[in] Handle to image.
HNStream hStream	[in] Handle to memory stream that contains image.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImageFormat*, *hImage* or *hStream* is NULL [1075].
N_E_NOT_SUPPORTED [431]	If the function fails, this error can be returned. This error code is returned when image format specified by *hImageFormat* does not support writing.

Module

NImageFormat Unit [1299]

7.1.5.9.1.43 NImageFormatOpenWriterToFile Function

Opens NImageWriter to a specified file.

C++

```
NResult N_API NImageFormatOpenWriterToFile(HNImageFormat hImageFormat, const NChar *
szFileName, NUInt flags, HNImageWriter * phWriter);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] A handle to the image format.
const NChar * szFileName	[in] A pointer to the file name.
NUInt flags	[in] Flags controlling function.
HNImageWriter * phWriter	[out] A pointer to the HNImageWriter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.44 NImageFormatSelect Function

Retrieves supported image format registered with file extension of specified file name and supporting reading/writing as specified.

C++

```
NResult N_API NImageFormatSelect(const NChar * szFileName, NFileAccess fileAccess,
HNImageFormat * phImageFormat);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
const NChar * szFileName	[in] Points to string that specifies file name.
NFileAccess fileAccess	[in] Specifies that image format should support reading, writing or both.
HNImageFormat * phImageFormat	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *fileAccess* value is invalid.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *szFileName* or *phImageFormat* is NULL [1075].

Remarks

If none of supported image formats that supports reading/writing as specified by *fileAccess* is registered with file extension of *szFileName* then handle returned via *phImageFormat* is NULL [1075].

This function is deprecated. Use NImageFormatSelectEx [1322] instead.

Module

NImageFormat Unit [1299]

7.1.5.9.1.45 NImageFormatSelectByInternetMediaType Function

Retrieves supported image format registered with Internet media type of specified file name and supporting reading/writing as specified.

C++

```
NResult N_API NImageFormatSelectByInternetMediaType(const NChar * szInternetMediaType,
NFileAccess fileAccess, HNImageFormat * phImageFormat);
```

Parameters

Parameters	Description
const NChar * szInternetMediaType	[in] A pointer to NChar [1005] containing Internet media type.
NFileAccess fileAccess	[in] Specifies that image format should support reading, writing or both.
HNImageFormat * phImageFormat	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.46 NImageFormatSelectEx Function

Retrieves supported image format registered with file extension of specified file name and supporting reading/writing as specified.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NImageFormatSelectEx(const NChar * szFileName, NFileAccess fileAccess,
HNImageFormat * phImageFormat);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Points to string that specifies file name.
NFileAccess fileAccess	[in] Specifies that image format should support reading, writing or both.
HNImageFormat * phImageFormat	[out] Pointer to HNImageFormat [1324] that receives handle to image format.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT [427]	If the function fails, this error can be returned. This error code is returned when *fileAccess* value is invalid.
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *szFileName* or *phImageFormat* is NULL [1075].

Module

NImageFormat Unit [1299]

7.1.5.9.1.47 NImageFormatOpenWriterToMemory Function

Opens NImageWriter to a memory buffer.

C++

```
NResult N_API NImageFormatOpenWriterToMemory(HNImageFormat hImageFormat, NUInt flags,
HNImageWriter * phWriter);
```

Parameters

Parameters	Description
HNImageFormat hImageFormat	[in] A handle to the image format.
NUInt flags	[in] Flags controlling function.
HNImageWriter * phWriter	[out] A pointer to the HNImageWriter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.48 NImageFormatSelectReaderFromStream Function

Selects a NImageReader from a specified stream.

C++

```
NResult N_API NImageFormatSelectReaderFromStream(HNStream hStream, NUInt flags,
HNImageReader * phReader);
```

Parameters

Parameters	Description
HNStream hStream	[in] A handle to the specified stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt flags	[in] Flags controlling function.
HNImageReader * phReader	[out] A pointer to the HNImageReader.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.49 NImageFormatSelectReaderFromFile Function

Selects a NImageReader from a specified file.

C++

```
NResult N_API NImageFormatSelectReaderFromFile(const NChar * szFileName, NUInt flags,
HNImageReader * phReader);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] A pointer to the file name.
NUInt flags	[in] Flags controlling function.
HNImageReader * phReader	[out] A pointer to the HNImageReader.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.50 NImageFormatSelectReaderFromMemory Function

Selects a NImageReader from a specified memory buffer.

C++

```
NResult N_API NImageFormatSelectReaderFromMemory(const void * pBuffer, NSizeType
bufferSize, NUInt flags, HNImageReader * phReader);
```

Parameters

Parameters	Description
const void * pBuffer	[in] A pointer to the specified memory buffer.
NSizeType bufferSize	[in] Size of the specified memory buffer.
NUInt flags	[in] Flags controlling function.
HNImageReader * phReader	[out] A pointer to the HNImageReader.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.1.51 HNImageFormat

Handle to image format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NImageFormat Unit [1299]

7.1.5.9.2 Macros

7.1.5.9.2.1 NImageFormatGetDefaultFileExtensionA Macro

Gets the default file extension for the specified format.

C++

```
#define NImageFormatGetDefaultFileExtensionA(hImageFormat, szValue)
NImageFormatGetDefaultFileExtensionExA(hImageFormat, szValue, N_INT_MAX)
```

Parameters

Parameters	Description
hImageFormat	[in] Handle to the image format.
szValue	[out] Pointer to the file extension.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.2.2 NImageFormatGetDefaultFileExtensionEx Macro

Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetDefaultFileExtensionEx function.

C++

```
#define NImageFormatGetDefaultFileExtensionEx
N_FUNC_AW(NImageFormatGetDefaultFileExtensionEx)
```

Module

NImageFormat Unit [1299]

7.1.5.9.2.3 NImageFormatGetDefaultFileExtensionW Macro

Gets the default file extension for the specified format.

C++

```
#define NImageFormatGetDefaultFileExtensionW(hImageFormat, szValue)
NImageFormatGetDefaultFileExtensionExW(hImageFormat, szValue, N_INT_MAX)
```

Parameters

Parameters	Description
hImageFormat	[in] Handle to the image format.
szValue	[out] Pointer to the file extension.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.9.2.4 NImageFormatGetDefaultInternetMediaType Macro

Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetDefaultInternetMediaType function.

C++

```
#define NImageFormatGetDefaultInternetMediaType
N_FUNC_AW(NImageFormatGetDefaultInternetMediaType)
```

Module

NImageFormat Unit [1299]

7.1.5.9.2.5 NImageFormatGetFileFilterA Macro

Gets the default file filter for the specified format.

C++

```
#define NImageFormatGetFileFilterA(hImageFormat, szValue)
NImageFormatGetFileFilterExA(hImageFormat, szValue, N_INT_MAX)
```

Parameters

Parameters	Description
hImageFormat	[in] Handle to the image format.
szValue	[out] Pointer to the file filter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.2.6 NImageFormatGetFileFilterEx Macro

Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetFileFilterEx function.

C++

```
#define NImageFormatGetFileFilterEx N_FUNC_AW(NImageFormatGetFileFilterEx)
```

Module

NImageFormat Unit [1299]

7.1.5.9.2.7 NImageFormatGetFileFilterW Macro

Gets the default file filter for the specified format.

C++

```
#define NImageFormatGetFileFilterW(hImageFormat, szValue)
NImageFormatGetFileFilterExW(hImageFormat, szValue, N_INT_MAX)
```

Parameters

Parameters	Description
hImageFormat	[in] Handle to the image format.
szValue	[out] Pointer to the file filter.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NImageFormat Unit [1299]

7.1.5.9.2.8 NImageFormatGetInternetMediaType Macro

Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetInternetMediaType function.

C++

```
#define NImageFormatGetInternetMediaType N_FUNC_AW(NImageFormatGetInternetMediaType)
```

Module

NImageFormat Unit [1299]

7.1.5.9.2.9 NImageFormatGetNameA Macro

Gets the name of the specified image format.

C++

```
#define NImageFormatGetNameA(hImageFormat, szValue) NImageFormatGetNameExA(hImageFormat,
szValue, N_INT_MAX)
```

Parameters

Parameters	Description
hImageFormat	[in] Handle to the image format.
szValue	[out] Pointer to the format name.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.2.10 NImageFormatGetNameEx Macro

Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetNameEx function.

C++

```
#define NImageFormatGetNameEx N_FUNC_AW(NImageFormatGetNameEx)
```

Module

NImageFormat Unit [1299]

7.1.5.9.2.11 NImageFormatGetNameW Macro

Gets the name of the specified image format.

C++

```
#define NImageFormatGetNameW(hImageFormat, szValue) NImageFormatGetNameExW(hImageFormat,
szValue, N_INT_MAX)
```

Parameters

Parameters	Description
hImageFormat	[in] Handle to the image format.
szValue	[out] Pointer to the format name.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImageFormat Unit [1299]

7.1.5.9.2.12 NImageFormatOpenReaderFromFile Macro

Picks a Unicode (W) or ANSCII (A) version of the NImageFormatOpenReaderFromFile function.

C++

```
#define NImageFormatOpenReaderFromFile N_FUNC_AW(NImageFormatOpenReaderFromFile)
```

Module

NImageFormat Unit [1299]

7.1.5.9.2.13 NImageFormatOpenWriterToFile Macro

Picks a Unicode (W) or ANSCII (A) version of the NImageFormatOpenWriterToFile function.

C++

```
#define NImageFormatOpenWriterToFile N_FUNC_AW(NImageFormatOpenWriterToFile)
```

Module

NImageFormat Unit [1299]

7.1.5.9.2.14 NImageFormatSelectByInternetMediaType Macro

Picks a Unicode (W) or ANSCII (A) version of the NImageFormatSelectByInternetMediaType function.

C++

```
#define NImageFormatSelectByInternetMediaType
N_FUNC_AW(NImageFormatSelectByInternetMediaType)
```

Module

NImageFormat Unit [1299]

7.1.5.9.2.15 NImageFormatSelectEx Macro

Picks a Unicode (W) or ANSCII (A) version of the NImageFormatSelectEx function.

C++

```
#define NImageFormatSelectEx N_FUNC_AW(NImageFormatSelectEx)
```

Module

NImageFormat Unit [1299]

7.1.5.9.2.16 NImageFormatSelectReaderFromFile Macro

Picks a Unicode (W) or ANSCII (A) version of the NImageFormatSelectReaderFromFile function.

C++

```
#define NImageFormatSelectReaderFromFile N_FUNC_AW(NImageFormatSelectReaderFromFile)
```

Module

NImageFormat Unit [1299]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.9.3 Files

7.1.5.9.3.1 NImageFormat.h

Header file for NImageFormat [1299] module. Provides functionality for loading and saving images in format-neutral way.

Functions

	Name	Description
	NImageFormatCanRead [1301]	Retrieves a value indicating whether the image format supports reading.
	NImageFormatCanWrite [1302]	Retrieves a value indicating whether the image format supports writing.
	NImageFormatCanWriteMultiple [1302]	Retrieves a value indicating whether the image format supports writing of multiple images.
	NImageFormatCreateInfo [1303]	Creates NImageInfo for the specified image with a specified format.
	NImageFormatGetBmp [1303]	Retrieves BMP image format.
	NImageFormatGetBmpEx [1304]	Retrieves BMP image format.
	NImageFormatGetDefaultFileExtensionEx [1303]	Retrieves default extension of the image format.
	NImageFormatGetDefaultInternetMediaType [1305]	Gets the default internet media type of the specified image format.
	NImageFormatGetFileFilterEx [1305]	Retrieves file filter of the image format.
	NImageFormatGetFormat [1304]	Retrieves supported image format with the specified index.
	NImageFormatGetFormatCount [1306]	Retrieves number of supported image formats.
	NImageFormatGetFormatEx [1306]	Retrieves supported image format with the specified index.
	NImageFormatGetIHead [1306]	Retrieves NIST IHead [1201] image format.
	NImageFormatGetIHeadEx [1307]	Retrieves NIST IHead [1201] image format.
	NImageFormatGetInternetMediaType [1308]	Retrieves Internet media type.
	NImageFormatGetJpeg [1307]	Retrieves JPEG image format.
	NImageFormatGetJpeg2K [1308]	Retrieves JPEG2K image format.
	NImageFormatGetJpeg2KEx [1310]	Retrieves JPEG2K image format.
	NImageFormatGetJpegEx [1311]	Retrieves JPEG image format.
	NImageFormatGetNameEx [1309]	Retrieves name of the image format.
	NImageFormatGetPng [1309]	Retrieves PNG image format.
	NImageFormatGetPngEx [1312]	Retrieves PNG image format.
	NImageFormatGetTiff [1311]	Retrieves TIFF image format.
	NImageFormatGetTiffEx [1313]	Retrieves TIFF image format.
	NImageFormatGetWsq [1311]	Retrieves WSQ image format.
	NImageFormatGetWsqEx [1315]	Retrieves WSQ image format.
	NImageFormatLoadImageFromFile [1312]	Loads image from file of specified image format.
	NImageFormatLoadImageFromMemory [1314]	Loads image from the memory buffer containing file of the specified format.
	NImageFormatLoadImageFromStream [1315]	Loads image from the memory stream containing file of the specified format.
	NImageFormatOpenFile [1310]	Opens image file with specified file name and image format.
	NImageFormatOpenFileFromMemory [1317]	Opens image file from the memory buffer containing file of specified format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NImageFormatOpenFileFromStream [1317]	Opens image file from the memory stream containing file of specified format.
	NImageFormatOpenReaderFromFile [1318]	Opens NImageReader from a specified file.
	NImageFormatOpenReaderFromMemory [1319]	Opens NImageReader from a specified memory buffer.
	NImageFormatOpenReaderFromStream [1318]	Opens NImageReader from a specified stream.
	NImageFormatOpenWriterToFile [1321]	Opens NImageWriter to a specified file.
	NImageFormatOpenWriterToMemory [1323]	Opens NImageWriter to a memory buffer.
	NImageFormatOpenWriterToStream [1319]	Opens NImageWriter to a specified stream.
	NImageFormatSaveImagesToFile [1313]	Saves specified number of images to the file in the specified format.
	NImageFormatSaveImagesToMemory [1314]	Saves specified number of images to a memory buffer in the specified format.
	NImageFormatSaveImagesToStream [1316]	Saves specified number of images to a stream in the specified format.
	NImageFormatSaveImageToFile [1319]	Saves image to the file in the specified format.
	NImageFormatSaveImageToMemory [1320]	Saves image to the memory buffer in the specified format.
	NImageFormatSaveImageToStream [1320]	Saves image to the memory stream in the specified format.
	NImageFormatSelect [1321]	Retrieves supported image format registered with file extension of specified file name and supporting reading/writing as specified.
	NImageFormatSelectByInternetMediaType [1322]	Retrieves supported image format registered with Internet media type of specified file name and supporting reading/writing as specified.
	NImageFormatSelectEx [1322]	Retrieves supported image format registered with file extension of specified file name and supporting reading/writing as specified.
	NImageFormatSelectReaderFromFile [1324]	Selects a NImageReader from a specified file.
	NImageFormatSelectReaderFromMemory [1324]	Selects a NImageReader from a specified memory buffer.
	NImageFormatSelectReaderFromStream [1323]	Selects a NImageReader from a specified stream.

Macros

Name	Description
NImageFormatGetDefaultFileExtensionA [1325]	Gets the default file extension for the specified format.
NImageFormatGetDefaultFileExtensionEx [1325]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetDefaultFileExtensionEx function.
NImageFormatGetDefaultFileExtensionW [1325]	Gets the default file extension for the specified format.
NImageFormatGetDefaultInternetMediaType [1326]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetDefaultInternetMediaType function.
NImageFormatGetFileFilterA [1326]	Gets the default file filter for the specified format.
NImageFormatGetFileFilterEx [1326]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetFileFilterEx function.
NImageFormatGetFileFilterW [1326]	Gets the default file filter for the specified format.
NImageFormatGetInternetMediaType [1327]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetInternetMediaType function.
NImageFormatGetNameA [1327]	Gets the name of the specified image format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NImageFormatGetNameEx [1327]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatGetNameEx function.
NImageFormatGetNameW [1327]	Gets the name of the specified image format.
NImageFormatOpenReaderFromFile [1328]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatOpenReaderFromFile function.
NImageFormatOpenWriterToFile [1328]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatOpenWriterToFile function.
NImageFormatSelectByInternetMediaType [1328]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatSelectByInternetMediaType function.
NImageFormatSelectEx [1328]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatSelectEx function.
NImageFormatSelectReaderFromFile [1328]	Picks a Unicode (W) or ANSCII (A) version of the NImageFormatSelectReaderFromFile function.

Module

NImageFormat Unit [1299]

7.1.5.10 NImages Unit

Provides library registration and other additional functionality.

Files

Name	Description
NImages.h [1333]	Header file for NImages module. Provides library registration and other additional functionality.

Functions

	Name	Description
	NImagesGetGrayscaleColorWrapperEx [1331]	Creates color wrapper for a grayscale image.
	NImagesRecolorImage [1332]	Recolors the specified image.

Macros

Name	Description
NImagesGetInfoA [1332]	Retrieves NImages library information.
NImagesGetInfoW [1333]	Retrieves NImages library information.
NImagesModuleOf [1333]	Returns handle of the module that can be used with functions from NModule [516] unit to get the information about the module.

7.1.5.10.1 Functions

7.1.5.10.1.1 NImagesGetGrayscaleColorWrapperEx Function

Creates color wrapper for a grayscale image.

C++

```
NResult N_API NImagesGetGrayscaleColorWrapperEx(HNImage hImage, const struct NRgb_ *
pMinColor, const struct NRgb_ * pMaxColor, HNImage * phDstImage);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const struct NRgb_ * pMinColor	[out] Specifies color to be used for black color.
const struct NRgb_ * pMaxColor	[out] Specifies color to be used for white color.
HNImage * phDstImage	[out] Pointer to HNImage [1276] that receives handle to created image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT [427]	This error code is returned when image specified by *hImage* has non-grayscale pixel format (not *npfGrayscale [1384]* or *npfMonochrome [1384]*).
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *phDstImage* is NULL [1075].

Remarks

Created image must be deleted using NObjectFree [542] function.

Created image is a thin wrapper for specified grayscale image. Therefore *hImage* must not be freed before created image.

Gray values in source image are replaced with according RGB values from range [minColor, maxColor] in created image.

Module

NImages Unit [1331]

7.1.5.10.1.2 NImagesRecolorImage Function

Recolors the specified image.

C++

```
NResult N_API NImagesRecolorImage(HNImage hImage, NPixelFormat_ pixelFormat, HNArray
hMinValue, HNArray hMaxValue, HNImage * phDstImage);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to the image.
NPixelFormat_ pixelFormat	[in] Format of the pixels.
HNArray hMinValue	[in] Handle to the minimum pixel value in the image.
HNArray hMaxValue	[in] Handle to the maximum pixel value in the image.
HNImage * phDstImage	[out] Pointer to a handle to the recolored image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NImages Unit [1331]

7.1.5.10.2 Macros

7.1.5.10.2.1 NImagesGetInfoA Macro

Retrieves NImages library information.

C++

```
#define NImagesGetInfoA NMediaGetInfoA
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NImages Unit [1331]

7.1.5.10.2.2 NImagesGetInfoW Macro

Retrieves NImages library information.

C++

```
#define NImagesGetInfoW NMediaGetInfoW
```

Returns

NSucceeded [434] or NFailed [434] macros can be used to detect if the function succeeded or failed.

Module

NImages Unit [1331]

7.1.5.10.2.3 NImagesModuleOf Macro

Returns handle of the module that can be used with functions from NModule [516] unit to get the information about the module.

C++

```
#define NImagesModuleOf NMediaModuleOf
```

Module

NImages Unit [1331]

7.1.5.10.3 Files

7.1.5.10.3.1 NImages.h

Header file for NImages module. Provides library registration and other additional functionality.

Functions

	Name	Description
	NImagesGetGrayscaleColorWrapperEx [1331]	Creates color wrapper for a grayscale image.
	NImagesRecolorImage [1332]	Recolors the specified image.

Macros

Name	Description
NImagesGetInfoA [1332]	Retrieves NImages library information.
NImagesGetInfoW [1333]	Retrieves NImages library information.
NImagesModuleOf [1333]	Returns handle of the module that can be used with functions from NModule [516] unit to get the information about the module.

Module

NImages Unit [1331]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.11 NMediaFormat Unit

Provides functions to retrieve information about media formats

Files

Name	Description
NMediaFormat.h [1336]	Header file for NMediaFormat Unit.

Functions

	Name	Description
	NMediaFormatGetMediaSubtype [1334]	Gets media subtype from specified media format.
	NMediaFormatGetMediaType [1334]	Gets media type from specified NMediaFormat.
	NMediaFormatIsCompatibleWith [1335]	Checks if specified media format is compatible with other media format.
	NMediaFormatSetMediaSubtype [1335]	Sets media subtype for specified media format.
	HNMediaFormat [1336]	Handle to HNMediaFormat.

7.1.5.11.1 Functions

7.1.5.11.1.1 NMediaFormatGetMediaSubtype Function

Gets media subtype from specified media format.

C++

```
NResult N_API NMediaFormatGetMediaSubtype(HNMediaFormat hFormat, NUInt * pValue);
```

Parameters

Parameters	Description
HNMediaFormat hFormat	[in] Handle to NMediaFormat [1334] to retrieve its subtype.
NUInt * pValue	[out] Media format subtype value (numeric constant).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For NAudioFormat one of N_AUDIO_FORMAT_* constants is returned as *pValue [870]*. For NVideoFormat [1395] one of N_VIDEO_FORMAT_* constants is returned as *pValue [870]*. These constants can be converted to string using these functions: NAudioFormatMediaSubtypeToString and NVideoFormatMediaSubtypeToString [1400].

Module

NMediaFormat Unit [1334]

7.1.5.11.1.2 NMediaFormatGetMediaType Function

Gets media type from specified NMediaFormat [1334].

C++

```
NResult N_API NMediaFormatGetMediaType(HNMediaFormat hFormat, NMediaType * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNMediaFormat hFormat	[in] Handle to NMediaFormat [1334] to retrieve its type.
NMediaType * pValue	[out] One of media type values. At the moment only *nmtAudio* and *nmtVideo* can be returned.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaFormat Unit [1334]

7.1.5.11.1.3 NMediaFormatIsCompatibleWith Function

Checks if specified media format is compatible with other media format.

C++

```
NResult N_API NMediaFormatIsCompatibleWith(HNMediaFormat hFormat, HNMediaFormat
hOtherFormat, NBool * pResult);
```

Parameters

Parameters	Description
HNMediaFormat hFormat	[in] Handle to NMediaFormat [1334] to check if it is compatible wither other media format.
HNMediaFormat hOtherFormat	[in] Handle to other media format to check if it is compatible with the first one.
NBool * pResult	[out] Two media formats compatibility check value (NBool [1004]).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaFormat Unit [1334]

7.1.5.11.1.4 NMediaFormatSetMediaSubtype Function

Sets media subtype for specified media format.

C++

```
NResult N_API NMediaFormatSetMediaSubtype(HNMediaFormat hFormat, NUInt value);
```

Parameters

Parameters	Description
HNMediaFormat hFormat	[in] Handle to NMediaFormat [1334] to set its subtype.
NUInt value	[in] Subtype constant to set.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

For NAudioFormat one of N_AUDIO_FORMAT_* constant values should be used as *value* parameter. For NVideoFormat [1395] one of N_VIDEO_FORMAT_* constant values should be used as *value* parameter.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NMediaFormat Unit [1334]

7.1.5.11.1.5 HNMediaFormat

Handle to HNMediaFormat.

Module

NMediaFormat Unit [1334]

7.1.5.11.2 Files

7.1.5.11.2.1 NMediaFormat.h

Header file for NMediaFormat [1334] Unit.

Functions

	Name	Description
	NMediaFormatGetMediaSubtype [1334]	Gets media subtype from specified media format.
	NMediaFormatGetMediaType [1334]	Gets media type from specified NMediaFormat [1334].
	NMediaFormatIsCompatibleWith [1335]	Checks if specified media format is compatible with other media format.
	NMediaFormatSetMediaSubtype [1335]	Sets media subtype for specified media format.

Module

NMediaFormat Unit [1334]

7.1.5.12 NMediaReader Unit

Provides functions which can be used to read media data.

Files

Name	Description
NMediaReader.h [1344]	Header file for NMediaReader Unit.

Functions

	Name	Description
	NMediaReaderCreate [1337]	Creates media reader object.
	NMediaReaderCreateFromFile [1338]	Creates NMediaReader from file.
	NMediaReaderCreateFromUrl [1338]	Creates NMediaReader from URL.
	NMediaReaderGetLength [1339]	Gets file length (100 ns units) when NMediaReader source is file.
	NMediaReaderGetPosition [1340]	Gets NMediaReader position in media source (file).
	NMediaReaderGetSource [1340]	Gets media source (file or device) of the specified NMediaReader.
	NMediaReaderGetState [1340]	Gets NMediaReader state.
	NMediaReaderReadAudioSample [1341]	Reads audio sample using specified NMediaReader.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NMediaReaderIsLive [1341]	Checks if media source which is used by specified NMediaReader is live.
	NMediaReaderReadVideoSample [1342]	Reads video sample using specified NMediaReader.
	NMediaReaderPause [1342]	Pauses specified NMediaReader.
	NMediaReaderSetPosition [1342]	Sets NMediaReader position in media source (file).
	NMediaReaderStart [1343]	Starts NMediaReader.
	NMediaReaderStop [1343]	Stops NMediaReader.
	HNMediaReader [1343]	Handle to NMediaReader

Macros

Name	Description
NMediaReaderCreateFromFile [1344]	Picks a Unicode (W) or ANSCII (A) version of the NMediaReaderFromFile function.
NMediaReaderCreateFromUrl [1344]	Picks a Unicode (W) or ANSCII (A) version of the NMediaReaderFromUrl function.

7.1.5.12.1 Functions

7.1.5.12.1.1 NMediaReaderCreate Function

Creates media reader object.

C++

```
NResult N_API NMediaReaderCreate(HNMediaSource hMediaSource, NMediaType mediaTypes, NBool
isLive, NUInt flags, HNMediaReader * phReader);
```

Parameters

Parameters	Description
HNMediaSource hMediaSource	[in] Handle to media source.
NMediaType mediaTypes	[in] Media types supported by NMediaReader [1336]. Can be *nmtAudio* and *nmtVideo* (only one at a time or both).
NBool isLive	[in] Flag indicating if media source is live. If set to NTrue [1064] media from source will be read in real time (from devices). In this case video or audio samples will be dropped until user calls `NMediaReaderReadAudioSample [1341]` or `NMediaReaderReadVideoSample [1342]` functions. If user calls these functions too quickly they are blocked until new video frame or audio sample appears. If set to NFalse [1041] video frames or audio samples are not dropped and user can read them in any speed (gets all samples/frames). When set to NFalse [1041], NMediaReader [1336] works only with video/audio data saved in files.
NUInt flags	[in] Flags controlling NMediaReader [1336].
HNMediaReader * phReader	[out] Handle to created media reader.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

These flags can be used (for *flags* parameter):

- NMS_DO_NOT_USE_DIRECT_SHOW [1351]
- NMS_DO_NOT_USE_WINDOWS_MEDIA_FOUNDATION [1351]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- NMS_PREFER_DIRECT_SHOW [1351]

Note: these flags work only on Windows OS.

Module

NMediaReader Unit [1336]

7.1.5.12.1.2 NMediaReaderCreateFromFile Function

Creates NMediaReader [1336] from file.

C++

```
NResult N_API NMediaReaderCreateFromFile(const NChar * szFileName, NMediaType mediaTypes,
NBool isLive, NUInt flags, HNMediaReader * phReader);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Zero-terminated string containing file name.
NMediaType mediaTypes	[in] Media types supported by NMediaReader [1336]. Can be *nmtAudio* and *nmtVideo* (only one at a time or both).
NBool isLive	[in] Flag indicating if media source is live. If set to NTrue [1064] media from source will be read in real time (from devices). In this case video or audio samples will be dropped until user calls `NMediaReaderReadAudioSample` [1341] or `NMediaReaderReadVideoSample` [1342] functions. If user calls these functions too quickly they are blocked until new video frame or audio sample appears. If set to NFalse [1041] video frames or audio samples are not dropped and user can read them in any speed (gets all samples/frames). When set to NFalse [1041], NMediaReader [1336] works only with video/audio data saved in files.
NUInt flags	[in] Flags controlling NMediaReader [1336].
HNMediaReader * phReader	[out] Handle to created media reader.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

These flags can be used (for *flags* parameter):

- NMS_DO_NOT_USE_DIRECT_SHOW [1351]
- NMS_DO_NOT_USE_WINDOWS_MEDIA_FOUNDATION [1351]
- NMS_PREFER_DIRECT_SHOW [1351]

Note: these flags work only on Windows OS.

Notes

This function also creates NMediaSource [1345] from file (does the same as `NMediaSourceCreateFromFile` [1346]).

Module

NMediaReader Unit [1336]

7.1.5.12.1.3 NMediaReaderCreateFromUrl Function

Creates NMediaReader [1336] from URL.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NMediaReaderCreateFromUrl(const NChar * szUrl, NMediaType mediaTypes, NBool
isLive, NUInt flags, HNMediaReader * phReader);
```

Parameters

Parameters	Description
const NChar * szUrl	[in] Zero-terminated string which contains URL address (address of IP camera).
NMediaType mediaTypes	[in] Media types supported by NMediaReader [1336]. Can be *nmtAudio* and *nmtVideo* (only one at a time or both).
NBool isLive	[in] Flag indicating if media source is live. If set to NTrue [1064] media from source will be read in real time (from URL address). In this case video or audio samples will be dropped until user calls `NMediaReaderReadAudioSample` [1341] or `NMediaReaderReadVideoSample` [1342] functions. If user calls these functions too quickly they are blocked until new video frame or audio sample appears. If set to NFalse [1041] video frames or audio samples are not dropped and user can read them in any speed (gets all samples/frames). When set to NFalse [1041], NMediaReader [1336] works only with video/audio data saved in files. In this case setting NFalse [1041] will cause function error.
NUInt flags	[in] Flags controlling NMediaReader [1336].
HNMediaReader * phReader	[out] Handle to created media reader.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

These flags can be used (for *flags* parameter):

- NMS_DO_NOT_USE_DIRECT_SHOW [1351]
- NMS_DO_NOT_USE_WINDOWS_MEDIA_FOUNDATION [1351]
- NMS_PREFER_DIRECT_SHOW [1351]

Note: these flags work only on Windows OS.

Notes

This function does the same as `NMediaSourceCreateFromUrl` [1346].

Module

NMediaReader Unit [1336]

7.1.5.12.1.4 NMediaReaderGetLength Function

Gets file length (100 ns units) when NMediaReader [1336] source is file.

C++

```
NResult N_API NMediaReaderGetLength(HNMediaReader hReader, NTimeSpan_ * pValue);
```

Parameters

Parameters	Description
HNMediaReader hReader	[in] Handle to NMediaReader [1336].
NTimeSpan_ * pValue	[out] File length (100ns - nano second units).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

If media source is device, this function returns length equal to 0.

Module

NMediaReader Unit [1336]

7.1.5.12.1.5 NMediaReaderGetPosition Function

Gets NMediaReader [1336] position in media source (file).

C++

```
NResult N_API NMediaReaderGetPosition(HNMediaReader hReader, NTimeSpan_ * pValue);
```

Parameters

Parameters	Description
HNMediaReader hReader	[in] Handle to NMediaReader [1336].
NTimeSpan_ * pValue	[out] Current position of NMediaReader [1336]. Position is time in nano seconds (100ns units)

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaReader Unit [1336]

7.1.5.12.1.6 NMediaReaderGetSource Function

Gets media source (file or device) of the specified NMediaReader [1336].

C++

```
NResult N_API NMediaReaderGetSource(HNMediaReader hReader, HNMediaSource * phValue);
```

Parameters

Parameters	Description
HNMediaReader hReader	[in] Handle to NMediaReader [1336].
HNMediaSource * phValue	[out] Handle to NMediaSource [1345].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaReader Unit [1336]

7.1.5.12.1.7 NMediaReaderGetState Function

Gets NMediaReader [1336] state.

C++

```
NResult N_API NMediaReaderGetState(HNMediaReader hReader, NMediaState * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNMediaReader hReader	[in] Handle to NMediaReader [1336].
NMediaState * pValue	[out] NMediaReader [1336] state value (paused, running or stopped).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaReader Unit [1336]

7.1.5.12.1.8 NMediaReaderReadAudioSample Function

Reads audio sample using specified NMediaReader [1336].

C++

```
NResult N_API NMediaReaderReadAudioSample(HNMediaReader hReader, NTimeSpan_ * pTimeStamp,
NTimeSpan_ * pDuration, HNSoundBuffer * phSoundBuffer);
```

Parameters

Parameters	Description
HNMediaReader hReader	[in] Handle to NMediaReader [1336].
NTimeSpan_ * pTimeStamp	[out] Time stamp (from the beginning). 100ns units.
NTimeSpan_ * pDuration	[out] Audio sample duration (100ns units).
HNSoundBuffer * phSoundBuffer	[out] Handle to NSoundBuffer which contains audio sample.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

Before using this function user should start NMediaReader [1336] using `NMediaReaderStart [1343]`.

For more information on how to use this function see *ReadAudio* and *ReadAudioFromDevice* tutorials.

Module

NMediaReader Unit [1336]

7.1.5.12.1.9 NMediaReaderIsLive Function

Checks if media source which is used by specified NMediaReader [1336] is live.

C++

```
NResult N_API NMediaReaderIsLive(HNMediaReader hReader, NBool * pValue);
```

Parameters

Parameters	Description
HNMediaReader hReader	[in] Handle to NMediaReader [1336].
NBool * pValue	[out] Media source liveness check result. NTrue [1064] - media source is live (device). NFalse [1041] - media source is not live (file).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NMediaReader Unit [1336]

7.1.5.12.1.10 NMediaReaderReadVideoSample Function

Reads video sample using specified NMediaReader [1336].

C++

```
NResult N_API NMediaReaderReadVideoSample(HNMediaReader hReader, NTimeSpan_ * pTimeStamp,
NTimeSpan_ * pDuration, HNImage * phImage);
```

Parameters

Parameters	Description
HNMediaReader hReader	[in] Handle to NMediaReader [1336].
NTimeSpan_ * pTimeStamp	[out] Time stamp (from the beginning). 100ns units.
NTimeSpan_ * pDuration	[out] Video sample duration (100ns units).
HNImage * phImage	[out] Handle to NImage [1232] which contains video sample.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

Before using this function user should start NMediaReader [1336] using `NMediaReaderStart [1343]`.

For more information on how to use this function see *ReadVideo* and *ReadVideoFromDevice* tutorials.

Module

NMediaReader Unit [1336]

7.1.5.12.1.11 NMediaReaderPause Function

Pauses specified NMediaReader [1336].

C++

```
NResult N_API NMediaReaderPause(HNMediaReader hReader);
```

Parameters

Parameters	Description
HNMediaReader hReader	[in] Handle to NMediaReader [1336].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaReader Unit [1336]

7.1.5.12.1.12 NMediaReaderSetPosition Function

Sets NMediaReader [1336] position in media source (file).

C++

```
NResult N_API NMediaReaderSetPosition(HNMediaReader hReader, NTimeSpan_ value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNMediaReader hReader	[in] Handle to NMediaReader [1336].
NTimeSpan_ value	[in] Position of NMediaReader [1336] to set. Position is time in nano seconds (100ns units)

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaReader Unit [1336]

7.1.5.12.1.13 NMediaReaderStart Function

Starts NMediaReader [1336].

C++

```
NResult N_API NMediaReaderStart(HNMediaReader hReader);
```

Parameters

Parameters	Description
HNMediaReader hReader	[in] Handle to NMediaReader [1336] to start.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaReader Unit [1336]

7.1.5.12.1.14 NMediaReaderStop Function

Stops NMediaReader [1336].

C++

```
NResult N_API NMediaReaderStop(HNMediaReader hReader);
```

Parameters

Parameters	Description
HNMediaReader hReader	[in] Handle to NMediaReader [1336] to stop.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaReader Unit [1336]

7.1.5.12.1.15 HNMediaReader

Handle to NMediaReader [1336]

Module

NMediaReader Unit [1336]

7.1.5.12.2 Macros

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.12.2.1 NMediaReaderCreateFromFile Macro

Picks a Unicode (W) or ANSCII (A) version of the NMediaReaderFromFile function.

C++

```
#define NMediaReaderCreateFromFile N_FUNC_AW(NMediaReaderCreateFromFile)
```

Module

NMediaReader Unit [1336]

7.1.5.12.2.2 NMediaReaderCreateFromUrl Macro

Picks a Unicode (W) or ANSCII (A) version of the NMediaReaderFromUrl function.

C++

```
#define NMediaReaderCreateFromUrl N_FUNC_AW(NMediaReaderCreateFromUrl)
```

Module

NMediaReader Unit [1336]

7.1.5.12.3 Files

7.1.5.12.3.1 NMediaReader.h

Header file for NMediaReader [1336] Unit.

Functions

	Name	Description
	NMediaReaderCreate [1337]	Creates media reader object.
	NMediaReaderCreateFromFile [1338]	Creates NMediaReader [1336] from file.
	NMediaReaderCreateFromUrl [1338]	Creates NMediaReader [1336] from URL.
	NMediaReaderGetLength [1339]	Gets file length (100 ns units) when NMediaReader [1336] source is file.
	NMediaReaderGetPosition [1340]	Gets NMediaReader [1336] position in media source (file).
	NMediaReaderGetSource [1340]	Gets media source (file or device) of the specified NMediaReader [1336].
	NMediaReaderGetState [1340]	Gets NMediaReader [1336] state.
	NMediaReaderIsLive [1341]	Checks if media source which is used by specified NMediaReader [1336] is live.
	NMediaReaderPause [1342]	Pauses specified NMediaReader [1336].
	NMediaReaderReadAudioSample [1341]	Reads audio sample using specified NMediaReader [1336].
	NMediaReaderReadVideoSample [1342]	Reads video sample using specified NMediaReader [1336].
	NMediaReaderSetPosition [1342]	Sets NMediaReader [1336] position in media source (file).
	NMediaReaderStart [1343]	Starts NMediaReader [1336].
	NMediaReaderStop [1343]	Stops NMediaReader [1336].

Macros

Name	Description
NMediaReaderCreateFromFile [1344]	Picks a Unicode (W) or ANSCII (A) version of the NMediaReaderFromFile function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMediaReaderCreateFromUrl [1344]	Picks a Unicode (W) or ANSCII (A) version of the NMediaReaderFromUrl function.

Module

NMediaReader Unit [1336]

7.1.5.13 NMediaSource Unit

Provides functions for working with media source.

Files

Name	Description
NMediaSource.h [1351]	Header file for NMediaSource Unit.

Functions

	Name	Description
	NMediaSourceCreateFromFile [1346]	Creates NMediaSource from specified file.
	NMediaSourceCreateFromUrl [1346]	Creates media source from URL.
	NMediaSourceEnumDevices [1347]	Enumerates all source devices by specified media type.
	NMediaSourceGetCurrentFormat [1347]	Gets media source format for specified media type.
	NMediaSourceGetDisplayName [1348]	Gets display name of specified media source.
	NMediaSourceGetFormats [1348]	Gets media source formats.
	NMediaSourceGetId [1349]	Gets media source Id.
	NMediaSourceGetMediaType [1349]	Gets media type of specified media source.
	NMediaSourceSetCurrentFormat [1349]	Sets media source format for specified media type.
	HNmediaSource [1350]	Handle to NMediaSource unit.

Macros

Name	Description
NMediaSourceCreateFromFile [1350]	Picks a Unicode (W) or ANSCII (A) version of the NMediaSourceCreateFromFile function.
NMediaSourceCreateFromUrl [1350]	Picks a Unicode (W) or ANSCII (A) version of the NMediaSourceCreateFromUrl function.
NMediaSourceGetDisplayName [1350]	Picks a Unicode (W) or ANSCII (A) version of the NMediaSourceGetDisplayName function.
NMediaSourceGetId [1350]	Picks a Unicode (W) or ANSCII (A) version of the NMediaSourceGetId function.
NMS_ALLOW_DUPLICATE_DEVICES [1351]	This flag is used to allow enumerating duplicate devices. If this flag was set and the same device is accessible through DirectShow and Windows Media Foundation (Windows 7) then this device will be enumerated twice.
NMS_DO_NOT_USE_DIRECT_SHOW [1351]	Used when user prefers not to use DirecShow for NMediaSource.
NMS_DO_NOT_USE_WINDOWS_MEDIA_FOUNDATION [1351]	Used when user prefers not to use Windows Media Foundation for NMediaSource.
NMS_PREFER_DIRECT_SHOW [1351]	Used when user prefers to use DirectShow for NMediaSource.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.13.1 Functions

7.1.5.13.1.1 NMediaSourceCreateFromFile Function

Creates NMediaSource [1345] from specified file.

C++

```
NResult N_API NMediaSourceCreateFromFile(const NChar * szFileName, NUInt flags,
HNMediaSource * phSource);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Zero terminated string which contains file name.
NUInt flags	[in] Flags controlling NMediaSource [1345].
HNMediaSource * phSource	[out] Handle to created media source.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

Created NMediaSource [1345] is used by NMediaReader [1336] to read data.

These flags can be used (for *flags* parameter):

- NMS_DO_NOT_USE_DIRECT_SHOW [1351]
- NMS_DO_NOT_USE_WINDOWS_MEDIA_FOUNDATION [1351]
- NMS_PREFER_DIRECT_SHOW [1351]

Note: these flags work only on Windows OS.

Notes

User also can call `NMediaReaderCreateFromFile` [1338] function which creates NMediaSource [1345] and NMediaReader [1336] at the same time.

Module

NMediaSource Unit [1345]

7.1.5.13.1.2 NMediaSourceCreateFromUrl Function

Creates media source from URL.

C++

```
NResult N_API NMediaSourceCreateFromUrl(const NChar * szUrl, NUInt flags, HNMediaSource *
phSource);
```

Parameters

Parameters	Description
const NChar * szUrl	[in] Zero-terminated string which contains URL address (address of IP camera).
NUInt flags	[in] Flags controlling NMediaSource [1345].
HNMediaSource * phSource	[out] Handle to created media source.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

These flags can be used (for *flags* parameter):

- NMS_DO_NOT_USE_DIRECT_SHOW [1351]
- NMS_DO_NOT_USE_WINDOWS_MEDIA_FOUNDATION [1351]
- NMS_PREFER_DIRECT_SHOW [1351]

Note: these flags work only on Windows OS.

Notes

User also can call `NMediaReaderCreateFromUrl` [1338] function which creates NMediaSource [1345] and NMediaReader [1336] at the same time.

Module

NMediaSource Unit [1345]

7.1.5.13.1.3 NMediaSourceEnumDevices Function

Enumerates all source devices by specified media type.

C++

```
NResult N_API NMediaSourceEnumDevices(NMediaType mediaType, NUInt flags, HNMediaSource * *
parhDevices, NInt * pDeviceCount);
```

Parameters

Parameters	Description
NMediaType mediaType	[in] Device media type.
NUInt flags	[in] Flags controlling NMediaSource [1345].
HNMediaSource * * parhDevices	[out] Pointer to array of handles to media sources.
NInt * pDeviceCount	[out] Devices count.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

User should free all resources (array and handles) used by this function. `NObjectFreeArray` [543] function is used for this purpose.

Module

NMediaSource Unit [1345]

7.1.5.13.1.4 NMediaSourceGetCurrentFormat Function

Gets media source format for specified media type.

C++

```
NResult N_API NMediaSourceGetCurrentFormat(HNMediaSource hSource, NMediaType mediaType,
HNMediaFormat * phFormat);
```

Parameters

Parameters	Description
HNMediaSource hSource	[in] Handle to NMediaSource [1345].
NMediaType mediaType	[in] Media type to retrieve its format.
HNMediaFormat * phFormat	[out] Handle to NMediaFormat [1334] which defines format of current media source.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

`NMediaSourceGetMediaType [1349]` function can be used to determine NMediaSource [1345] type. This function can be called only when media source is inside NMediaReader [1336].

Module

NMediaSource Unit [1345]

7.1.5.13.1.5 NMediaSourceGetDisplayName Function

Gets display name of specified media source.

C++

```
NResult N_API NMediaSourceGetDisplayName(HNMediaSource hSource, NChar * szValue, NInt
valueSize);
```

Parameters

Parameters	Description
HNMediaSource hSource	[in] Handle to NMediaSource [1345] to retrieve display name of it source (file or live stream).
NChar * szValue	[out] Zero terminated string containing media source display name.
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

Returned string can be used to display media source name on a screen.

Module

NMediaSource Unit [1345]

7.1.5.13.1.6 NMediaSourceGetFormats Function

Gets media source formats.

C++

```
NResult N_API NMediaSourceGetFormats(HNMediaSource hSource, NMediaType mediaType,
HNMediaFormat * * parhFormats, NInt * pFormatCount);
```

Parameters

Parameters	Description
HNMediaSource hSource	[in] Handle to NMediaSource [1345].
NMediaType mediaType	[in] Media type to retrieve its formats.
HNMediaFormat * * parhFormats	[out] Pointer to array which contains handles to NMediaFormat [1334].
NInt * pFormatCount	[out] Array items count.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NMediaSource Unit [1345]

7.1.5.13.1.7 NMediaSourceGetId Function

Gets media source Id.

C++

```
NResult N_API NMediaSourceGetId(HNMediaSource hSource, NChar * szValue, NInt valueSize);
```

Parameters

Parameters	Description
HNMediaSource hSource	[in] Handle to NMediaSource [1345].
NChar * szValue	[out] Zero terminated string which contains media source Id.
NInt valueSize	[in] Size of memory buffer containing zero terminated string.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaSource Unit [1345]

7.1.5.13.1.8 NMediaSourceGetMediaType Function

Gets media type of specified media source.

C++

```
NResult N_API NMediaSourceGetMediaType(HNMediaSource hSource, NMediaType * pValue);
```

Parameters

Parameters	Description
HNMediaSource hSource	[in] Handle to NMediaSource [1345].
NMediaType * pValue	[out] Pointer to NMediaType [1354] which contains the type of current media source.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaSource Unit [1345]

7.1.5.13.1.9 NMediaSourceSetCurrentFormat Function

Sets media source format for specified media type.

C++

```
NResult N_API NMediaSourceSetCurrentFormat(HNMediaSource hSource, NMediaType mediaType,
HNMediaFormat hFormat);
```

Parameters

Parameters	Description
HNMediaSource hSource	[in] Handle to NMediaSource [1345].
NMediaType mediaType	[in] Media type for which format should be set.
HNMediaFormat hFormat	[in] Handle to media format to set.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaSource Unit [1345]

7.1.5.13.1.10 HNmediaSource

Handle to NMediaSource [1345] unit.

Module

NMediaSource Unit [1345]

7.1.5.13.2 Macros

7.1.5.13.2.1 NMediaSourceCreateFromFile Macro

Picks a Unicode (W) or ANSCII (A) version of the NMediaSourceCreateFromFile function.

C++

```
#define NMediaSourceCreateFromFile N_FUNC_AW(NMediaSourceCreateFromFile)
```

Module

NMediaSource Unit [1345]

7.1.5.13.2.2 NMediaSourceCreateFromUrl Macro

Picks a Unicode (W) or ANSCII (A) version of the NMediaSourceCreateFromUrl function.

C++

```
#define NMediaSourceCreateFromUrl N_FUNC_AW(NMediaSourceCreateFromUrl)
```

Module

NMediaSource Unit [1345]

7.1.5.13.2.3 NMediaSourceGetDisplayName Macro

Picks a Unicode (W) or ANSCII (A) version of the NMediaSourceGetDisplayName function.

C++

```
#define NMediaSourceGetDisplayName N_FUNC_AW(NMediaSourceGetDisplayName)
```

Module

NMediaSource Unit [1345]

7.1.5.13.2.4 NMediaSourceGetId Macro

Picks a Unicode (W) or ANSCII (A) version of the NMediaSourceGetId function.

C++

```
#define NMediaSourceGetId N_FUNC_AW(NMediaSourceGetId)
```

Module

NMediaSource Unit [1345]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.13.2.5 NMS_ALLOW_DUPLICATE_DEVICES Macro

This flag is used to allow enumerating duplicate devices. If this flag was set and the same device is accessible through DirectShow and Windows Media Foundation (Windows 7) then this device will be enumerated twice.

C++

```
#define NMS_ALLOW_DUPLICATE_DEVICES 0x00000008
```

Module

NMediaSource Unit [1345]

7.1.5.13.2.6 NMS_DO_NOT_USE_DIRECT_SHOW Macro

Used when user prefers not to use DirecShow for NMediaSource [1345].

C++

```
#define NMS_DO_NOT_USE_DIRECT_SHOW 0x00000001
```

Module

NMediaSource Unit [1345]

7.1.5.13.2.7 NMS_DO_NOT_USE_WINDOWS_MEDIA_FOUNDATION Macro

Used when user prefers not to use Windows Media Foundation for NMediaSource [1345].

C++

```
#define NMS_DO_NOT_USE_WINDOWS_MEDIA_FOUNDATION 0x00000002
```

Module

NMediaSource Unit [1345]

7.1.5.13.2.8 NMS_PREFER_DIRECT_SHOW Macro

Used when user prefers to use DirectShow for NMediaSource [1345].

C++

```
#define NMS_PREFER_DIRECT_SHOW 0x00000004
```

Module

NMediaSource Unit [1345]

7.1.5.13.3 Files

7.1.5.13.3.1 NMediaSource.h

Header file for NMediaSource [1345] Unit.

Functions

	Name	Description
	NMediaSourceCreateFromFile [1346]	Creates NMediaSource [1345] from specified file.
	NMediaSourceCreateFromUrl [1346]	Creates media source from URL.
	NMediaSourceEnumDevices [1347]	Enumerates all source devices by specified media type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NMediaSourceGetCurrentFormat [1347]	Gets media source format for specified media type.
	NMediaSourceGetDisplayName [1348]	Gets display name of specified media source.
	NMediaSourceGetFormats [1348]	Gets media source formats.
	NMediaSourceGetId [1349]	Gets media source Id.
	NMediaSourceGetMediaType [1349]	Gets media type of specified media source.
	NMediaSourceSetCurrentFormat [1349]	Sets media source format for specified media type.

Macros

Name	Description
NMediaSourceCreateFromFile [1350]	Picks a Unicode (W) or ANSCII (A) version of the NMediaSourceCreateFromFile function.
NMediaSourceCreateFromUrl [1350]	Picks a Unicode (W) or ANSCII (A) version of the NMediaSourceCreateFromUrl function.
NMediaSourceGetDisplayName [1350]	Picks a Unicode (W) or ANSCII (A) version of the NMediaSourceGetDisplayName function.
NMediaSourceGetId [1350]	Picks a Unicode (W) or ANSCII (A) version of the NMediaSourceGetId function.
NMS_ALLOW_DUPLICATE_DEVICES [1351]	This flag is used to allow enumerating duplicate devices. If this flag was set and the same device is accessible through DirectShow and Windows Media Foundation (Windows 7) then this device will be enumerated twice.
NMS_DO_NOT_USE_DIRECT_SHOW [1351]	Used when user prefers not to use DirecShow for NMediaSource [1345].
NMS_DO_NOT_USE_WINDOWS_MEDIA_FOUNDATION [1351]	Used when user prefers not to use Windows Media Foundation for NMediaSource [1345].
NMS_PREFER_DIRECT_SHOW [1351]	Used when user prefers to use DirectShow for NMediaSource [1345].

Module

NMediaSource Unit [1345]

7.1.5.14 NMediaTypes Unit

Enumerates media types and provides functions to check if media type is valid.

Files

Name	Description
NMediaTypes.h [1355]	Header file for NMediaTypes unit.

Functions

	Name	Description
	NMediaTypeIsValid [1353]	Checks if the value passed to a function is a valid media type.
	NMediaTypeIsValidSingle [1353]	Checks if the single value passed to a function is a valid media type.
	NVideoInterlaceModeIsValid [1353]	Checks if video interlace mode is valid.

Structs, Records, Enums

	Name	Description
	NMediaState_ [1354]	Enumerates media state values.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NMediaType_ [1354]	Enumerates media type values.
	NVideoInterlaceMode_ [1354]	Enumerates video scan mode values.
	NMediaState [1354]	Enumerates media state values.
	NMediaType [1354]	Enumerates media type values.
	NVideoInterlaceMode [1354]	Enumerates video scan mode values.

7.1.5.14.1 Functions

7.1.5.14.1.1 NMediaTypeIsValid Function

Checks if the value passed to a function is a valid media type.

C++

```
NBool N_API NMediaTypeIsValid(NMediaType value);
```

Parameters

Parameters	Description
NMediaType value	[in] On of NMediaTypes [1352] values to check.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaTypes Unit [1352]

7.1.5.14.1.2 NMediaTypeIsValidSingle Function

Checks if the single value passed to a function is a valid media type.

C++

```
NBool N_API NMediaTypeIsValidSingle(NMediaType value);
```

Parameters

Parameters	Description
NMediaType value	[in] On of NMediaTypes [1352] values to check.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaTypes Unit [1352]

7.1.5.14.1.3 NVideoInterlaceModeIsValid Function

Checks if video interlace mode is valid.

C++

```
NBool N_API NVideoInterlaceModeIsValid(NVideoInterlaceMode value);
```

Parameters

Parameters	Description
NVideoInterlaceMode value	in] On of NVideoInterlaceMode [1354] values to check.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMediaTypes Unit [1352]

7.1.5.14.2 Structs, Records, Enums

7.1.5.14.2.1 NMediaState Enumeration

Enumerates media state values.

C++

```
typedef enum NMediaState_ {
  nmsStopped = 0,
  nmsPaused = 1,
  nmsRunning = 2
} NMediaState;
```

Members

Members	Description
nmsStopped = 0	Indicates stopped media stream.
nmsPaused = 1	Indicates paused media stream.
nmsRunning = 2	Indicates running media stream.

Module

NMediaTypes Unit [1352]

7.1.5.14.2.2 NMediaType Enumeration

Enumerates media type values.

C++

```
typedef enum NMediaType_ {
  nmtNone = 0,
  nmtAudio = 1,
  nmtVideo = 2
} NMediaType;
```

Members

Members	Description
nmtNone = 0	Media type was not recognized.
nmtAudio = 1	Indicates that media type is audio.
nmtVideo = 2	Indicates that media type is video.

Module

NMediaTypes Unit [1352]

7.1.5.14.2.3 NVideoInterlaceMode Enumeration

Enumerates video scan mode values.

C++

```
typedef enum NVideoInterlaceMode_ {
  nvimUnknown = 0,
  nvimInterlaced = 1,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
  nvimProgressive = 2
} NVideoInterlaceMode;
```

Members

Members	Description
nvimUnknown = 0	Indicates that video modes is unknown.
nvimInterlaced = 1	Indicates that video is displayed in interlaced scan.
nvimProgressive = 2	Indicates that video is displayed in progressive scan.

Module

NMediaTypes Unit [1352]

7.1.5.14.3 Files

7.1.5.14.3.1 NMediaTypes.h

Header file for NMediaTypes [1352] unit.

Enumerations

	Name	Description
	NMediaState_ [1354]	Enumerates media state values.
	NMediaType_ [1354]	Enumerates media type values.
	NVideoInterlaceMode_ [1354]	Enumerates video scan mode values.
	NMediaState [1354]	Enumerates media state values.
	NMediaType [1354]	Enumerates media type values.
	NVideoInterlaceMode [1354]	Enumerates video scan mode values.

Functions

	Name	Description
	NMediaTypeIsValid [1353]	Checks if the value passed to a function is a valid media type.
	NMediaTypeIsValidSingle [1353]	Checks if the single value passed to a function is a valid media type.
	NVideoInterlaceModeIsValid [1353]	Checks if video interlace mode is valid.

Module

NMediaTypes Unit [1352]

7.1.5.15 NMonochromeImage Unit

Creates color wrapper for grayscale image.

Files

Name	Description
NMonochromeImage.h [1358]	Header file for NMonochromeImage module. Provides functionality for managing 1-bit monochrome images.

Functions

	Name	Description
	NMonochromeImageGetMaxValue [1356]	Retrieves maximum pixel value for the specified monochrome image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NMonochromeImageGetMinValue [1356]	Retrieves minimum pixel value for the specified monochrome image.
	NMonochromeImageGetPixel [1356]	Retrieves value of pixel at the specified coordinates in 1-bit monochrome image.
	NMonochromeImageSetMaxValue [1357]	Sets the maximum pixel value for the specified monochrome image.
	NMonochromeImageSetMinValue [1357]	Sets the minimum pixel value for the specified monochrome image.
	NMonochromeImageSetPixel [1358]	Sets value of pixel at the specified coordinates in 1-bit monochrome image.

7.1.5.15.1 Functions

7.1.5.15.1.1 NMonochromeImageGetMaxValue Function

Retrieves maximum pixel value for the specified monochrome image.

C++

```
NResult N_API NMonochromeImageGetMaxValue(HNMonochromeImage hImage, NBool * pValue);
```

Parameters

Parameters	Description
HNMonochromeImage hImage	[in] Handle to the image.
NBool * pValue	[out] Pointer to the maximum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMonochromeImage Unit [1355]

7.1.5.15.1.2 NMonochromeImageGetMinValue Function

Retrieves minimum pixel value for the specified monochrome image.

C++

```
NResult N_API NMonochromeImageGetMinValue(HNMonochromeImage hImage, NBool * pValue);
```

Parameters

Parameters	Description
HNMonochromeImage hImage	[in] Handle to the image.
NBool * pValue	[out] Pointer to the minimum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMonochromeImage Unit [1355]

7.1.5.15.1.3 NMonochromeImageGetPixel Function

Retrieves value of pixel at the specified coordinates in 1-bit monochrome image.

C++

```
NResult N_API NMonochromeImageGetPixel(HNMonochromeImage hImage, NUInt x, NUInt y, NBool *
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
pValue);
```

Parameters

Parameters	Description
HNMonochromeImage hImage	[in] Handle to image.
NUInt x	[in] Specifies x-coordinate of the pixel.
NUInt y	[in] Specifies y-coordinate of the pixel.
NBool * pValue	[out] Points to NBool [1004] that receives pixel value.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *pValue [870]* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	This error code is returned when x is greater than or equal to image width. - or - y is greater than or equal to image height.
N_E_FORMAT [429]	This error code is returned when image pixel format is not equal to *npfMonochrome [1384]*.

Remarks

If pixel has black color then a value that *pValue [870]* points to receives NFalse [1041] and if it is white then value receives NTrue [1064].

Module

NMonochromeImage Unit [1355]

7.1.5.15.1.4 NMonochromeImageSetMaxValue Function

Sets the maximum pixel value for the specified monochrome image.

C++

```
NResult N_API NMonochromeImageSetMaxValue(HNMonochromeImage hImage, NBool value);
```

Parameters

Parameters	Description
HNMonochromeImage hImage	[in] Handle to the image.
NBool value	[in] Maximum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMonochromeImage Unit [1355]

7.1.5.15.1.5 NMonochromeImageSetMinValue Function

Sets the minimum pixel value for the specified monochrome image.

C++

```
NResult N_API NMonochromeImageSetMinValue(HNMonochromeImage hImage, NBool value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNMonochromeImage hImage	[in] Handle to the image.
NBool value	[in] Minimum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NMonochromeImage Unit [1355]

7.1.5.15.1.6 NMonochromeImageSetPixel Function

Sets value of pixel at the specified coordinates in 1-bit monochrome image.

C++

```
NResult N_API NMonochromeImageSetPixel(HNMonochromeImage hImage, NUInt x, NUInt y, NBool
value);
```

Parameters

Parameters	Description
HNMonochromeImage hImage	[in] Handle to image.
NUInt x	[in] Specifies x-coordinate of the pixel.
NUInt y	[in] Specifies y-coordinate of the pixel.
NBool value	[in] Specifies new pixel value.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	This error code is returned when *hImage* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	This error code is returned when x is greater than or equal to image width. - or - y is greater than or equal to image height.
N_E_FORMAT [429]	This error code is returned when image pixel format is not equal to *npfMonochrome [1384]*.

Remarks

If value is NFalse [1041] then pixel will be black and if it is NTrue [1064] then pixel will be white.

Module

NMonochromeImage Unit [1355]

7.1.5.15.2 Files

7.1.5.15.2.1 NMonochromeImage.h

Header file for NMonochromeImage [1355] module. Provides functionality for managing 1-bit monochrome images.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NMonochromeImageGetMaxValue [1356]	Retrieves maximum pixel value for the specified monochrome image.
	NMonochromeImageGetMinValue [1356]	Retrieves minimum pixel value for the specified monochrome image.
	NMonochromeImageGetPixel [1356]	Retrieves value of pixel at the specified coordinates in 1-bit monochrome image.
	NMonochromeImageSetMaxValue [1357]	Sets the maximum pixel value for the specified monochrome image.
	NMonochromeImageSetMinValue [1357]	Sets the minimum pixel value for the specified monochrome image.
	NMonochromeImageSetPixel [1358]	Sets value of pixel at the specified coordinates in 1-bit monochrome image.

Module

NMonochromeImage Unit [1355]

7.1.5.16 NPixelFormat Unit

Provides functionality for working with image pixel format.

Files

Name	Description
NPixelFormat.h [1385]	Header file for NPixelFormat [1369] module. Provides functionality for working with image pixel format.

Functions

	Name	Description
	NPixelFormatCalcRowSize [1363]	Calculates number of bytes needed to store line of specified length of pixels with specified NPixelFormat [1369].
	NPixelFormatCreate [1364]	Creates a new NPixelFormat [1369].
	NPixelFormatGetBitsPerPixel [1364]	Retrieves number of bits used to store a pixel from NPixelFormat [1369].
	NPixelFormatGetBytesPerPixel [1365]	Retrieves number of bytes used to store a pixel from NPixelFormat [1369].
	NPixelFormatGetIndexed [1365]	Retrieves an indexed version of the NPixelFormat [1369].
	NPixelFormatGetNonIndexed [1365]	Retrieves a non-indexed version of the NPixelFormat [1369].
	NPixelFormatGetNonSeparated [1366]	Retrieves a non-separated version of the NPixelFormat [1369].
	NPixelFormatGetPaletteSize [1366]	Retrieves the palette size for the pixel format.
	NPixelFormatGetPixelType [1366]	Retrieves the pixel type for the specified pixel format.
	NPixelFormatGetRowSize [1367]	Calculates number of bytes needed to store line of specified length of pixels with specified NPixelFormat [1369].
	NPixelFormatGetSeparated [1367]	Retrieves a separated version of the NPixelFormat [1369].
	NPixelFormatGetWithExtraChannel [1367]	Retrieves a version of the NPixelFormat [1369] with an extra channel.
	NPixelFormatIsValid [1368]	Checks if specified pixel format is valid.
	NPixelFormatToString [1368]	Converts a pixel format into a string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NPixelTypeCanBeIndexed [1368]	Checks if specified pixel type can be indexed.
	NPixelTypeIsValid [1369]	Checks if specified pixel type is valid.

Macros

Name	Description
NPF_GRAYSCALE_16S [1370]	Specifies grayscale image pixel format where each pixel is represented by 16 bits per channel (channel format - signed integer).
NPF_GRAYSCALE_16U [1370]	Specifies grayscale image pixel format where each pixel is represented by 16 bits per channel (channel format - unsigned integer).
NPF_GRAYSCALE_1U [1370]	Specifies grayscale image pixel format where each pixel is represented by 1 bit per channel (channel format - unsigned integer).
NPF_GRAYSCALE_2U [1370]	Specifies grayscale image pixel format where each pixel is represented by 2 bits per channel (channel format - unsigned integer).
NPF_GRAYSCALE_32F [1371]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per channel (channel format - floating point).
NPF_GRAYSCALE_32S [1371]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per channel (channel format - signed integer).
NPF_GRAYSCALE_32U [1371]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per channel (channel format - unsigned integer).
NPF_GRAYSCALE_4U [1371]	Specifies grayscale image pixel format where each pixel is represented by 4 bits per channel (channel format - unsigned integer).
NPF_GRAYSCALE_64F [1371]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per channel (channel format - floating point).
NPF_GRAYSCALE_64S [1372]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per channel (channel format - signed integer).
NPF_GRAYSCALE_64U [1372]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per channel (channel format - unsigned integer).
NPF_GRAYSCALE_8S [1372]	Specifies grayscale image pixel format where each pixel is represented by 8 bits per channel (channel format - signed integer).
NPF_GRAYSCALE_8U [1372]	Specifies grayscale image pixel format where each pixel is represented by 8 bits per channel (channel format - unsigned integer).
NPF_GRAYSCALE_A_16S [1372]	Specifies grayscale image pixel format where each pixel is represented by 16 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_16U [1373]	Specifies grayscale image pixel format where each pixel is represented by 16 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_32F [1373]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per alpha channel (channel format - floating point). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_32S [1373]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_32U [1373]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_64F [1373]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per alpha channel (channel format - floating point). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_64S [1374]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_64U [1374]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPF_GRAYSCALE_A_8S [1374]	Specifies grayscale image pixel format where each pixel is represented by 8 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_8U [1374]	Specifies grayscale image pixel format where each pixel is represented by 8 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_PA_16S [1374]	Specifies grayscale image pixel format where each pixel is represented by 16 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_GRAYSCALE_PA_16U [1375]	Specifies grayscale image pixel format where each pixel is represented by 16 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_GRAYSCALE_PA_32F [1375]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per pre-multiplied alpha channel (channel format - floating point).
NPF_GRAYSCALE_PA_32S [1375]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_GRAYSCALE_PA_32U [1375]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_GRAYSCALE_PA_64F [1375]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per pre-multiplied alpha channel (channel format - floating point).
NPF_GRAYSCALE_PA_64S [1376]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_GRAYSCALE_PA_64U [1376]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_GRAYSCALE_PA_8S [1376]	Specifies grayscale image pixel format where each pixel is represented by 8 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_GRAYSCALE_PA_8U [1376]	Specifies grayscale image pixel format where each pixel is represented by 8 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_RGB_16S [1376]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per channel (channel format - signed integer).
NPF_RGB_16U [1377]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per channel (channel format - unsigned integer).
NPF_RGB_32F [1377]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per channel (channel format - floating point).
NPF_RGB_32S [1377]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per channel (channel format - signed integer).
NPF_RGB_32U [1377]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per channel (channel format - unsigned integer).
NPF_RGB_64F [1377]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per channel (channel format - floating point).
NPF_RGB_64S [1378]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per channel (channel format - signed integer).
NPF_RGB_64U [1378]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per channel (channel format - unsigned integer).
NPF_RGB_8S [1378]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - signed integer).
NPF_RGB_8U [1378]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPF_RGB_8U_INDEXED_1 [1378]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer). Also 1 bit is for saving index to image palette.
NPF_RGB_8U_INDEXED_2 [1379]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer). Also 2 bits is for saving index to image palette.
NPF_RGB_8U_INDEXED_4 [1379]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer). Also 4 bits is for saving index to image palette.
NPF_RGB_8U_INDEXED_8 [1379]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer). Also 8 bits is for saving index to image palette.
NPF_RGB_A_16S [1379]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_16U [1379]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_32F [1380]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per alpha channel (channel format - floating point). Alpha channel specifies pixel transparency.
NPF_RGB_A_32S [1380]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_32U [1380]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_64F [1380]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per alpha channel (channel format - floating point). Alpha channel specifies pixel transparency.
NPF_RGB_A_64S [1380]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_64U [1381]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_8S [1381]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_8U [1381]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_8U_INDEXED_1 [1381]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Also 1 bit is for saving index to image palette.
NPF_RGB_A_8U_INDEXED_2 [1381]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Also 2 bits is for saving index to image palette.
NPF_RGB_A_8U_INDEXED_4 [1382]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Also 4 bits is for saving index to image palette.
NPF_RGB_A_8U_INDEXED_8 [1382]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Also 8 bits is for saving index to image palette.
NPF_RGB_PA_16S [1382]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per pre-multiplied alpha channel (channel format - signed integer).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPF_RGB_PA_16U [1382]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_RGB_PA_32F [1382]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per pre-multiplied alpha channel (channel format - floating point).
NPF_RGB_PA_32S [1383]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_RGB_PA_32U [1383]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_RGB_PA_64F [1383]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per pre-multiplied alpha channel (channel format - floating point).
NPF_RGB_PA_64S [1383]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_RGB_PA_64U [1383]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_RGB_PA_8S [1384]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_RGB_PA_8U [1384]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_UNDEFINED [1384]	Indicates undefined pixel format.
npfGrayscale [1384]	Indicates a grayscale pixel.
npfMonochrome [1384]	Indicates a monochrome pixel.
npfRgb [1385]	Indiactes an RGB pixel.
NPixelFormatToString [1385]	Picks a Unicode (W) or ANSCII (A) version of the NPixelFormatToString function.

Structs, Records, Enums

	Name	Description
	NPixelType_ [1369]	Specifies pixel type of each pixel in the image.
	NPixelType [1369]	Specifies pixel type of each pixel in the image.

Types

Name	Description
NPixelFormat [1369]	Specifies pixel format of each pixel in the image.
NPixelFormat_ [1370]	Specifies pixel format of each pixel in the image.

7.1.5.16.1 Functions

7.1.5.16.1.1 NPixelFormatCalcRowSize Function

Calculates number of bytes needed to store line of specified length of pixels with specified NPixelFormat [1369].

C++

```
NResult N_API NPixelFormatCalcRowSize(NInt bitsPerPixel, NUInt length, NSizeType alignment,
NSizeType * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NInt bitsPerPixel	[in] A number of bits per pixel.
NUInt length	[in] Length of pixels.
NSizeType alignment	[in] Alignment.
NSizeType * pValue	[out] Pointer to NSizeType [1008] that receives the number of bytes needed to store line of specified length of pixels.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.2 NPixelFormatCreate Function

Creates a new NPixelFormat [1369].

C++

```
NResult N_API NPixelFormatCreate(NPixelType pixelType, NExtraChannel extraChannel, NInt
channelCount, NChannelFormat channelFormat, NInt bitsPerChannel, NInt bitsPerIndex, NBool
isSeparated, NPixelFormat_ * pValue);
```

Parameters

Parameters	Description
NPixelType pixelType	[in] The specified type of the pixels.
NExtraChannel extraChannel	[in] The extra channel.
NInt channelCount	[in] The number of channels.
NChannelFormat channelFormat	[in] The format of the channel.
NInt bitsPerChannel	[in] The number of bits per chanel
NInt bitsPerIndex	[in] The number of bits per index.
NBool isSeparated	[in] NTrue [1064] if seprated; else NFalse [1041].
NPixelFormat_ * pValue	[out] A pointer to the created NPixelFormat [1369].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.3 NPixelFormatGetBitsPerPixel Function

Retrieves number of bits used to store a pixel from NPixelFormat [1369].

C++

```
NInt N_API NPixelFormatGetBitsPerPixel(NPixelFormat_ pixelFormat);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.

Returns

Number of bits used to store a pixel from NPixelFormat [1369].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.4 NPixelFormatGetBytesPerPixel Function

Retrieves number of bytes used to store a pixel from NPixelFormat [1369].

C++

```
NSizeType N_API NPixelFormatGetBytesPerPixel(NPixelFormat_ pixelFormat);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.5 NPixelFormatGetIndexed Function

Retrieves an indexed version of the NPixelFormat [1369].

C++

```
NResult N_API NPixelFormatGetIndexed(NPixelFormat_ pixelFormat, NInt bitsPerIndex,
NPixelFormat_ * pValue);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NInt bitsPerIndex	[in] The number of bits per index.
NPixelFormat_ * pValue	[out] A pointer to the

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.6 NPixelFormatGetNonIndexed Function

Retrieves a non-indexed version of the NPixelFormat [1369].

C++

```
NResult N_API NPixelFormatGetNonIndexed(NPixelFormat_ pixelFormat, NPixelFormat_ * pValue);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NPixelFormat_ * pValue	[out] A pointer to the

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.7 NPixelFormatGetNonSeparated Function

Retrieves a non-separated version of the NPixelFormat [1369].

C++

```
NResult N_API NPixelFormatGetNonSeparated(NPixelFormat_ pixelFormat, NPixelFormat_ *
pValue);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NPixelFormat_ * pValue	[out] A pointer to the

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.8 NPixelFormatGetPaletteSize Function

Retrieves the palette size for the pixel format.

C++

```
NResult N_API NPixelFormatGetPaletteSize(NPixelFormat_ pixelFormat, NInt length, NSizeType
* pValue);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NInt length	[in] Length of pixels.
NSizeType * pValue	[out] A pointer to the size of the palette.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.9 NPixelFormatGetPixelType Function

Retrieves the pixel type for the specified pixel format.

C++

```
NPixelType N_API NPixelFormatGetPixelType(NPixelFormat_ pixelFormat);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.10 NPixelFormatGetRowSize Function

Calculates number of bytes needed to store line of specified length of pixels with specified NPixelFormat [1369].

C++

```
NResult N_API NPixelFormatGetRowSize(NPixelFormat_ pixelFormat, NUInt length, NSizeType
alignment, NSizeType * pValue);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NUInt length	[in] Length of pixels.
NSizeType alignment	[in] Alignment.
NSizeType * pValue	[out] Pointer to NSizeType [1008] that receives the number of bytes needed to store line of specified length of pixels.

Returns

If the function succeeds and *pValue [870]* is not NULL [1075], the return value is N_OK [433].

N_E_ARGUMENT_OUT_OF_RANGE [428] error code is returned when *alignment* is less than 0.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.11 NPixelFormatGetSeparated Function

Retrieves a separated version of the NPixelFormat [1369].

C++

```
NResult N_API NPixelFormatGetSeparated(NPixelFormat_ pixelFormat, NPixelFormat_ * pValue);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.
NPixelFormat_ * pValue	[out] A pointer to the

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.12 NPixelFormatGetWithExtraChannel Function

Retrieves a version of the NPixelFormat [1369] with an extra channel.

C++

```
NResult N_API NPixelFormatGetWithExtraChannel(NPixelFormat_ pixelFormat, NExtraChannel
extraChannel, NPixelFormat_ * pValue);
```

Parameters

Parameters	Description
NPixelFormat_ pixelFormat	[in] Specifies pixel format of the image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NExtraChannel extraChannel	[in] The extra channel.
NPixelFormat_ * pValue	[out] A pointer to the

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.13 NPixelFormatIsValid Function

Checks if specified pixel format is valid.

C++

```
NBool N_API NPixelFormatIsValid(NPixelFormat_ value);
```

Parameters

Parameters	Description
NPixelFormat_ value	The NPixelFormat [1369] object.

Returns

True, if pixel format is valid; false otherwise.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.14 NPixelFormatToString Function

Converts a pixel format into a string.

C++

```
NResult N_API NPixelFormatToString(NPixelFormat pixelFormat, const NChar * szFormat,
HNString * phValue);
```

Parameters

Parameters	Description
NPixelFormat pixelFormat	[in] Specifies pixel format of the image.
const NChar * szFormat	[in] For internal use.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.15 NPixelTypeCanBeIndexed Function

Checks if specified pixel type can be indexed.

C++

```
NBool N_API NPixelTypeCanBeIndexed(NPixelType pixelType);
```

Parameters

Parameters	Description
NPixelType pixelType	The NPixelType [1369] object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPixelFormat Unit [1359]

7.1.5.16.1.16 NPixelTypeIsValid Function

Checks if specified pixel type is valid.

C++

```
NBool N_API NPixelTypeIsValid(NPixelType value);
```

Parameters

Parameters	Description
NPixelType value	The NPixelType [1369] object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NPixelFormat Unit [1359]

7.1.5.16.2 Structs, Records, Enums

7.1.5.16.2.1 NPixelType Enumeration

Specifies pixel type of each pixel in the image.

C++

```
typedef enum NPixelType_ {
  nptUndefined = 0,
  nptGrayscale = 1,
  nptRgb = 3
} NPixelType;
```

Members

Members	Description
nptUndefined = 0	Pixel type is undefined.
nptGrayscale = 1	Each pixel value is stored in 8 bits representing 256 shades of gray.
nptRgb = 3	Each pixel value is stored in 24 bits consisting of three 8-bit values representing red, green and blue color components.

Module

NPixelFormat Unit [1359]

7.1.5.16.3 Types

7.1.5.16.3.1 NPixelFormat Type

Specifies pixel format of each pixel in the image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
typedef NPixelFormat_ NPixelFormat;
```

Module

NPixelFormat Unit [1359]

7.1.5.16.3.2 NPixelFormat_ Type

Specifies pixel format of each pixel in the image.

C++

```
typedef NSampleFormat_ NPixelFormat_;
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4 Macros

7.1.5.16.4.1 NPF_GRAYSCALE_16S Macro

Specifies grayscale image pixel format where each pixel is represented by 16 bits per channel (channel format - signed integer).

C++

```
#define NPF_GRAYSCALE_16S 0x00411001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.2 NPF_GRAYSCALE_16U Macro

Specifies grayscale image pixel format where each pixel is represented by 16 bits per channel (channel format - unsigned integer).

C++

```
#define NPF_GRAYSCALE_16U 0x00401001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.3 NPF_GRAYSCALE_1U Macro

Specifies grayscale image pixel format where each pixel is represented by 1 bit per channel (channel format - unsigned integer).

C++

```
#define NPF_GRAYSCALE_1U 0x00001001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.4 NPF_GRAYSCALE_2U Macro

Specifies grayscale image pixel format where each pixel is represented by 2 bits per channel (channel format - unsigned integer).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_GRAYSCALE_2U 0x00101001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.5 NPF_GRAYSCALE_32F Macro

Specifies grayscale image pixel format where each pixel is represented by 32 bits per channel (channel format - floating point).

C++

```
#define NPF_GRAYSCALE_32F 0x00521001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.6 NPF_GRAYSCALE_32S Macro

Specifies grayscale image pixel format where each pixel is represented by 32 bits per channel (channel format - signed integer).

C++

```
#define NPF_GRAYSCALE_32S 0x00511001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.7 NPF_GRAYSCALE_32U Macro

Specifies grayscale image pixel format where each pixel is represented by 32 bits per channel (channel format - unsigned integer).

C++

```
#define NPF_GRAYSCALE_32U 0x00501001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.8 NPF_GRAYSCALE_4U Macro

Specifies grayscale image pixel format where each pixel is represented by 4 bits per channel (channel format - unsigned integer).

C++

```
#define NPF_GRAYSCALE_4U 0x00201001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.9 NPF_GRAYSCALE_64F Macro

Specifies grayscale image pixel format where each pixel is represented by 64 bits per channel (channel format - floating point).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_GRAYSCALE_64F 0x00621001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.10 NPF_GRAYSCALE_64S Macro

Specifies grayscale image pixel format where each pixel is represented by 64 bits per channel (channel format - signed integer).

C++

```
#define NPF_GRAYSCALE_64S 0x00611001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.11 NPF_GRAYSCALE_64U Macro

Specifies grayscale image pixel format where each pixel is represented by 64 bits per channel (channel format - unsigned integer).

C++

```
#define NPF_GRAYSCALE_64U 0x00601001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.12 NPF_GRAYSCALE_8S Macro

Specifies grayscale image pixel format where each pixel is represented by 8 bits per channel (channel format - signed integer).

C++

```
#define NPF_GRAYSCALE_8S 0x00311001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.13 NPF_GRAYSCALE_8U Macro

Specifies grayscale image pixel format where each pixel is represented by 8 bits per channel (channel format - unsigned integer).

C++

```
#define NPF_GRAYSCALE_8U 0x00301001
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.14 NPF_GRAYSCALE_A_16S Macro

Specifies grayscale image pixel format where each pixel is represented by 16 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_GRAYSCALE_A_16S 0x00412101
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.15 NPF_GRAYSCALE_A_16U Macro

Specifies grayscale image pixel format where each pixel is represented by 16 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_GRAYSCALE_A_16U 0x00402101
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.16 NPF_GRAYSCALE_A_32F Macro

Specifies grayscale image pixel format where each pixel is represented by 32 bits per alpha channel (channel format - floating point). Alpha channel specifies pixel transparency.

C++

```
#define NPF_GRAYSCALE_A_32F 0x00522101
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.17 NPF_GRAYSCALE_A_32S Macro

Specifies grayscale image pixel format where each pixel is represented by 32 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_GRAYSCALE_A_32S 0x00512101
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.18 NPF_GRAYSCALE_A_32U Macro

Specifies grayscale image pixel format where each pixel is represented by 32 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_GRAYSCALE_A_32U 0x00502101
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.19 NPF_GRAYSCALE_A_64F Macro

Specifies grayscale image pixel format where each pixel is represented by 64 bits per alpha channel (channel format - floating point). Alpha channel specifies pixel transparency.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_GRAYSCALE_A_64F 0x00622101
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.20 NPF_GRAYSCALE_A_64S Macro

Specifies grayscale image pixel format where each pixel is represented by 64 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_GRAYSCALE_A_64S 0x00612101
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.21 NPF_GRAYSCALE_A_64U Macro

Specifies grayscale image pixel format where each pixel is represented by 64 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_GRAYSCALE_A_64U 0x00602101
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.22 NPF_GRAYSCALE_A_8S Macro

Specifies grayscale image pixel format where each pixel is represented by 8 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_GRAYSCALE_A_8S 0x00312101
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.23 NPF_GRAYSCALE_A_8U Macro

Specifies grayscale image pixel format where each pixel is represented by 8 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_GRAYSCALE_A_8U 0x00302101
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.24 NPF_GRAYSCALE_PA_16S Macro

Specifies grayscale image pixel format where each pixel is represented by 16 bits per pre-multiplied alpha channel (channel format - signed integer).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_GRAYSCALE_PA_16S 0x00412201
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.25 NPF_GRAYSCALE_PA_16U Macro

Specifies grayscale image pixel format where each pixel is represented by 16 bits per pre-multiplied alpha channel (channel format - unsigned integer).

C++

```
#define NPF_GRAYSCALE_PA_16U 0x00402201
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.26 NPF_GRAYSCALE_PA_32F Macro

Specifies grayscale image pixel format where each pixel is represented by 32 bits per pre-multiplied alpha channel (channel format - floating point).

C++

```
#define NPF_GRAYSCALE_PA_32F 0x00522201
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.27 NPF_GRAYSCALE_PA_32S Macro

Specifies grayscale image pixel format where each pixel is represented by 32 bits per pre-multiplied alpha channel (channel format - signed integer).

C++

```
#define NPF_GRAYSCALE_PA_32S 0x00512201
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.28 NPF_GRAYSCALE_PA_32U Macro

Specifies grayscale image pixel format where each pixel is represented by 32 bits per pre-multiplied alpha channel (channel format - unsigned integer).

C++

```
#define NPF_GRAYSCALE_PA_32U 0x00502201
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.29 NPF_GRAYSCALE_PA_64F Macro

Specifies grayscale image pixel format where each pixel is represented by 64 bits per pre-multiplied alpha channel (channel format - floating point).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_GRAYSCALE_PA_64F 0x00622201
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.30 NPF_GRAYSCALE_PA_64S Macro

Specifies grayscale image pixel format where each pixel is represented by 64 bits per pre-multiplied alpha channel (channel format - signed integer).

C++

```
#define NPF_GRAYSCALE_PA_64S 0x00612201
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.31 NPF_GRAYSCALE_PA_64U Macro

Specifies grayscale image pixel format where each pixel is represented by 64 bits per pre-multiplied alpha channel (channel format - unsigned integer).

C++

```
#define NPF_GRAYSCALE_PA_64U 0x00602201
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.32 NPF_GRAYSCALE_PA_8S Macro

Specifies grayscale image pixel format where each pixel is represented by 8 bits per pre-multiplied alpha channel (channel format - signed integer).

C++

```
#define NPF_GRAYSCALE_PA_8S 0x00312201
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.33 NPF_GRAYSCALE_PA_8U Macro

Specifies grayscale image pixel format where each pixel is represented by 8 bits per pre-multiplied alpha channel (channel format - unsigned integer).

C++

```
#define NPF_GRAYSCALE_PA_8U 0x00302201
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.34 NPF_RGB_16S Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per channel (channel format - signed integer).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_RGB_16S 0x00413003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.35 NPF_RGB_16U Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per channel (channel format - unsigned integer).

C++

```
#define NPF_RGB_16U 0x00403003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.36 NPF_RGB_32F Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per channel (channel format - floating point).

C++

```
#define NPF_RGB_32F 0x00523003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.37 NPF_RGB_32S Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per channel (channel format - signed integer).

C++

```
#define NPF_RGB_32S 0x00513003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.38 NPF_RGB_32U Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per channel (channel format - unsigned integer).

C++

```
#define NPF_RGB_32U 0x00503003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.39 NPF_RGB_64F Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per channel (channel format - floating point).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_RGB_64F 0x00623003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.40 NPF_RGB_64S Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per channel (channel format - signed integer).

C++

```
#define NPF_RGB_64S 0x00613003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.41 NPF_RGB_64U Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per channel (channel format - unsigned integer).

C++

```
#define NPF_RGB_64U 0x00603003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.42 NPF_RGB_8S Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - signed integer).

C++

```
#define NPF_RGB_8S 0x00313003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.43 NPF_RGB_8U Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer).

C++

```
#define NPF_RGB_8U 0x00303003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.44 NPF_RGB_8U_INDEXED_1 Macro

Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer). Also 1 bit is for saving index to image palette.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_RGB_8U_INDEXED_1 0x01303003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.45 NPF_RGB_8U_INDEXED_2 Macro

Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer). Also 2 bits is for saving index to image palette.

C++

```
#define NPF_RGB_8U_INDEXED_2 0x02303003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.46 NPF_RGB_8U_INDEXED_4 Macro

Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer). Also 4 bits is for saving index to image palette.

C++

```
#define NPF_RGB_8U_INDEXED_4 0x03303003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.47 NPF_RGB_8U_INDEXED_8 Macro

Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer). Also 8 bits is for saving index to image palette.

C++

```
#define NPF_RGB_8U_INDEXED_8 0x04303003
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.48 NPF_RGB_A_16S Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_RGB_A_16S 0x00414103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.49 NPF_RGB_A_16U Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_RGB_A_16U 0x00404103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.50 NPF_RGB_A_32F Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per alpha channel (channel format - floating point). Alpha channel specifies pixel transparency.

C++

```
#define NPF_RGB_A_32F 0x00524103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.51 NPF_RGB_A_32S Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_RGB_A_32S 0x00514103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.52 NPF_RGB_A_32U Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_RGB_A_32U 0x00504103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.53 NPF_RGB_A_64F Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per alpha channel (channel format - floating point). Alpha channel specifies pixel transparency.

C++

```
#define NPF_RGB_A_64F 0x00624103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.54 NPF_RGB_A_64S Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_RGB_A_64S 0x00614103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.55 NPF_RGB_A_64U Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_RGB_A_64U 0x00604103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.56 NPF_RGB_A_8S Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_RGB_A_8S 0x00314103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.57 NPF_RGB_A_8U Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.

C++

```
#define NPF_RGB_A_8U 0x00304103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.58 NPF_RGB_A_8U_INDEXED_1 Macro

Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Also 1 bit is for saving index to image palette.

C++

```
#define NPF_RGB_A_8U_INDEXED_1 0x01304103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.59 NPF_RGB_A_8U_INDEXED_2 Macro

Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Also 2 bits is for saving index to image palette.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_RGB_A_8U_INDEXED_2 0x02304103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.60 NPF_RGB_A_8U_INDEXED_4 Macro

Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Also 4 bits is for saving index to image palette.

C++

```
#define NPF_RGB_A_8U_INDEXED_4 0x03304103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.61 NPF_RGB_A_8U_INDEXED_8 Macro

Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Also 8 bits is for saving index to image palette.

C++

```
#define NPF_RGB_A_8U_INDEXED_8 0x04304103
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.62 NPF_RGB_PA_16S Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per pre-multiplied alpha channel (channel format - signed integer).

C++

```
#define NPF_RGB_PA_16S 0x00414203
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.63 NPF_RGB_PA_16U Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per pre-multiplied alpha channel (channel format - unsigned integer).

C++

```
#define NPF_RGB_PA_16U 0x00404203
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.64 NPF_RGB_PA_32F Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per pre-multiplied alpha channel (channel format - floating point).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_RGB_PA_32F 0x00524203
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.65 NPF_RGB_PA_32S Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per pre-multiplied alpha channel (channel format - signed integer).

C++

```
#define NPF_RGB_PA_32S 0x00514203
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.66 NPF_RGB_PA_32U Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per pre-multiplied alpha channel (channel format - unsigned integer).

C++

```
#define NPF_RGB_PA_32U 0x00504203
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.67 NPF_RGB_PA_64F Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per pre-multiplied alpha channel (channel format - floating point).

C++

```
#define NPF_RGB_PA_64F 0x00624203
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.68 NPF_RGB_PA_64S Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per pre-multiplied alpha channel (channel format - signed integer).

C++

```
#define NPF_RGB_PA_64S 0x00614203
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.69 NPF_RGB_PA_64U Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per pre-multiplied alpha channel (channel format - unsigned integer).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NPF_RGB_PA_64U 0x00604203
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.70 NPF_RGB_PA_8S Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per pre-multiplied alpha channel (channel format - signed integer).

C++

```
#define NPF_RGB_PA_8S 0x00314203
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.71 NPF_RGB_PA_8U Macro

Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per pre-multiplied alpha channel (channel format - unsigned integer).

C++

```
#define NPF_RGB_PA_8U 0x00304203
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.72 NPF_UNDEFINED Macro

Indicates undefined pixel format.

C++

```
#define NPF_UNDEFINED 0
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.73 npfGrayscale Macro

Indicates a grayscale pixel.

C++

```
#define npfGrayscale NPF_GRAYSCALE_8U
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.74 npfMonochrome Macro

Indicates a monochrome pixel.

C++

```
#define npfMonochrome NPF_GRAYSCALE_1U
```

Module

NPixelFormat Unit [1359]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.16.4.75 npfRgb Macro

Indiactes an RGB pixel.

C++

```
#define npfRgb NPF_RGB_8U
```

Module

NPixelFormat Unit [1359]

7.1.5.16.4.76 NPixelFormatToString Macro

Picks a Unicode (W) or ANSCII (A) version of the NPixelFormatToString function.

C++

```
#define NPixelFormatToString N_FUNC_AW(NPixelFormatToString)
```

Module

NPixelFormat Unit [1359]

7.1.5.16.5 Files

7.1.5.16.5.1 NPixelFormat.h

Header file for NPixelFormat [1369] module. Provides functionality for working with image pixel format.

Enumerations

	Name	Description
	NPixelType_ [1369]	Specifies pixel type of each pixel in the image.
	NPixelType [1369]	Specifies pixel type of each pixel in the image.

Functions

	Name	Description
	NPixelFormatCalcRowSize [1363]	Calculates number of bytes needed to store line of specified length of pixels with specified NPixelFormat [1369].
	NPixelFormatCreate [1364]	Creates a new NPixelFormat [1369].
	NPixelFormatGetBitsPerPixel [1364]	Retrieves number of bits used to store a pixel from NPixelFormat [1369].
	NPixelFormatGetBytesPerPixel [1365]	Retrieves number of bytes used to store a pixel from NPixelFormat [1369].
	NPixelFormatGetIndexed [1365]	Retrieves an indexed version of the NPixelFormat [1369].
	NPixelFormatGetNonIndexed [1365]	Retrieves a non-indexed version of the NPixelFormat [1369].
	NPixelFormatGetNonSeparated [1366]	Retrieves a non-separated version of the NPixelFormat [1369].
	NPixelFormatGetPaletteSize [1366]	Retrieves the palette size for the pixel format.
	NPixelFormatGetPixelType [1366]	Retrieves the pixel type for the specified pixel format.
	NPixelFormatGetRowSize [1367]	Calculates number of bytes needed to store line of specified length of pixels with specified NPixelFormat [1369].
	NPixelFormatGetSeparated [1367]	Retrieves a separated version of the NPixelFormat [1369].
	NPixelFormatGetWithExtraChannel [1367]	Retrieves a version of the NPixelFormat [1369] with an extra channel.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NPixelFormatIsValid [1368]	Checks if specified pixel format is valid.
	NPixelFormatToString [1368]	Converts a pixel format into a string.
	NPixelTypeCanBeIndexed [1368]	Checks if specified pixel type can be indexed.
	NPixelTypeIsValid [1369]	Checks if specified pixel type is valid.

Macros

Name	Description
NPF_GRAYSCALE_16S [1370]	Specifies grayscale image pixel format where each pixel is represented by 16 bits per channel (channel format - signed integer).
NPF_GRAYSCALE_16U [1370]	Specifies grayscale image pixel format where each pixel is represented by 16 bits per channel (channel format - unsigned integer).
NPF_GRAYSCALE_1U [1370]	Specifies grayscale image pixel format where each pixel is represented by 1 bit per channel (channel format - unsigned integer).
NPF_GRAYSCALE_2U [1370]	Specifies grayscale image pixel format where each pixel is represented by 2 bits per channel (channel format - unsigned integer).
NPF_GRAYSCALE_32F [1371]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per channel (channel format - floating point).
NPF_GRAYSCALE_32S [1371]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per channel (channel format - signed integer).
NPF_GRAYSCALE_32U [1371]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per channel (channel format - unsigned integer).
NPF_GRAYSCALE_4U [1371]	Specifies grayscale image pixel format where each pixel is represented by 4 bits per channel (channel format - unsigned integer).
NPF_GRAYSCALE_64F [1371]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per channel (channel format - floating point).
NPF_GRAYSCALE_64S [1372]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per channel (channel format - signed integer).
NPF_GRAYSCALE_64U [1372]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per channel (channel format - unsigned integer).
NPF_GRAYSCALE_8S [1372]	Specifies grayscale image pixel format where each pixel is represented by 8 bits per channel (channel format - signed integer).
NPF_GRAYSCALE_8U [1372]	Specifies grayscale image pixel format where each pixel is represented by 8 bits per channel (channel format - unsigned integer).
NPF_GRAYSCALE_A_16S [1372]	Specifies grayscale image pixel format where each pixel is represented by 16 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_16U [1373]	Specifies grayscale image pixel format where each pixel is represented by 16 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_32F [1373]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per alpha channel (channel format - floating point). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_32S [1373]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_32U [1373]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_64F [1373]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per alpha channel (channel format - floating point). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_64S [1374]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPF_GRAYSCALE_A_64U [1374]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_8S [1374]	Specifies grayscale image pixel format where each pixel is represented by 8 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_A_8U [1374]	Specifies grayscale image pixel format where each pixel is represented by 8 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_GRAYSCALE_PA_16S [1374]	Specifies grayscale image pixel format where each pixel is represented by 16 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_GRAYSCALE_PA_16U [1375]	Specifies grayscale image pixel format where each pixel is represented by 16 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_GRAYSCALE_PA_32F [1375]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per pre-multiplied alpha channel (channel format - floating point).
NPF_GRAYSCALE_PA_32S [1375]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_GRAYSCALE_PA_32U [1375]	Specifies grayscale image pixel format where each pixel is represented by 32 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_GRAYSCALE_PA_64F [1375]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per pre-multiplied alpha channel (channel format - floating point).
NPF_GRAYSCALE_PA_64S [1376]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_GRAYSCALE_PA_64U [1376]	Specifies grayscale image pixel format where each pixel is represented by 64 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_GRAYSCALE_PA_8S [1376]	Specifies grayscale image pixel format where each pixel is represented by 8 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_GRAYSCALE_PA_8U [1376]	Specifies grayscale image pixel format where each pixel is represented by 8 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_RGB_16S [1376]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per channel (channel format - signed integer).
NPF_RGB_16U [1377]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per channel (channel format - unsigned integer).
NPF_RGB_32F [1377]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per channel (channel format - floating point).
NPF_RGB_32S [1377]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per channel (channel format - signed integer).
NPF_RGB_32U [1377]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per channel (channel format - unsigned integer).
NPF_RGB_64F [1377]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per channel (channel format - floating point).
NPF_RGB_64S [1378]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per channel (channel format - signed integer).
NPF_RGB_64U [1378]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per channel (channel format - unsigned integer).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPF_RGB_8S [1378]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - signed integer).
NPF_RGB_8U [1378]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer).
NPF_RGB_8U_INDEXED_1 [1378]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer). Also 1 bit is for saving index to image palette.
NPF_RGB_8U_INDEXED_2 [1379]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer). Also 2 bits is for saving index to image palette.
NPF_RGB_8U_INDEXED_4 [1379]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer). Also 4 bits is for saving index to image palette.
NPF_RGB_8U_INDEXED_8 [1379]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per channel (channel format - unsigned integer). Also 8 bits is for saving index to image palette.
NPF_RGB_A_16S [1379]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_16U [1379]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_32F [1380]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per alpha channel (channel format - floating point). Alpha channel specifies pixel transparency.
NPF_RGB_A_32S [1380]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_32U [1380]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_64F [1380]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per alpha channel (channel format - floating point). Alpha channel specifies pixel transparency.
NPF_RGB_A_64S [1380]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_64U [1381]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_8S [1381]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - signed integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_8U [1381]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Alpha channel specifies pixel transparency.
NPF_RGB_A_8U_INDEXED_1 [1381]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Also 1 bit is for saving index to image palette.
NPF_RGB_A_8U_INDEXED_2 [1381]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Also 2 bits is for saving index to image palette.
NPF_RGB_A_8U_INDEXED_4 [1382]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Also 4 bits is for saving index to image palette.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPF_RGB_A_8U_INDEXED_8 [1382]	Specifies indexed Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per alpha channel (channel format - unsigned integer). Also 8 bits is for saving index to image palette.
NPF_RGB_PA_16S [1382]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_RGB_PA_16U [1382]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 16 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_RGB_PA_32F [1382]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per pre-multiplied alpha channel (channel format - floating point).
NPF_RGB_PA_32S [1383]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_RGB_PA_32U [1383]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 32 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_RGB_PA_64F [1383]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per pre-multiplied alpha channel (channel format - floating point).
NPF_RGB_PA_64S [1383]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_RGB_PA_64U [1383]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 64 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_RGB_PA_8S [1384]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per pre-multiplied alpha channel (channel format - signed integer).
NPF_RGB_PA_8U [1384]	Specifies Rgb image pixel format where each pixel (Red, Green, Blue) is represented by 8 bits per pre-multiplied alpha channel (channel format - unsigned integer).
NPF_UNDEFINED [1384]	Indicates undefined pixel format.
npfGrayscale [1384]	Indicates a grayscale pixel.
npfMonochrome [1384]	Indicates a monochrome pixel.
npfRgb [1385]	Indiactes an RGB pixel.
NPixelFormatToString [1385]	Picks a Unicode (W) or ANSCII (A) version of the NPixelFormatToString function.

Module

NPixelFormat Unit [1359]

Types

Name	Description
NPixelFormat [1369]	Specifies pixel format of each pixel in the image.
NPixelFormat_ [1370]	Specifies pixel format of each pixel in the image.

7.1.5.17 NRgb Unit

Provides functionality for working with Rgb image data.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Files

Name	Description
NRgb.h [1391]	Header file for NRgb [1391] Unit.

Functions

	Name	Description
	NRgbToString [1390]	Converts RGB value into NString [678].

Macros

Name	Description
NRgbToString [1391]	Picks a Unicode (W) or ANSCII (A) version of the NRgbToString function.

Structs, Records, Enums

	Name	Description
	NRgb_ [1390]	Represents an RGB color.

Types

Name	Description
NRgb [1391]	Defines the RGB color.

7.1.5.17.1 Functions

7.1.5.17.1.1 NRgbToString Function

Converts RGB value into NString [678].

C++

```
NResult N_API NRgbToString(const NRgb * pValue, const NChar * szFormat, HNString * phValue);
```

Parameters

Parameters	Description
const NRgb * pValue	[in] A pointer to the RGB value to be converted.
const NChar * szFormat	[in] For internal use.
HNString * phValue	[out] A pointer to the converted NString [678].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NRgb Unit [1389]

7.1.5.17.2 Structs, Records, Enums

7.1.5.17.2.1 NRgb_ Structure

Represents an RGB color.

C++

```
struct NRgb_ {
  NByte Red;
  NByte Green;
  NByte Blue;
};
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Members

Members	Description
NByte Red;	Red component value of this NRgb_.
NByte Green;	Green component value of this NRgb_.
NByte Blue;	Blue component value of this NRgb_.

Module

NRgb Unit [1389]

7.1.5.17.3 Types

7.1.5.17.3.1 NRgb Type

Defines the RGB color.

C++

```
typedef struct NRgb_ NRgb;
```

Module

NRgb Unit [1389]

7.1.5.17.4 Macros

7.1.5.17.4.1 NRgbToString Macro

Picks a Unicode (W) or ANSCII (A) version of the NRgbToString function.

C++

```
#define NRgbToString N_FUNC_AW(NRgbToString)
```

Module

NRgb Unit [1389]

7.1.5.17.5 Files

7.1.5.17.5.1 NRgb.h

Header file for NRgb [1391] Unit.

Functions

	Name	Description
	NRgbToString [1390]	Converts RGB value into NString [678].

Macros

Name	Description
NRgbToString [1391]	Picks a Unicode (W) or ANSCII (A) version of the NRgbToString function.

Module

NRgb Unit [1389]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Structures

	Name	Description
	NRgb_ [1390]	Represents an RGB color.

Types

Name	Description
NRgb [1391]	Defines the RGB color.

7.1.5.18 NRgbImage Unit

Provides functionality for managing 24-bit RGB images.

Files

Name	Description
NRgbImage.h [1395]	Header file for NRgbImage module. Provides functionality for managing 24-bit RGB images.

Functions

	Name	Description
	NRgbImageGetMaxValue [1392]	Retrieves maximum pixel value for the specified RGB image.
	NRgbImageGetMinValue [1392]	Retrieves minimum pixel value for the specified RGB image.
	NRgbImageGetPixel [1393]	Retrieves value of pixel at the specified coordinates in 24-bit RGB image.
	NRgbImageSetMaxValue [1393]	Sets the maximum pixel value for the specified RGB image.
	NRgbImageSetMinValue [1394]	Sets the minimum pixel value for the specified RGB image.
	NRgbImageSetPixel [1394]	Sets value of pixel at the specified coordinates in 24-bit RGB image.

7.1.5.18.1 Functions

7.1.5.18.1.1 NRgbImageGetMaxValue Function

Retrieves maximum pixel value for the specified RGB image.

C++

```
NResult N_API NRgbImageGetMaxValue(HNRgbImage hImage, struct NRgb_ * pValue);
```

Parameters

Parameters	Description
HNRgbImage hImage	[in] Handle to the image.
struct NRgb_ * pValue	[out] Pointer to the maximum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NRgbImage Unit [1392]

7.1.5.18.1.2 NRgbImageGetMinValue Function

Retrieves minimum pixel value for the specified RGB image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NRgbImageGetMinValue(HNRgbImage hImage, struct NRgb_ * pValue);
```

Parameters

Parameters	Description
HNRgbImage hImage	[in] Handle to the image.
struct NRgb_ * pValue	[out] Pointer to the minimum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NRgbImage Unit [1392]

7.1.5.18.1.3 NRgbImageGetPixel Function

Retrieves value of pixel at the specified coordinates in 24-bit RGB image.

C++

```
NResult N_API NRgbImageGetPixel(HNRgbImage hImage, NUInt x, NUInt y, struct NRgb_ * pValue);
```

Parameters

Parameters	Description
HNRgbImage hImage	[in] Handle to image.
NUInt x	[in] Specifies x-coordinate of the pixel.
NUInt y	[in] Specifies y-coordinate of the pixel.
struct NRgb_ * pValue	[out] Pointer to NRgb [1391] that receives pixel value.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *pValue [870]* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when x is greater than or equal to image width. - or - y is greater than or equal to image height.
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when image pixel format is not equal to *npfRgb [1385]*.

Module

NRgbImage Unit [1392]

7.1.5.18.1.4 NRgbImageSetMaxValue Function

Sets the maximum pixel value for the specified RGB image.

C++

```
NResult N_API NRgbImageSetMaxValue(HNRgbImage hImage, const struct NRgb_ * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNRgbImage hImage	[in] Handle to the image.
const struct NRgb_ * pValue	[in] Maximum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NRgbImage Unit [1392]

7.1.5.18.1.5 NRgbImageSetMinValue Function

Sets the minimum pixel value for the specified RGB image.

C++

```
NResult N_API NRgbImageSetMinValue(HNRgbImage hImage, const struct NRgb_ * pValue);
```

Parameters

Parameters	Description
HNRgbImage hImage	[in] Handle to the image.
const struct NRgb_ * pValue	[in] Minimum pixel value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NRgbImage Unit [1392]

7.1.5.18.1.6 NRgbImageSetPixel Function

Sets value of pixel at the specified coordinates in 24-bit RGB image.

C++

```
NResult N_API NRgbImageSetPixel(HNRgbImage hImage, NUInt x, NUInt y, const struct NRgb_ *
pValue);
```

Parameters

Parameters	Description
HNRgbImage hImage	[in] Handle to image.
NUInt x	[in] Specifies x-coordinate of the pixel.
NUInt y	[in] Specifies y-coordinate of the pixel.
const struct NRgb_ * pValue	[out] Pointer to NRgb [1391] that specifies new pixel value.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *pValue [870]* is NULL [1075].

7

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when x is greater than or equal to image width. - or - y is greater than or equal to image height.
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when image pixel format is not equal to *npfRgb [1385]*.

Module

NRgbImage Unit [1392]

7.1.5.18.2 Files

7.1.5.18.2.1 NRgbImage.h

Header file for NRgbImage [1392] module. Provides functionality for managing 24-bit RGB images.

Functions

	Name	Description
	NRgbImageGetMaxValue [1392]	Retrieves maximum pixel value for the specified RGB image.
	NRgbImageGetMinValue [1392]	Retrieves minimum pixel value for the specified RGB image.
	NRgbImageGetPixel [1393]	Retrieves value of pixel at the specified coordinates in 24-bit RGB image.
	NRgbImageSetMaxValue [1393]	Sets the maximum pixel value for the specified RGB image.
	NRgbImageSetMinValue [1394]	Sets the minimum pixel value for the specified RGB image.
	NRgbImageSetPixel [1394]	Sets value of pixel at the specified coordinates in 24-bit RGB image.

Module

NRgbImage Unit [1392]

7.1.5.19 NVideoFormat Unit

Files

Name	Description
NVideoFormat.h [1412]	Header file for NVideoFormat module. Provides functionality for managing video formats.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NVideoFormatCreate [1398]	Creates default video format. Video format properties can be modified and retrieved using following functions: • NVideoFormatGetFrameRate [1398] • NVideoFormatGetHeight [1398] • NVideoFormatGetInterlaceMode [1399] • NVideoFormatGetPixelAspectRatio [1399] • NVideoFormatGetWidth [1399] • NVideoFormatSetFrameRate [1400] • NVideoFormatSetHeight [1401] • NVideoFormatSetInterlaceMode [1401] • NVideoFormatSetPixelAspectRatio [1401] • NVideoFormatSetWidth [1402]
	NVideoFormatGetFrameRate [1398]	Gets frame rate of video format.
	NVideoFormatGetHeight [1398]	Gets frame height in pixels for specified video format.
	NVideoFormatGetInterlaceMode [1399]	Gets interlace mode [1354] for specified video format.
	NVideoFormatGetPixelAspectRatio [1399]	Gets frame aspect ratio for specified video format.
	NVideoFormatGetWidth [1399]	Gets frame width of specified video format.
	NVideoFormatMediaSubtypeToString [1400]	Writes video format string into memory buffer.
	NVideoFormatSetFrameRate [1400]	Sets video format frame rate.
	NVideoFormatSetHeight [1401]	Sets video format frame height (in pixels).
	NVideoFormatSetInterlaceMode [1401]	Sets video format interlace mode [1354].
	NVideoFormatSetPixelAspectRatio [1401]	Sets video format aspect ratio.
	NVideoFormatSetWidth [1402]	Sets video format frame width (in pixels).

Macros

Name	Description
N_VIDEO_FORMAT_AI44 [1402]	Video format AI44
N_VIDEO_FORMAT_ARGB32 [1402]	Video format ARGB32.
N_VIDEO_FORMAT_AVC1 [1402]	Video format AVC1.
N_VIDEO_FORMAT_AYUV [1402]	Video format AYUV.
N_VIDEO_FORMAT_DV25 [1403]	Video format DVCPRO 25 (525-60 or 625-50).
N_VIDEO_FORMAT_DV50 [1403]	Video format DVCPRO 50 (525-60 or 625-50).
N_VIDEO_FORMAT_DVC [1403]	Video format DVC/DV Video.
N_VIDEO_FORMAT_DVH1 [1403]	Video format DVCPRO 100 (1080/60i, 1080/50i, or 720/60P).
N_VIDEO_FORMAT_DVHD [1403]	Video format HD-DVCR (1125-60 or 1250-50).
N_VIDEO_FORMAT_DVSD [1404]	Video format SDL-DVCR (525-60 or 625-50).
N_VIDEO_FORMAT_DVSL [1404]	Video format SD-DVCR (525-60 or 625-50).
N_VIDEO_FORMAT_DX50 [1404]	Video format DX50.
N_VIDEO_FORMAT_H264 [1404]	Video format H.264.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_VIDEO_FORMAT_I420 [1404]	Video format I420.
N_VIDEO_FORMAT_IYUV [1404]	Video format IYUV.
N_VIDEO_FORMAT_L16 [1405]	Video format L16.
N_VIDEO_FORMAT_L8 [1405]	Video format L8.
N_VIDEO_FORMAT_M4S2 [1405]	Video format MPEG-4 part 2.
N_VIDEO_FORMAT_MJPG [1405]	Video format Motion JPEG.
N_VIDEO_FORMAT_MP43 [1405]	Video format Microsoft MPEG 4 codec version 3.
N_VIDEO_FORMAT_MP4S [1405]	Video format ISO MPEG 4 codec version 1.
N_VIDEO_FORMAT_MP4V [1406]	Video format MPEG-4 part 2 video.
N_VIDEO_FORMAT_MPG1 [1406]	Video format MPEG-1.
N_VIDEO_FORMAT_MSS1 [1406]	Video format Windows Media Screen codec version 1.
N_VIDEO_FORMAT_MSS2 [1406]	Video format Windows Media Video 9 Screen codec.
N_VIDEO_FORMAT_NV11 [1406]	Video format NV11.
N_VIDEO_FORMAT_NV12 [1407]	Video format NV12.
N_VIDEO_FORMAT_P010 [1407]	Video format P010.
N_VIDEO_FORMAT_P016 [1407]	Video format P016.
N_VIDEO_FORMAT_P210 [1407]	Video format P210.
N_VIDEO_FORMAT_P216 [1407]	Video format P216.
N_VIDEO_FORMAT_RGB24 [1407]	Video format RGB, 24 bpp.
N_VIDEO_FORMAT_RGB32 [1408]	Video format RGB, 32 bpp.
N_VIDEO_FORMAT_RGB555 [1408]	Video format RGB 555, 16 bpp.
N_VIDEO_FORMAT_RGB565 [1408]	Video format RGB 565, 16 bpp.
N_VIDEO_FORMAT_RGB8 [1408]	Video format RGB, 8 bits per pixel.
N_VIDEO_FORMAT_UNKNOWN [1408]	Unknown video format.
N_VIDEO_FORMAT_UYVY [1408]	Video format UYVY.
N_VIDEO_FORMAT_V210 [1409]	Video format V210.
N_VIDEO_FORMAT_V216 [1409]	Video format V216.
N_VIDEO_FORMAT_V410 [1409]	Video format V40.
N_VIDEO_FORMAT_WMV1 [1409]	Video format Windows Media Video codec version 7.
N_VIDEO_FORMAT_WMV2 [1409]	Video format Windows Media Video 8 codec.
N_VIDEO_FORMAT_WMV3 [1410]	Video format Windows Media Video 9 codec.
N_VIDEO_FORMAT_WVC1 [1410]	Video format SMPTE 421M ("VC-1").
N_VIDEO_FORMAT_Y210 [1410]	Video format Y210.
N_VIDEO_FORMAT_Y216 [1410]	Video format Y216.
N_VIDEO_FORMAT_Y410 [1410]	Video format Y40.
N_VIDEO_FORMAT_Y416 [1410]	Video format Y416.
N_VIDEO_FORMAT_Y41P [1411]	Video format Y41P.
N_VIDEO_FORMAT_Y41T [1411]	Video format Y41T.
N_VIDEO_FORMAT_Y42T [1411]	Video format Y42T.
N_VIDEO_FORMAT_YUY2 [1411]	Video format YUY2.
N_VIDEO_FORMAT_YV12 [1411]	Video format YV12.
N_VIDEO_FORMAT_YVU9 [1411]	Video format YVU9.
N_VIDEO_FORMAT_YVYU [1412]	Video format YVYU.
NVideoFormatMediaSubtypeToString [1412]	Picks a Unicode (W) or ANSCII (A) version of the NVideoFormatMediaSubtype function.

7.1.5.19.1 Functions

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.19.1.1 NVideoFormatCreate Function

Creates default video format. Video format properties can be modified and retrieved using following functions:

- NVideoFormatGetFrameRate [1398]
- NVideoFormatGetHeight [1398]
- NVideoFormatGetInterlaceMode [1399]
- NVideoFormatGetPixelAspectRatio [1399]
- NVideoFormatGetWidth [1399]
- NVideoFormatSetFrameRate [1400]
- NVideoFormatSetHeight [1401]
- NVideoFormatSetInterlaceMode [1401]
- NVideoFormatSetPixelAspectRatio [1401]
- NVideoFormatSetWidth [1402]

C++

```
NResult N_API NVideoFormatCreate(HNVideoFormat * phVideoFormat);
```

Parameters

Parameters	Description
HNVideoFormat * phVideoFormat	Pointer to video format handle. Handle will be initialized if function succeeds.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoFormat Unit [1395]

7.1.5.19.1.2 NVideoFormatGetFrameRate Function

Gets frame rate of video format.

C++

```
NResult N_API NVideoFormatGetFrameRate(HNVideoFormat hVideoFormat, struct NURational_ *
pValue);
```

Parameters

Parameters	Description
HNVideoFormat hVideoFormat	[in] Handle to video format.
struct NURational_ * pValue	[out] Frame rate value (frames per second).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoFormat Unit [1395]

7.1.5.19.1.3 NVideoFormatGetHeight Function

Gets frame height in pixels for specified video format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NVideoFormatGetHeight(HNVideoFormat hVideoFormat, NUInt * pValue);
```

Parameters

Parameters	Description
HNVideoFormat hVideoFormat	[in] Handle to video format.
NUInt * pValue	[out] Frame height (in pixels).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoFormat Unit [1395]

7.1.5.19.1.4 NVideoFormatGetInterlaceMode Function

Gets interlace mode [1354] for specified video format.

C++

```
NResult N_API NVideoFormatGetInterlaceMode(HNVideoFormat hVideoFormat, NVideoInterlaceMode
* pValue);
```

Parameters

Parameters	Description
HNVideoFormat hVideoFormat	[in] Handle to video format.
NVideoInterlaceMode * pValue	[out] Interlace mode value [1354].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoFormat Unit [1395]

7.1.5.19.1.5 NVideoFormatGetPixelAspectRatio Function

Gets frame aspect ratio for specified video format.

C++

```
NResult N_API NVideoFormatGetPixelAspectRatio(HNVideoFormat hVideoFormat, struct
NURational_ * pValue);
```

Parameters

Parameters	Description
HNVideoFormat hVideoFormat	[in] Handle to video format.
struct NURational_ * pValue	[out] Aspect ratio.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoFormat Unit [1395]

7.1.5.19.1.6 NVideoFormatGetWidth Function

Gets frame width of specified video format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NVideoFormatGetWidth(HNVideoFormat hVideoFormat, NUInt * pValue);
```

Parameters

Parameters	Description
HNVideoFormat hVideoFormat	[in] Handle to video format.
NUInt * pValue	[out] Frame width (in pixels).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoFormat Unit [1395]

7.1.5.19.1.7 NVideoFormatMediaSubtypeToString Function

Writes video format string into memory buffer.

C++

```
NResult N_API NVideoFormatMediaSubtypeToString(NUInt value, const NChar * szFormat,
HNString * phValue);
```

Parameters

Parameters	Description
NUInt value	[in] Video format value.
const NChar * szFormat	[in] String format. Value can be NULL [1075].
HNString * phValue	[out] Pointer to handle which contains string representation of NVideoFormat [1395] media subtype.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoFormat Unit [1395]

7.1.5.19.1.8 NVideoFormatSetFrameRate Function

Sets video format frame rate.

C++

```
NResult N_API NVideoFormatSetFrameRate(HNVideoFormat hVideoFormat, const struct NURational_
* pValue);
```

Parameters

Parameters	Description
HNVideoFormat hVideoFormat	[in] Handle to video format.
const struct NURational_ * pValue	[in] Frame rate.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoFormat Unit [1395]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.19.1.9 NVideoFormatSetHeight Function

Sets video format frame height (in pixels).

C++

```
NResult N_API NVideoFormatSetHeight(HNVideoFormat hVideoFormat, NUInt value);
```

Parameters

Parameters	Description
HNVideoFormat hVideoFormat	[in] Handle to video format.
NUInt value	[in] Frame height (in pixels).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoFormat Unit [1395]

7.1.5.19.1.10 NVideoFormatSetInterlaceMode Function

Sets video format interlace mode [1354].

C++

```
NResult N_API NVideoFormatSetInterlaceMode(HNVideoFormat hVideoFormat, NVideoInterlaceMode
value);
```

Parameters

Parameters	Description
HNVideoFormat hVideoFormat	[in] Handle to video format.
NVideoInterlaceMode value	[in] Interlace mode [1354].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoFormat Unit [1395]

7.1.5.19.1.11 NVideoFormatSetPixelAspectRatio Function

Sets video format aspect ratio.

C++

```
NResult N_API NVideoFormatSetPixelAspectRatio(HNVideoFormat hVideoFormat, const struct
NURational_ * pValue);
```

Parameters

Parameters	Description
HNVideoFormat hVideoFormat	[in] Handle to video format.
const struct NURational_ * pValue	[in] Aspect ratio.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoFormat Unit [1395]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.19.1.12 NVideoFormatSetWidth Function

Sets video format frame width (in pixels).

C++

```
NResult N_API NVideoFormatSetWidth(HNVideoFormat hVideoFormat, NUInt value);
```

Parameters

Parameters	Description
HNVideoFormat hVideoFormat	[in] Handle to video format.
NUInt value	[in] Frame width (in pixels).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoFormat Unit [1395]

7.1.5.19.2 Macros

7.1.5.19.2.1 N_VIDEO_FORMAT_AI44 Macro

Video format AI44

C++

```
#define N_VIDEO_FORMAT_AI44 0x34344941
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.2 N_VIDEO_FORMAT_ARGB32 Macro

Video format ARGB32.

C++

```
#define N_VIDEO_FORMAT_ARGB32 21
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.3 N_VIDEO_FORMAT_AVC1 Macro

Video format AVC1.

C++

```
#define N_VIDEO_FORMAT_AVC1 0x31435641
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.4 N_VIDEO_FORMAT_AYUV Macro

Video format AYUV.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_VIDEO_FORMAT_AYUV 0x56555941
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.5 N_VIDEO_FORMAT_DV25 Macro

Video format DVCPRO 25 (525-60 or 625-50).

C++

```
#define N_VIDEO_FORMAT_DV25 0x35327664
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.6 N_VIDEO_FORMAT_DV50 Macro

Video format DVCPRO 50 (525-60 or 625-50).

C++

```
#define N_VIDEO_FORMAT_DV50 0x30357664
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.7 N_VIDEO_FORMAT_DVC Macro

Video format DVC/DV Video.

C++

```
#define N_VIDEO_FORMAT_DVC 0x20637664
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.8 N_VIDEO_FORMAT_DVH1 Macro

Video format DVCPRO 100 (1080/60i, 1080/50i, or 720/60P).

C++

```
#define N_VIDEO_FORMAT_DVH1 0x31687664
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.9 N_VIDEO_FORMAT_DVHD Macro

Video format HD-DVCR (1125-60 or 1250-50).

C++

```
#define N_VIDEO_FORMAT_DVHD 0x64687664
```

Module

NVideoFormat Unit [1395]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.19.2.10 N_VIDEO_FORMAT_DVSD Macro

Video format SDL-DVCR (525-60 or 625-50).

C++

```
#define N_VIDEO_FORMAT_DVSD 0x64737664
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.11 N_VIDEO_FORMAT_DVSL Macro

Video format SD-DVCR (525-60 or 625-50).

C++

```
#define N_VIDEO_FORMAT_DVSL 0x6C737664
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.12 N_VIDEO_FORMAT_DX50 Macro

Video format DX50.

C++

```
#define N_VIDEO_FORMAT_DX50 0x30355844
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.13 N_VIDEO_FORMAT_H264 Macro

Video format H.264.

C++

```
#define N_VIDEO_FORMAT_H264 0x34363248
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.14 N_VIDEO_FORMAT_I420 Macro

Video format I420.

C++

```
#define N_VIDEO_FORMAT_I420 0x30323449
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.15 N_VIDEO_FORMAT_IYUV Macro

Video format IYUV.

C++

```
#define N_VIDEO_FORMAT_IYUV 0x56555949
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NVideoFormat Unit [1395]

7.1.5.19.2.16 N_VIDEO_FORMAT_L16 Macro

Video format L16.

C++

```
#define N_VIDEO_FORMAT_L16 81
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.17 N_VIDEO_FORMAT_L8 Macro

Video format L8.

C++

```
#define N_VIDEO_FORMAT_L8 50
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.18 N_VIDEO_FORMAT_M4S2 Macro

Video format MPEG-4 part 2.

C++

```
#define N_VIDEO_FORMAT_M4S2 0x3253344D
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.19 N_VIDEO_FORMAT_MJPG Macro

Video format Motion JPEG.

C++

```
#define N_VIDEO_FORMAT_MJPG 0x47504A4D
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.20 N_VIDEO_FORMAT_MP43 Macro

Video format Microsoft MPEG 4 codec version 3.

C++

```
#define N_VIDEO_FORMAT_MP43 0x3334504D
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.21 N_VIDEO_FORMAT_MP4S Macro

Video format ISO MPEG 4 codec version 1.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_VIDEO_FORMAT_MP4S 0x5334504D
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.22 N_VIDEO_FORMAT_MP4V Macro

Video format MPEG-4 part 2 video.

C++

```
#define N_VIDEO_FORMAT_MP4V 0x5634504D
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.23 N_VIDEO_FORMAT_MPG1 Macro

Video format MPEG-1.

C++

```
#define N_VIDEO_FORMAT_MPG1 0x3147504D
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.24 N_VIDEO_FORMAT_MSS1 Macro

Video format Windows Media Screen codec version 1.

C++

```
#define N_VIDEO_FORMAT_MSS1 0x3153534D
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.25 N_VIDEO_FORMAT_MSS2 Macro

Video format Windows Media Video 9 Screen codec.

C++

```
#define N_VIDEO_FORMAT_MSS2 0x3253534D
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.26 N_VIDEO_FORMAT_NV11 Macro

Video format NV11.

C++

```
#define N_VIDEO_FORMAT_NV11 0x3131564E
```

Module

NVideoFormat Unit [1395]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.19.2.27 N_VIDEO_FORMAT_NV12 Macro

Video format NV12.

C++

```
#define N_VIDEO_FORMAT_NV12 0x3231564E
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.28 N_VIDEO_FORMAT_P010 Macro

Video format P010.

C++

```
#define N_VIDEO_FORMAT_P010 0x30313050
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.29 N_VIDEO_FORMAT_P016 Macro

Video format P016.

C++

```
#define N_VIDEO_FORMAT_P016 0x36313050
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.30 N_VIDEO_FORMAT_P210 Macro

Video format P210.

C++

```
#define N_VIDEO_FORMAT_P210 0x30313250
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.31 N_VIDEO_FORMAT_P216 Macro

Video format P216.

C++

```
#define N_VIDEO_FORMAT_P216 0x36313250
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.32 N_VIDEO_FORMAT_RGB24 Macro

Video format RGB, 24 bpp.

C++

```
#define N_VIDEO_FORMAT_RGB24 20
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NVideoFormat Unit [1395]

7.1.5.19.2.33 N_VIDEO_FORMAT_RGB32 Macro

Video format RGB, 32 bpp.

C++

```
#define N_VIDEO_FORMAT_RGB32 22
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.34 N_VIDEO_FORMAT_RGB555 Macro

Video format RGB 555, 16 bpp.

C++

```
#define N_VIDEO_FORMAT_RGB555 24
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.35 N_VIDEO_FORMAT_RGB565 Macro

Video format RGB 565, 16 bpp.

C++

```
#define N_VIDEO_FORMAT_RGB565 23
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.36 N_VIDEO_FORMAT_RGB8 Macro

Video format RGB, 8 bits per pixel.

C++

```
#define N_VIDEO_FORMAT_RGB8 41
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.37 N_VIDEO_FORMAT_UNKNOWN Macro

Unknown video format.

C++

```
#define N_VIDEO_FORMAT_UNKNOWN 0
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.38 N_VIDEO_FORMAT_UYVY Macro

Video format UYVY.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_VIDEO_FORMAT_UYVY 0x59565955
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.39 N_VIDEO_FORMAT_V210 Macro

Video format V210.

C++

```
#define N_VIDEO_FORMAT_V210 0x30313276
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.40 N_VIDEO_FORMAT_V216 Macro

Video format V216.

C++

```
#define N_VIDEO_FORMAT_V216 0x36313276
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.41 N_VIDEO_FORMAT_V410 Macro

Video format V40.

C++

```
#define N_VIDEO_FORMAT_V410 0x30313476
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.42 N_VIDEO_FORMAT_WMV1 Macro

Video format Windows Media Video codec version 7.

C++

```
#define N_VIDEO_FORMAT_WMV1 0x31564D57
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.43 N_VIDEO_FORMAT_WMV2 Macro

Video format Windows Media Video 8 codec.

C++

```
#define N_VIDEO_FORMAT_WMV2 0x32564D57
```

Module

NVideoFormat Unit [1395]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.19.2.44 N_VIDEO_FORMAT_WMV3 Macro

Video format Windows Media Video 9 codec.

C++

```
#define N_VIDEO_FORMAT_WMV3 0x33564D57
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.45 N_VIDEO_FORMAT_WVC1 Macro

Video format SMPTE 421M ("VC-1").

C++

```
#define N_VIDEO_FORMAT_WVC1 0x31435657
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.46 N_VIDEO_FORMAT_Y210 Macro

Video format Y210.

C++

```
#define N_VIDEO_FORMAT_Y210 0x30313259
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.47 N_VIDEO_FORMAT_Y216 Macro

Video format Y216.

C++

```
#define N_VIDEO_FORMAT_Y216 0x36313259
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.48 N_VIDEO_FORMAT_Y410 Macro

Video format Y40.

C++

```
#define N_VIDEO_FORMAT_Y410 0x30313459
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.49 N_VIDEO_FORMAT_Y416 Macro

Video format Y416.

C++

```
#define N_VIDEO_FORMAT_Y416 0x36313459
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NVideoFormat Unit [1395]

7.1.5.19.2.50 N_VIDEO_FORMAT_Y41P Macro

Video format Y41P.

C++

```
#define N_VIDEO_FORMAT_Y41P 0x50313459
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.51 N_VIDEO_FORMAT_Y41T Macro

Video format Y41T.

C++

```
#define N_VIDEO_FORMAT_Y41T 0x54313459
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.52 N_VIDEO_FORMAT_Y42T Macro

Video format Y42T.

C++

```
#define N_VIDEO_FORMAT_Y42T 0x54323459
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.53 N_VIDEO_FORMAT_YUY2 Macro

Video format YUY2.

C++

```
#define N_VIDEO_FORMAT_YUY2 0x32595559
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.54 N_VIDEO_FORMAT_YV12 Macro

Video format YV12.

C++

```
#define N_VIDEO_FORMAT_YV12 0x32315659
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.55 N_VIDEO_FORMAT_YVU9 Macro

Video format YVU9.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define N_VIDEO_FORMAT_YVU9 0x39555659
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.56 N_VIDEO_FORMAT_YVYU Macro

Video format YVYU.

C++

```
#define N_VIDEO_FORMAT_YVYU 0x55595659
```

Module

NVideoFormat Unit [1395]

7.1.5.19.2.57 NVideoFormatMediaSubtypeToString Macro

Picks a Unicode (W) or ANSCII (A) version of the NVideoFormatMediaSubtype function.

C++

```
#define NVideoFormatMediaSubtypeToString N_FUNC_AW(NVideoFormatMediaSubtypeToString)
```

Module

NVideoFormat Unit [1395]

7.1.5.19.3 Files

7.1.5.19.3.1 NVideoFormat.h

Header file for NVideoFormat [1395] module. Provides functionality for managing video formats.

Functions

	Name	Description
	NVideoFormatCreate [1398]	Creates default video format. Video format properties can be modified and retrieved using following functions: • NVideoFormatGetFrameRate [1398] • NVideoFormatGetHeight [1398] • NVideoFormatGetInterlaceMode [1399] • NVideoFormatGetPixelAspectRatio [1399] • NVideoFormatGetWidth [1399] • NVideoFormatSetFrameRate [1400] • NVideoFormatSetHeight [1401] • NVideoFormatSetInterlaceMode [1401] • NVideoFormatSetPixelAspectRatio [1401] • NVideoFormatSetWidth [1402]
	NVideoFormatGetFrameRate [1398]	Gets frame rate of video format.
	NVideoFormatGetHeight [1398]	Gets frame height in pixels for specified video format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NVideoFormatGetInterlaceMode [1399]	Gets interlace mode [1354] for specified video format.
	NVideoFormatGetPixelAspectRatio [1399]	Gets frame aspect ratio for specified video format.
	NVideoFormatGetWidth [1399]	Gets frame width of specified video format.
	NVideoFormatMediaSubtypeToString [1400]	Writes video format string into memory buffer.
	NVideoFormatSetFrameRate [1400]	Sets video format frame rate.
	NVideoFormatSetHeight [1401]	Sets video format frame height (in pixels).
	NVideoFormatSetInterlaceMode [1401]	Sets video format interlace mode [1354].
	NVideoFormatSetPixelAspectRatio [1401]	Sets video format aspect ratio.
	NVideoFormatSetWidth [1402]	Sets video format frame width (in pixels).

Macros

Name	Description
N_VIDEO_FORMAT_AI44 [1402]	Video format AI44
N_VIDEO_FORMAT_ARGB32 [1402]	Video format ARGB32.
N_VIDEO_FORMAT_AVC1 [1402]	Video format AVC1.
N_VIDEO_FORMAT_AYUV [1402]	Video format AYUV.
N_VIDEO_FORMAT_DV25 [1403]	Video format DVCPRO 25 (525-60 or 625-50).
N_VIDEO_FORMAT_DV50 [1403]	Video format DVCPRO 50 (525-60 or 625-50).
N_VIDEO_FORMAT_DVC [1403]	Video format DVC/DV Video.
N_VIDEO_FORMAT_DVH1 [1403]	Video format DVCPRO 100 (1080/60i, 1080/50i, or 720/60P).
N_VIDEO_FORMAT_DVHD [1403]	Video format HD-DVCR (1125-60 or 1250-50).
N_VIDEO_FORMAT_DVSD [1404]	Video format SDL-DVCR (525-60 or 625-50).
N_VIDEO_FORMAT_DVSL [1404]	Video format SD-DVCR (525-60 or 625-50).
N_VIDEO_FORMAT_DX50 [1404]	Video format DX50.
N_VIDEO_FORMAT_H264 [1404]	Video format H.264.
N_VIDEO_FORMAT_I420 [1404]	Video format I420.
N_VIDEO_FORMAT_IYUV [1404]	Video format IYUV.
N_VIDEO_FORMAT_L16 [1405]	Video format L16.
N_VIDEO_FORMAT_L8 [1405]	Video format L8.
N_VIDEO_FORMAT_M4S2 [1405]	Video format MPEG-4 part 2.
N_VIDEO_FORMAT_MJPG [1405]	Video format Motion JPEG.
N_VIDEO_FORMAT_MP43 [1405]	Video format Microsoft MPEG 4 codec version 3.
N_VIDEO_FORMAT_MP4S [1405]	Video format ISO MPEG 4 codec version 1.
N_VIDEO_FORMAT_MP4V [1406]	Video format MPEG-4 part 2 video.
N_VIDEO_FORMAT_MPG1 [1406]	Video format MPEG-1.
N_VIDEO_FORMAT_MSS1 [1406]	Video format Windows Media Screen codec version 1.
N_VIDEO_FORMAT_MSS2 [1406]	Video format Windows Media Video 9 Screen codec.
N_VIDEO_FORMAT_NV11 [1406]	Video format NV11.
N_VIDEO_FORMAT_NV12 [1407]	Video format NV12.
N_VIDEO_FORMAT_P010 [1407]	Video format P010.
N_VIDEO_FORMAT_P016 [1407]	Video format P016.
N_VIDEO_FORMAT_P210 [1407]	Video format P210.
N_VIDEO_FORMAT_P216 [1407]	Video format P216.
N_VIDEO_FORMAT_RGB24 [1407]	Video format RGB, 24 bpp.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_VIDEO_FORMAT_RGB32 [1408]	Video format RGB, 32 bpp.
N_VIDEO_FORMAT_RGB555 [1408]	Video format RGB 555, 16 bpp.
N_VIDEO_FORMAT_RGB565 [1408]	Video format RGB 565, 16 bpp.
N_VIDEO_FORMAT_RGB8 [1408]	Video format RGB, 8 bits per pixel.
N_VIDEO_FORMAT_UNKNOWN [1408]	Unknown video format.
N_VIDEO_FORMAT_UYVY [1408]	Video format UYVY.
N_VIDEO_FORMAT_V210 [1409]	Video format V210.
N_VIDEO_FORMAT_V216 [1409]	Video format V216.
N_VIDEO_FORMAT_V410 [1409]	Video format V40.
N_VIDEO_FORMAT_WMV1 [1409]	Video format Windows Media Video codec version 7.
N_VIDEO_FORMAT_WMV2 [1409]	Video format Windows Media Video 8 codec.
N_VIDEO_FORMAT_WMV3 [1410]	Video format Windows Media Video 9 codec.
N_VIDEO_FORMAT_WVC1 [1410]	Video format SMPTE 421M ("VC-1").
N_VIDEO_FORMAT_Y210 [1410]	Video format Y210.
N_VIDEO_FORMAT_Y216 [1410]	Video format Y216.
N_VIDEO_FORMAT_Y410 [1410]	Video format Y40.
N_VIDEO_FORMAT_Y416 [1410]	Video format Y416.
N_VIDEO_FORMAT_Y41P [1411]	Video format Y41P.
N_VIDEO_FORMAT_Y41T [1411]	Video format Y41T.
N_VIDEO_FORMAT_Y42T [1411]	Video format Y42T.
N_VIDEO_FORMAT_YUY2 [1411]	Video format YUY2.
N_VIDEO_FORMAT_YV12 [1411]	Video format YV12.
N_VIDEO_FORMAT_YVU9 [1411]	Video format YVU9.
N_VIDEO_FORMAT_YVYU [1412]	Video format YVYU.
NVideoFormatMediaSubtypeToString [1412]	Picks a Unicode (W) or ANSCII (A) version of the NVideoFormatMediaSubtype function.

Module

NVideoFormat Unit [1395]

7.1.5.20 NVideoReader Unit

Files

Name	Description
NVideoReader.h [1418]	Header file for NVideoReader module. Provides functionality for reading video files.

Functions

	Name	Description
	NVideoReaderCreateFromFile [1415]	Creates a HNVideoReader for the specified video file.
	NVideoReaderGetFrame [1415]	Reads one frame from video file.
	NVideoReaderGetFrameCount [1416]	Counts number of frames in the video file which was assigned to the HNVideoReader handle (using function NVideoReaderCreateFromFile [1415]).
	NVideoReaderGetFrameHeight [1416]	The function retrieves height of frames in the video file which was assigned to the HNVideoReader handle (using function NVideoReaderCreateFromFile [1415]).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NVideoReaderGetFrameRate [1417]	The function calculates a frame rate (number of frames presented in one second) of the video file. The file was assigned to the HNVideoReader handle (using function NVideoReaderCreateFromFile [1415]).
	NVideoReaderGetFrameWidth [1417]	The function retrieves width of frames in the video file which was assigned to the HNVideoReader handle (using function NVideoReaderCreateFromFile [1415]).

Macros

Name	Description
NVideoReaderCreateFromFile [1418]	Picks a Unicode (W) or ANSCII (A) version of the NVideoReaderCreateFromFile function.
NVideoReaderFree [1418]	Frees the specified video reader.

7.1.5.20.1 Functions

7.1.5.20.1.1 NVideoReaderCreateFromFile Function

Creates a HNVideoReader for the specified video file.

C++

```
NResult N_API NVideoReaderCreateFromFile(const NChar * szFileName, HNVideoReader *
phReader);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] A filename of a video file to be opened.
HNVideoReader * phReader	[out] A handle of HNVideoReader type to be allocated and assigned to the specified video file.

Returns

Return value	Description
N_OK [433]	If function succeeds it returns N_OK [433].
N_E_ARGUMENT_NULL [427]	If provided file name is NULL [1075], function fails and returns.
N_E_FAILED [429].	If function fails to open a file or assign a handle to the file.

Remarks

When handle is not needed anymore, it must be freed using NVideoFree function.

Module

NVideoReader Unit [1414]

7.1.5.20.1.2 NVideoReaderGetFrame Function

Reads one frame from video file.

C++

```
NResult N_API NVideoReaderGetFrame(HNVideoReader hReader, NInt frameIndex, HNImage *
phFrame);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNVideoReader hReader	[in] A handle of HNVideoReader type. This is the same handle as was created using NVideoReaderCreateFromFile [1415] function.
NInt frameIndex	[in] An index of a frame to be read form the video file.
HNImage * phFrame	[out] An image which will be filled with information available in the frame.

Returns

Return value	Description
N_OK [433]	If function succeeds it returns N_OK [433].
N_E_ARGUMENT_NULL [427]	A HNVideoReader handle or phFrame are NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	A frame index is larger than the number of frames in the video file.
N_E_FAILED [429]	Failed to grab frame form video file.

Module

NVideoReader Unit [1414]

7.1.5.20.1.3 NVideoReaderGetFrameCount Function

Counts number of frames in the video file which was assigned to the HNVideoReader handle (using function NVideoReaderCreateFromFile [1415]).

C++

```
NResult N_API NVideoReaderGetFrameCount(HNVideoReader hReader, NInt * pValue);
```

Parameters

Parameters	Description
HNVideoReader hReader	[in] A handle of HNVideoReader type. This is the same handle as was created using NVideo-ReaderCreateFromFile function.
NInt * pValue	[out] A number of frames in the video file.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT [427] NULL [1075]	Function might fail if a handle of HNVideoReader is not created.

Module

NVideoReader Unit [1414]

7.1.5.20.1.4 NVideoReaderGetFrameHeight Function

The function retrieves height of frames in the video file which was assigned to the HNVideoReader handle (using function NVideoReaderCreateFromFile [1415]).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NVideoReaderGetFrameHeight(HNVideoReader hReader, NUInt * pValue);
```

Parameters

Parameters	Description
HNVideoReader hReader	[in] A handle of HNVideoReader type. This is the same handle as was created using NVideoReaderCreateFromFile [1415] function.
NUInt * pValue	[out] A height of frames in the video file.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT [427] NULL [1075].	Function might fail if a handle of HNVideoReader is not created.

Module

NVideoReader Unit [1414]

7.1.5.20.1.5 NVideoReaderGetFrameRate Function

The function calculates a frame rate (number of frames presented in one second) of the video file. The file was assigned to the HNVideoReader handle (using function NVideoReaderCreateFromFile [1415]).

C++

```
NResult N_API NVideoReaderGetFrameRate(HNVideoReader hReader, NDouble * pValue);
```

Parameters

Parameters	Description
HNVideoReader hReader	[in] A handle of HNVideoReader type. This is the same handle as was created using NVideoReaderCreateFromFile [1415] function.
NDouble * pValue	[out] A number of frames presented in one second when the video file is playing.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	Function might fail if a handle of HNVideoReader is not created.

Module

NVideoReader Unit [1414]

7.1.5.20.1.6 NVideoReaderGetFrameWidth Function

The function retrieves width of frames in the video file which was assigned to the HNVideoReader handle (using function NVideoReaderCreateFromFile [1415]).

C++

```
NResult N_API NVideoReaderGetFrameWidth(HNVideoReader hReader, NUInt * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNVideoReader hReader	A handle of HNVideoReader type. This is the same handle as was created using NVideoReaderCreateFromFile [1415] function.
NUInt * pValue	A width of frames in the video file.

Returns

Return value	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	Function might fail if a handle of HNVideoReader is not created.

Module

NVideoReader Unit [1414]

7.1.5.20.2 Macros

7.1.5.20.2.1 NVideoReaderCreateFromFile Macro

Picks a Unicode (W) or ANSCII (A) version of the NVideoReaderCreateFromFile function.

C++

```
#define NVideoReaderCreateFromFile N_FUNC_AW(NVideoReaderCreateFromFile)
```

Module

NVideoReader Unit [1414]

7.1.5.20.2.2 NVideoReaderFree Macro

Frees the specified video reader.

C++

```
#define NVideoReaderFree(hReader) NObjectFree(hReader)
```

Parameters

Parameters	Description
hReader	[in] Handle to the reader to be freed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoReader Unit [1414]

7.1.5.20.3 Files

7.1.5.20.3.1 NVideoReader.h

Header file for NVideoReader [1414] module. Provides functionality for reading video files.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	NVideoReaderCreateFromFile [1415]	Creates a HNVideoReader for the specified video file.
	NVideoReaderGetFrame [1415]	Reads one frame from video file.
	NVideoReaderGetFrameCount [1416]	Counts number of frames in the video file which was assigned to the HNVideoReader handle (using function NVideoReaderCreateFromFile [1415]).
	NVideoReaderGetFrameHeight [1416]	The function retrieves height of frames in the video file which was assigned to the HNVideoReader handle (using function NVideoReaderCreateFromFile [1415]).
	NVideoReaderGetFrameRate [1417]	The function calculates a frame rate (number of frames presented in one second) of the video file. The file was assigned to the HNVideoReader handle (using function NVideoReaderCreateFromFile [1415]).
	NVideoReaderGetFrameWidth [1417]	The function retrieves width of frames in the video file which was assigned to the HNVideoReader handle (using function NVideoReaderCreateFromFile [1415]).

Macros

Name	Description
NVideoReaderCreateFromFile [1418]	Picks a Unicode (W) or ANSCII (A) version of the NVideoReaderCreateFromFile function.
NVideoReaderFree [1418]	Frees the specified video reader.

Module

NVideoReader Unit [1414]

7.1.5.21 NVideoWriter Module

Provides functionality for writing to a video file.

Remarks

When using XVID encoder make sure that automatic optimisations are turned off. Otherwise NVideoWriter may crash due to a bug in XVID code.

If writing video files with XVID still crashes, then turn off all optimisations. Optimisation settings can be located in Options->Other Options...->Common starting in a popup dialog box where you choose XVID codec.

Files

Name	Description
NVideoWriter.h [1422]	Header file for the NVideoWriter module. Provides functionality for writing to a video file.
NVideoWriterOptions.h [1423]	Header file for the NVideoWriter module. Provides options for writing to a video file.

Functions

	Name	Description
	NVideoWriterCreateFile [1420]	Creates a HNVideoWriter [1421] for the specified video file.
	NVideoWriterWriteFrame [1420]	Appends a frame to previously created video file.
	NVideoWriterOptionsCreateWithGui [1421]	Creates a video writer options with GUI.
	HNVideoWriter [1421]	Handle to NVideoWriter object.
	HNVideoWriterOptions [1421]	A handle of information about video codec.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Macros

Name	Description
NVideoWriterCreateFile [1421]	Picks a Unicode (W) or ANSCII (A) version of the NVideoWriterCreateFile function.
NVideoWriterFree [1422]	Frees the specified video writer.
NVideoWriterOptionsFree [1422]	Frees the specified video writer options.

7.1.5.21.1 Functions

7.1.5.21.1.1 NVideoWriterCreateFile Function

Creates a HNVideoWriter [1421] for the specified video file.

C++

```
NResult N_API NVideoWriterCreateFile(const NChar * szFileName, NUInt width, NUInt height,
NDouble frameRate, HNVideoWriterOptions hOptions, HNVideoWriter * phWriter);
```

Parameters

Parameters	Description
const NChar * szFileName	A filename of a video file to be opened.
NUInt width	A width of video frame.
NUInt height	A height of video frame.
NDouble frameRate	Frame rate.
HNVideoWriterOptions hOptions	A handle to Video writer options structure.
HNVideoWriter * phWriter	A handle of HNVideoWriter [1421] type to be allocated and assigned to the specified video file.

Returns

If function succeeds it returns N_OK [433].

If provided file name is NULL [1075], function fails and returns N_E_ARGUMENT_NULL [427].

If function fails to create a file or assign a handle to the file, it returns an error of N_E_FAILED [429].

Remarks

When handle is not needed anymore, it must be freed using NVideoWriterClose function.

Module

NVideoWriter Module [1419]

7.1.5.21.1.2 NVideoWriterWriteFrame Function

Appends a frame to previously created video file.

C++

```
NResult N_API NVideoWriterWriteFrame(HNVideoWriter hWriter, HNImage hFrame);
```

Parameters

Parameters	Description
HNVideoWriter hWriter	A handle of HNVideoWriter [1421] type. This is the same handle as was created using NVideoWriterCreateFile [1420] function.
HNImage hFrame	[out] A handle to HNImage [1276].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

If function succeeds it returns N_OK [433].

Module

NVideoWriter Module [1419]

7.1.5.21.1.3 NVideoWriterOptionsCreateWithGui Function

Creates a video writer options with GUI.

C++

```
NResult N_API NVideoWriterOptionsCreateWithGui(NHandle hParentWnd, HNVideoWriterOptions *
phOptions);
```

Parameters

Parameters	Description
NHandle hParentWnd	[in] Handle to the parent window.
HNVideoWriterOptions * phOptions	[out] Pointer to a handle to NVideoWriterOptions object.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoWriter Module [1419]

7.1.5.21.1.4 HNVideoWriter

Handle to NVideoWriter [1419] object.

Module

NVideoWriter Module [1419]

7.1.5.21.1.5 HNVideoWriterOptions

A handle of information about video codec.

Remarks

This handle is allocated by calling NVideoChooseCompression function with appropriate parameters. This handle can be associated with created video file later on by calling NVideoSetCompression function. When writing to the file is done this handle can be freed by calling NVideoFreeCompression function. If this function is called between NVideoSetCompression and NVideoWriterClose functions expect unexpected behaviour.

Module

NVideoWriter Module [1419]

7.1.5.21.2 Macros

7.1.5.21.2.1 NVideoWriterCreateFile Macro

Picks a Unicode (W) or ANSCII (A) version of the NVideoWriterCreateFile function.

C++

```
#define NVideoWriterCreateFile N_FUNC_AW(NVideoWriterCreateFile)
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NVideoWriter Module [1419]

7.1.5.21.2.2 NVideoWriterFree Macro

Frees the specified video writer.

C++

```
#define NVideoWriterFree(hWriter) NObjectFree(hWriter)
```

Parameters

Parameters	Description
hWriter	[in] Handle to the writer to be freed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoWriter Module [1419]

7.1.5.21.2.3 NVideoWriterOptionsFree Macro

Frees the specified video writer options.

C++

```
#define NVideoWriterOptionsFree(hOptions) NObjectFree(hOptions)
```

Parameters

Parameters	Description
hOptions	[in] Handle to the options to be freed.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NVideoWriter Module [1419]

7.1.5.21.3 Files

7.1.5.21.3.1 NVideoWriter.h

Header file for the NVideoWriter [1419] module. Provides functionality for writing to a video file.

Functions

	Name	Description
	NVideoWriterCreateFile [1420]	Creates a HNVideoWriter [1421] for the specified video file.
	NVideoWriterWriteFrame [1420]	Appends a frame to previously created video file.

Macros

Name	Description
NVideoWriterCreateFile [1421]	Picks a Unicode (W) or ANSCII (A) version of the NVideoWriterCreateFile function.
NVideoWriterFree [1422]	Frees the specified video writer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NVideoWriter Module [1419]

7.1.5.21.3.2 NVideoWriterOptions.h

Header file for the NVideoWriter [1419] module. Provides options for writing to a video file.

Functions

	Name	Description
	NVideoWriterOptionsCreateWithGui [1421]	Creates a video writer options with GUI.

Macros

Name	Description
NVideoWriterOptionsFree [1422]	Frees the specified video writer options.

Module

NVideoWriter Module [1419]

7.1.5.22 Png Unit

Provides functionality for loading and saving images in PNG format.

Files

Name	Description
Png.h [1428]	Header file for Png module. Provides functionality for loading and saving images in PNG format.

Functions

	Name	Description
	PngInfoGetCompressionLevel [1424]	Retrieves compression level of a PNG image.
	PngInfoSetCompressionLevel [1424]	Sets compression level of a PNG image.
	PngLoadImageFromFile [1424]	Loads image from PNG file.
	PngLoadImageFromMemory [1425]	Loads image from the memory buffer containing PNG file.
	PngLoadImageFromStream [1425]	Loads image from the stream containing PNG file.
	PngSaveImageToFile [1426]	Saves image to file in PNG format.
	PngSaveImageToMemory [1426]	Saves image to the memory buffer in PNG format.
	PngSaveImageToStream [1427]	Saves image to the stream in PNG format.

Macros

Name	Description
PNG_DEFAULT_COMPRESSION_LEVEL [1428]	Specifies default PNG compression level.

7.1.5.22.1 Functions

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.22.1.1 PngInfoGetCompressionLevel Function

Retrieves compression level of a PNG image.

C++

```
NResult N_API PngInfoGetCompressionLevel(HPngInfo hInfo, NInt * pValue);
```

Parameters

Parameters	Description
HPngInfo hInfo	[in] A handle to PngInfo.
NInt * pValue	[out] A pointer to NInt [1006] value defining the compression level.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Png Unit [1423]

7.1.5.22.1.2 PngInfoSetCompressionLevel Function

Sets compression level of a PNG image.

C++

```
NResult N_API PngInfoSetCompressionLevel(HPngInfo hInfo, NInt value);
```

Parameters

Parameters	Description
HPngInfo hInfo	[in] A handle to PngInfo.
NInt value	[in] New compression level value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Png Unit [1423]

7.1.5.22.1.3 PngLoadImageFromFile Function

Loads image from PNG file.

C++

```
NResult N_API PngLoadImageFromFile(const NChar * szFileName, HNImage * phImage);
```

Parameters

Parameters	Description
const NChar * szFileName	[out] Points to string that specifies file name.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Png Unit [1423]

7.1.5.22.1.4 PngLoadImageFromMemory Function

Loads image from the memory buffer containing PNG file.

C++

```
NResult N_API PngLoadImageFromMemory(const void * pBuffer, NSizeType bufferSize, HNImage *
phImage);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer.
NSizeType bufferSize	[in] Size of memory buffer.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pBuffer* is NULL [1075] and *bufferSize* is not equal to zero or *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file contained in memory buffer specified by buffer is invalid.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Png Unit [1423]

7.1.5.22.1.5 PngLoadImageFromStream Function

Loads image from the stream containing PNG file.

C++

```
NResult N_API PngLoadImageFromStream(HNStream hStream, HNImage * phImage);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to HNStream.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when*phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file contained in memory buffer specified by *buffer* is invalid.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Png Unit [1423]

7.1.5.22.1.6 PngSaveImageToFile Function

Saves image to file in PNG format.

C++

```
NResult N_API PngSaveImageToFile(HNImage hImage, NInt compressionLevel, const NChar *
szFileName);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
NInt compressionLevel	[in] Specifies level of PNG compression. The value can be in range [0, 9]. 0 - uncompressed, 9 - maximal compression level. See PNG_DEFAULT_COMPRESSION_LEVEL [1428].
const NChar * szFileName	[out] Points to string that specifies file name.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage* or *szFileName* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *compressionLevel* is less than 0 or greater than 9.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Png Unit [1423]

7.1.5.22.1.7 PngSaveImageToMemory Function

Saves image to the memory buffer in PNG format.

C++

```
NResult N_API PngSaveImageToMemory(HNImage hImage, NInt compressionLevel, void * *
ppBuffer, NSizeType * pBufferSize);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
NInt compressionLevel	[in] Specifies level of PNG compression. The value can be in range [0, 9]. 0 - uncompressed, 9 - maximal compression level. See PNG_DEFAULT_COMPRESSION_LEVEL [1428].
void * * ppBuffer	[out] Pointer to void * that receives pointer to allocated memory buffer.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage*, *ppBuffer* or *pBufferSize* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *compressionLevel* is less than 0 or greater than 9.
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. This error code is returned when there was not enough memory to allocate memory buffer.

Remarks

This is a low-level function and can be changed in future version of the library.

Memory buffer allocated by this function must be deallocated by NFree [506] function when it is no longer needed.

Module

Png Unit [1423]

7.1.5.22.1.8 PngSaveImageToStream Function

Saves image to the stream in PNG format.

C++

```
NResult N_API PngSaveImageToStream(HNImage hImage, NInt compressionLevel, HNStream hStream);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
NInt compressionLevel	[in] Specifies level of PNG compression. The value can be in range [0, 9]. 0 - uncompressed, 9 - maximal compression level. See PNG_DEFAULT_COMPRESSION_LEVEL [1428].
HNStream hStream	[out] Handle to HNStream where image will be written.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage*, *pBuffer* or *pBufferLength* is NULL [1075].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *compressionLevel* is less than 0 or greater than 9.
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. This error code is returned when there was not enough memory to allocate memory buffer.

Remarks

This is a low-level function and can be changed in future version of the library.

Memory buffer allocated by this function must be deallocated by NFree [506] function when it is no longer needed.

Module

Png Unit [1423]

7.1.5.22.2 Macros

7.1.5.22.2.1 PNG_DEFAULT_COMPRESSION_LEVEL Macro

Specifies default PNG compression level.

C++

```
#define PNG_DEFAULT_COMPRESSION_LEVEL 6
```

Module

Png Unit [1423]

7.1.5.22.3 Files

7.1.5.22.3.1 Png.h

Header file for Png [1423] module. Provides functionality for loading and saving images in PNG format.

Functions

	Name	Description
	PngInfoGetCompressionLevel [1424]	Retrieves compression level of a PNG image.
	PngInfoSetCompressionLevel [1424]	Sets compression level of a PNG image.
	PngLoadImageFromFile [1424]	Loads image from PNG file.
	PngLoadImageFromMemory [1425]	Loads image from the memory buffer containing PNG file.
	PngLoadImageFromStream [1425]	Loads image from the stream containing PNG file.
	PngSaveImageToFile [1426]	Saves image to file in PNG format.
	PngSaveImageToMemory [1426]	Saves image to the memory buffer in PNG format.
	PngSaveImageToStream [1427]	Saves image to the stream in PNG format.

Macros

Name	Description
PNG_DEFAULT_COMPRESSION_LEVEL [1428]	Specifies default PNG compression level.

Module

Png Unit [1423]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.23 Tiff Unit

Provides functionality for loading images in TIFF format.

Files

Name	Description
Tiff.h [1430]	Header file for Tiff module. Provides functionality for loading images in TIFF format.

Functions

	Name	Description
	TiffLoadImageFromFile [1429]	Loads image from TIFF file.
	TiffLoadImageFromMemory [1429]	Loads image from memory buffer containing TIFF file.
	TiffLoadImageFromStream [1430]	Loads image from stream containing TIFF file.

7.1.5.23.1 Functions

7.1.5.23.1.1 TiffLoadImageFromFile Function

Loads image from TIFF file.

C++

```
NResult N_API TiffLoadImageFromFile(const NChar * szFileName, HNImage * phImage);
```

Parameters

Parameters	Description
const NChar * szFileName	[out] Points to string that specifies file name.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	the function fails, this error can be returned. This error code is returned when *szFileName* or *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file specified by *szFileName* is invalid.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Tiff Unit [1429]

7.1.5.23.1.2 TiffLoadImageFromMemory Function

Loads image from memory buffer containing TIFF file.

C++

```
NResult N_API TiffLoadImageFromMemory(const void * pBuffer, NSizeType bufferSize, HNImage *
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
phImage);
```

Parameters

Parameters	Description
const void * pBuffer	[in] Pointer to memory buffer.
NSizeType bufferSize	[in] Size of memory buffer.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pBuffer* is NULL [1075] and *bufferSize* is not equal to zero. - or - *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file specified by *pBuffer* is invalid.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Tiff Unit [1429]

7.1.5.23.1.3 TiffLoadImageFromStream Function

Loads image from stream containing TIFF file.

C++

```
NResult N_API TiffLoadImageFromStream(HNStream hStream, HNImage * phImage);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to HNStream containing Tiff [1429] image.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].

Module

Tiff Unit [1429]

7.1.5.23.2 Files

7.1.5.23.2.1 Tiff.h

Header file for Tiff [1429] module. Provides functionality for loading images in TIFF format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	TiffLoadImageFromFile [1429]	Loads image from TIFF file.
	TiffLoadImageFromMemory [1429]	Loads image from memory buffer containing TIFF file.
	TiffLoadImageFromStream [1430]	Loads image from stream containing TIFF file.

Module

Tiff Unit [1429]

7.1.5.24 Wsq Unit

Provides functionality for loading and saving images in WSQ format.

Remarks

WSQ images can be saved compressed. Compression is controlled using bit rate which can be calculated using this formula: 11.25/N where N is compression ratio. Typical compression ratios:

Compression ratio	Bit rate
1:15	0.75
1:5	2.25
1:N	11.25/N

WSQ image bit rate is set using WsqInfoSetBitRate [1433] function.

Files

Name	Description
Wsq.h [1438]	Header file for Wsq module. Provides functionality for loading and saving images in WSQ format.

Functions

	Name	Description
	WsqInfoGetBitRate [1432]	Gets WSQ image bit rate (compression level)
	WsqInfoGetImplementationNumber [1432]	Gets WSQ image implementation number.
	WsqInfoGetNistCom [1433]	Retrieves the NIST comment of the specified WSQ info file.
	WsqInfoHasNistCom [1433]	Checks whether the WSQ info file has a NIST comment.
	WsqInfoSetBitRate [1433]	Sets WSQ image bit rate (compression level)
	WsqInfoSetHasNistCom [1434]	Sets whether the WSQ info file has a NIST comment.
	WsqLoadImageFromFile [1434]	Loads image from memory buffer containing WSQ file.
	WsqLoadImageFromMemory [1434]	Loads image from memory buffer containing WSQ file.
	WsqLoadImageFromStream [1435]	Loads image from stream containing WSQ file.
	WsqSaveImageToFile [1435]	Saves image to file in WSQ format with specified bitrate.
	WsqSaveImageToMemory [1436]	Saves image to memory buffer in WSQ format with specified bitrate.
	WsqSaveImageToStream [1437]	Saves image to stream in WSQ format with specified bitrate.

Macros

Name	Description
WSQ_DEFAULT_BIT_RATE [1437]	Specifies default bit rate (compression level).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_DEBIAN_AMD64 [1437]	Represents Neurotechnology Debian AMD64 implementation.
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_DEBIAN_I386 [1437]	Represents Neurotechnology Debian I386 implementation.
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_INTEL [1438]	Represents Neurotechnology MacOS X Intel implementation.
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_INTEL64 [1438]	Represents Neurotechnology MacOS X Intel 64 implementation.
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_POWERPC [1438]	Represents Neurotechnology MacOS X PowerPC implementation.
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_WIN32_X86 [1438]	Represents Neurotechnology Windows 32 x86 implementation.
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_WIN64_X64 [1438]	Represents Neurotechnology Windows 64 x64 implementation.

7.1.5.24.1 Functions

7.1.5.24.1.1 WsqInfoGetBitRate Function

Gets WSQ image bit rate (compression level)

C++

```
NResult N_API WsqInfoGetBitRate(HWsqInfo hInfo, NFloat * pValue);
```

Parameters

Parameters	Description
HWsqInfo hInfo	[in] Handle to WsqInfo.
NFloat * pValue	[out] WSQ image bit rate.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

Wsq Unit [1431]

7.1.5.24.1.2 WsqInfoGetImplementationNumber Function

Gets WSQ image implementation number.

C++

```
NResult N_API WsqInfoGetImplementationNumber(HWsqInfo hInfo, NUShort * pValue);
```

Parameters

Parameters	Description
HWsqInfo hInfo	[in] Handle to WsqInfo.
NUShort * pValue	[out] A pointer to WSQ image implementation number.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Wsq Unit [1431]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.24.1.3 WsqInfoGetNistCom Function

Retrieves the NIST comment of the specified WSQ info file.

C++

```
NResult N_API WsqInfoGetNistCom(HWsqInfo hInfo, HNistCom * phValue);
```

Parameters

Parameters	Description
HWsqInfo hInfo	[in] Handle to the WsqInfo object.
HNistCom * phValue	[out] Pointer to a handle to the NIST comment.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Wsq Unit [1431]

7.1.5.24.1.4 WsqInfoHasNistCom Function

Checks whether the WSQ info file has a NIST comment.

C++

```
NResult N_API WsqInfoHasNistCom(HWsqInfo hInfo, NBool * pValue);
```

Parameters

Parameters	Description
HWsqInfo hInfo	[in] Handle to the WsqInfo object.
NBool * pValue	[out] Pointer to NTrue [1064] value if the WSQ info file contains NIST comment. Otherwise, pointer to NFalse [1041] value.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Wsq Unit [1431]

7.1.5.24.1.5 WsqInfoSetBitRate Function

Sets WSQ image bit rate (compression level)

C++

```
NResult N_API WsqInfoSetBitRate(HWsqInfo hInfo, NFloat value);
```

Parameters

Parameters	Description
HWsqInfo hInfo	[in] Handle to WsqInfo.
NFloat value	[out] WSQ image bit rate to set.

Returns

If the function succeeds, the return value is N_OK [433]. Otherwise error code [17] is returned. NSucceeded [434] or NFailed [434] macros can be used to detect whether the function succeeded or failed.

Module

Wsq Unit [1431]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.24.1.6 WsqInfoSetHasNistCom Function

Sets whether the WSQ info file has a NIST comment.

C++

```
NResult N_API WsqInfoSetHasNistCom(HWsqInfo hInfo, NBool value);
```

Parameters

Parameters	Description
HWsqInfo hInfo	[in] Handle to the WsqInfo object.
NBool value	[in] NTrue [1064] if the WSQ info file contains NIST comment, else NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

Wsq Unit [1431]

7.1.5.24.1.7 WsqLoadImageFromFile Function

Loads image from memory buffer containing WSQ file.

C++

```
NResult N_API WsqLoadImageFromFile(const NChar * szFileName, HNImage * phImage);
```

Parameters

Parameters	Description
const NChar * szFileName	[in] Pointer to a string that specifies filename.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

If the function succeeds the return value is N_OK [433]. Otherwise one of the following error codes is returned:

Return values	Description
N_E_ARGUMENT_NULL [427]	*szFileName* is NULL [1075] - or - *phImage* is NULL [1075].

Module

Wsq Unit [1431]

7.1.5.24.1.8 WsqLoadImageFromMemory Function

Loads image from memory buffer containing WSQ file.

C++

```
NResult N_API WsqLoadImageFromMemory(const void * pBuffer, NSizeType bufferSize, HNImage *
phImage);
```

Parameters

Parameters	Description
const void * pBuffer	[out] Pointer to memory buffer.
NSizeType bufferSize	[in] Size of memory buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *pBuffer* is NULL [1075] and *bufferSize* is not equal to zero. - or - *phImage* is NULL [1075].
N_E_FORMAT [429]	If the function fails, this error can be returned. This error code is returned when format of file contained in buffer specified by *pBuffer* is invalid.

Remarks

This is a low-level function and can be changed in future version of the library.

Module

Wsq Unit [1431]

7.1.5.24.1.9 WsqLoadImageFromStream Function

Loads image from stream containing WSQ file.

C++

```
NResult N_API WsqLoadImageFromStream(HNStream hStream, HNImage * phImage);
```

Parameters

Parameters	Description
HNStream hStream	[in] Handle to HNStream containing WSQ image.
HNImage * phImage	[out] Pointer to HNImage [1276] that receives handle to loaded image.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].

Module

Wsq Unit [1431]

7.1.5.24.1.10 WsqSaveImageToFile Function

Saves image to file in WSQ format with specified bitrate.

C++

```
NResult N_API WsqSaveImageToFile(HNImage hImage, NFloat bitRate, const NChar * szFileName);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
NFloat bitRate	[in] Specifies bit rate (compression level).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

const NChar * szFileName	[in] Points to string that specifies file name.

Returns

If the function succeeds the return value is N_OK [433]. Otherwise one of the following error codes is returned:

Return values	Description
N_E_ARGUMENT_NULL [427]	*hImage* or *szFileName* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	*bitRate* is less than or equal to zero.

Remarks

This is a low-level function and can be changed in future version of the library.

bitRate can be set to WSQ_DEFAULT_BIT_RATE [1437] (0.75) that corresponds to 15:1 compression. The lower the bit rate is, the higher is the compression level and vice versa. Another common bit rate is 2.25 for 5:1 compression.

Module

Wsq Unit [1431]

7.1.5.24.1.11 WsqSaveImageToMemory Function

Saves image to memory buffer in WSQ format with specified bitrate.

C++

```
NResult N_API WsqSaveImageToMemory(HNImage hImage, NFloat bitRate, void * * ppBuffer,
NSizeType * pBufferSize);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
NFloat bitRate	[in] Specifies bit rate (compression level).
void * * ppBuffer	[out] Pointer to void * that receives pointer to allocated memory buffer.
NSizeType * pBufferSize	[out] Pointer to NSizeType [1008] that receives size of allocated memory buffer.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].
N_E_ARGUMENT_NULL [427]	If the function fails, this error can be returned. This error code is returned when *hImage*, *ppBuffer* or *pBufferSize* is NULL [1075].
N_E_ARGUMENT_OUT_OF_RANGE [428]	If the function fails, this error can be returned. This error code is returned when *bitRate* is less than or equal to zero.
N_E_OUT_OF_MEMORY [432]	If the function fails, this error can be returned. This error code is returned when there was not enough memory to allocate memory buffer.

Remarks

This is a low-level function and can be changed in future version of the library.

Memory buffer allocated by the function must be deallocated using NFree [506] function when it is no longer needed.

Module

Wsq Unit [1431]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.24.1.12 WsqSaveImageToStream Function

Saves image to stream in WSQ format with specified bitrate.

C++

```
NResult N_API WsqSaveImageToStream(HNImage hImage, NFloat bitRate, HNStream hStream);
```

Parameters

Parameters	Description
HNImage hImage	[in] Handle to image.
NFloat bitRate	[in] Specifies bit rate (compression level).
HNStream hStream	[out] Handle to stream.

Returns

Return values	Description
N_OK [433]	If the function succeeds, the return value is N_OK [433].

Module

Wsq Unit [1431]

7.1.5.24.2 Macros

7.1.5.24.2.1 WSQ_DEFAULT_BIT_RATE Macro

Specifies default bit rate (compression level).

C++

```
#define WSQ_DEFAULT_BIT_RATE 0.75f
```

Module

Wsq Unit [1431]

7.1.5.24.2.2 WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_DEBIAN_AMD64 Macro

Represents Neurotechnology Debian AMD64 implementation.

C++

```
#define WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_DEBIAN_AMD64 10153
```

Module

Wsq Unit [1431]

7.1.5.24.2.3 WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_DEBIAN_I386 Macro

Represents Neurotechnology Debian I386 implementation.

C++

```
#define WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_DEBIAN_I386 10152
```

Module

Wsq Unit [1431]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.5.24.2.4 WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_INTEL Macro

Represents Neurotechnology MacOS X Intel implementation.

C++

```
#define WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_INTEL 10154
```

Module

Wsq Unit [1431]

7.1.5.24.2.5 WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_INTEL64 Macro

Represents Neurotechnology MacOS X Intel 64 implementation.

C++

```
#define WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_INTEL64 10155
```

Module

Wsq Unit [1431]

7.1.5.24.2.6 WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_POWERPC Macro

Represents Neurotechnology MacOS X PowerPC implementation.

C++

```
#define WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_POWERPC 10156
```

Module

Wsq Unit [1431]

7.1.5.24.2.7 WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_WIN32_X86 Macro

Represents Neurotechnology Windows 32 x86 implementation.

C++

```
#define WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_WIN32_X86 10150
```

Module

Wsq Unit [1431]

7.1.5.24.2.8 WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_WIN64_X64 Macro

Represents Neurotechnology Windows 64 x64 implementation.

C++

```
#define WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_WIN64_X64 10151
```

Module

Wsq Unit [1431]

7.1.5.24.3 Files

7.1.5.24.3.1 Wsq.h

Header file for Wsq [1431] module. Provides functionality for loading and saving images in WSQ format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Functions

	Name	Description
	WsqInfoGetBitRate [1432]	Gets WSQ image bit rate (compression level)
	WsqInfoGetImplementationNumber [1432]	Gets WSQ image implementation number.
	WsqInfoGetNistCom [1433]	Retrieves the NIST comment of the specified WSQ info file.
	WsqInfoHasNistCom [1433]	Checks whether the WSQ info file has a NIST comment.
	WsqInfoSetBitRate [1433]	Sets WSQ image bit rate (compression level)
	WsqInfoSetHasNistCom [1434]	Sets whether the WSQ info file has a NIST comment.
	WsqLoadImageFromFile [1434]	Loads image from memory buffer containing WSQ file.
	WsqLoadImageFromMemory [1434]	Loads image from memory buffer containing WSQ file.
	WsqLoadImageFromStream [1435]	Loads image from stream containing WSQ file.
	WsqSaveImageToFile [1435]	Saves image to file in WSQ format with specified bitrate.
	WsqSaveImageToMemory [1436]	Saves image to memory buffer in WSQ format with specified bitrate.
	WsqSaveImageToStream [1437]	Saves image to stream in WSQ format with specified bitrate.

Macros

Name	Description
WSQ_DEFAULT_BIT_RATE [1437]	Specifies default bit rate (compression level).
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_DEBIAN_AMD64 [1437]	Represents Neurotechnology Debian AMD64 implementation.
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_DEBIAN_I386 [1437]	Represents Neurotechnology Debian I386 implementation.
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_INTEL [1438]	Represents Neurotechnology MacOS X Intel implementation.
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_INTEL64 [1438]	Represents Neurotechnology MacOS X Intel 64 implementation.
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_POWERPC [1438]	Represents Neurotechnology MacOS X PowerPC implementation.
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_WIN32_X86 [1438]	Represents Neurotechnology Windows 32 x86 implementation.
WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_WIN64_X64 [1438]	Represents Neurotechnology Windows 64 x64 implementation.

Module

Wsq Unit [1431]

7.1.6 NSurveillance Library

Provides functionality for real-time faces identification.

Remarks

DLL

- *NSurveillance.dll*

Import library

- NSurveillance.dll.lib

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Shared object (Linux)

- libNSurveillance.so

Modules

Name	Description
NSurveillance Module [1440]	

7.1.6.1 NSurveillance Module

Files

Name	Description
NSurveillanceEventDetails.h [1464]	A header file containing containing EventDetails functions.
NSurveillance.h [1465]	Main NSurveillance [1439] library C header file containing function declaration and event type declarations.
NSurveillanceParams.h [1466]	A header file containing NSurveillance [1439] library parameter macros.

Functions

	Name	Description
	NsedGetFaces [1443]	Gets NleFace [118] from specified event details.
	NsedGetBestMatchesEx [1443]	Gets best face match results (from all frames).
	NsedGetOriginalImage [1444]	Gets original image from specified NSurveillanceEventDetails.
	NsedGetExtractionStatus [1444]	Gets template extraction status form specified NSurveillanceEventDetails.
	NsedGetSubjectTemplate [1445]	Gets generalized Subject template.
	NsedGetTemplate [1445]	Gets extracted face template.
	NsedGetFrameIndex [1445]	Gets frame index (the number of captured images prior to this event).
	NsedGetGrayscaleImage [1446]	Gets grayscale version of originally captured image.
	NsedGetTraceIndex [1446]	Gets an index of the current subject being tracked. The index is zero-based and represents unique id of subject being tracked at the current moment. After subject disappears this index becomes invalid.
	NsedGetSourceIndex [1446]	Gets camera Id from specified event details.
	NsedGetDetectionDetails [1447]	Gets detection details of a face being processed.
	NsedGetTimeStamp [1447]	Gets the date and time of the captured image.
	NSurveillanceAddEventCallbackEx [1447]	Adds callback function to Surveillance framework.
	NSurveillanceAddIsRunningChangedCallback [1448]	Adds user specified callback for managing Surveillance changed events.
	NSurveillanceAddSourceEx [1448]	Adds capturing source (handle of Camera object) to the Surveillance framework. The capturing source must not be capturing or being already in use by other library.
	NSurveillanceAddSourceEx2 [1449]	Adds capturing source (handle of NCamera [1145] object) to the Surveillance framework. The capturing source must not be capturing or being already in use by other library.
	NSurveillanceAddTemplateEx [1449]	Adds serialized NLTemplate [153] to the surveillance framework's internal database.
	NSurveillanceClearSourcesEx [1450]	Removes any added capturing sources from surveillance framework.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NSurveillanceGetInfo [1450]	Fills NLibraryInfo structure with library information.
	NSurveillanceCreate [1451]	Creates NSurveillance [1439] object.
	NSurveillanceGetDeviceManager [1451]	Gets NDeviceManager associated with specified NSurveillance [1439] handle.
	NSurveillanceGetSourceCountEx [1451]	Retrieves the number of capturing sources previously added to the Surveillance framework.
	NSurveillanceGetSourceEx [1452]	Retrieves a handle for previously added capturing source.
	NSurveillanceGetSourceEx2 [1452]	Retrieves a handle for previously added capturing source.
	NSurveillanceGetTemplateIdsEx [1452]	Retrieves an array of stored in the database template Ids. You must call this function twice: first to retrieve the number of elements in the array, and second to retrieve the array itself.
	NSurveillanceIsRunning [1453]	Checks if NSurveillance [1439] object is running.
	NSurveillanceRemoveAllTemplatesEx [1453]	Removed templates from internal Surveillance database.
	NSurveillanceRemoveEventCallbackEx [1454]	Removes callback function from Surveillance framework.
	NSurveillanceRemoveIsRunningChangedCallback [1454]	Removes user specified callback for managing Surveillance changed events.
	NSurveillanceRemoveSourceEx [1455]	Removes capturing source from Surveillance framework.
	NSurveillanceRemoveTemplateEx [1455]	Removes template from Surveillance internal database.
	NSurveillanceStartEx [1455]	Enables callbacks and starts Surveillance. After surveillance is no longer needed NSurveillanceStop should be called.
	NSurveillanceStopEx [1456]	Stops surveillance and disables callbacks.
	NsedGetBestMatch [1456]	Gets best face match results (from all frames).
	NsedGetBestMatchCount [1456]	Gets best face match results (from all frames) count.

Macros

Name	Description
NSP_DETECTION_DETECT_MORE_FACES [1458]	Identifier used to choose different face detection procedures which result in high recall.
NSP_DETECTION_FACE_DETECTION_SCALE_COUNT [1458]	Identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate. Values are in range [1..16].
NSP_DETECTION_FACE_ROLL_ANGLE_BASE [1458]	Identifier specifying a base angle for faces roll angle in the image to search. Values are in range [-180..180]. The default value is 0. Identifier is used to set extractor base angle to support mobile devices with orientation sensor.
NSP_DETECTION_MAX_ROLL_ANGLE_DEVIATION [1459]	Defines maximum roll angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180].
NSP_DETECTION_MAX_YAW_ANGLE_DEVIATION [1459]	Identifier specifying the maximum yaw angle deviation from frontal face in degrees. The value of this parameter can be in range [0..90]. The default value is 15. Currently supports deviations up to 45 degrees.
NSP_EXTRACTION_BLINK_CONFIDENCE_THRESHOLD [1459]	Closed eyes absence/presence confidence threshold.
NSP_EXTRACTION_DARK_GLASSES_CONFIDENCE_THRESHOLD [1459]	Dark glasses on a face absence/presence confidence threshold.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSP_EXTRACTION_DETECT_BLINK [1459]	Identifier used to detect closed eyes.
NSP_EXTRACTION_DETECT_DARK_GLASSES [1460]	Identifier used to detect faces with dark glasses.
NSP_EXTRACTION_DETECT_EXPRESSION [1460]	Identifier used to detect face expressions.
NSP_EXTRACTION_DETECT_GENDER [1460]	Identifier indicating whether to use gender detection procedure.
NSP_EXTRACTION_DETECT_GLASSES [1460]	Identifier used to detect faces with glasses.
NSP_EXTRACTION_DETECT_MOUTH_OPEN [1460]	Identifier used to detect faces with mouth open.
NSP_EXTRACTION_ENROLLMENT_TEMPLATE_SIZE [1461]	Specifies enrollment template size. It is used in generating a template from a tracked subject.
NSP_EXTRACTION_EXPRESSION_CONFIDENCE_THRESHOLD [1461]	Face expressions absence/presence confidence threshold.
NSP_EXTRACTION_GENDER_CONFIDENCE_THRESHOLD [1461]	This threshold checks the gender of the captured face. The higher the value of this parameter, the more strictly the gender of the face is checked. The value of this parameter can be in range [0..100].
NSP_EXTRACTION_GLASSES_CONFIDENCE_THRESHOLD [1461]	Glasses on a face absence/presence confidence threshold.
NSP_EXTRACTION_IDENTIFICATION_TEMPLATE_SIZE [1461]	Specifies identification template size.
NSP_EXTRACTION_MOUTH_OPEN_CONFIDENCE_THRESHOLD [1462]	Face with mouth open absence/presence confidence threshold.
NSP_MAX_THREAD_COUNT [1462]	Used as ParameterId for setting the number of concurrent surveillance threads. Recommended value is "number of logical processors on CPU" plus one. Must be in range [3..8]. The default value is 5.
NSP_IDENTIFICATION_MAX_BEST_MATCH_COUNT [1462]	Maximum length of the list which is returned by each identification. The default value is 10.
NSP_DETECTION_MAX_IOD [1463]	Identifier of type N_TYPE_INT [577] specifying maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 4000.
NSP_DETECTION_MIN_IOD [1463]	Identifier of type N_TYPE_INT [577] specifying minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 40.
NSP_DETECTION_FACE_CONFIDENCE_THRESHOLD [1463]	Identifier of type N_TYPE_DOUBLE [576]. Specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100]. The default value is 52.
NSP_EXTRACTION_FACE_QUALITY_THRESHOLD [1463]	Identifier of type N_TYPE_BYTE [576]. Specifies the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSP_EXTRACTION_MAX_RECORDS_PER_TEMPLATE [1464]	Identifier of type N_TYPE_INT [577]. Sets the maximum number of records an extraction function can return in one NLTemplate [153]. The value of this parameter can be in range [1..16]. The default value is 5.
NSP_IDENTIFICATION_MATCHING_THRESHOLD [1464]	Identifier specifying identification matching threshold (biggest allowed FAR) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%). The default value is 48.

Structs, Records, Enums

	Name	Description
	NSurveillanceEventType_ [1457]	Enumeration type for different Surveillance events. Different information is available inside callback function being called depending on the event type. Events are asynchronous, so no assumption on the order of events fired or callbacks called should be made. It is possible to set several callback functions for a single event. Data available during event should be retrieved using Nsed* functions. The table below explains what data is available during different events.
	NSurveillanceEventType [1457]	Enumeration type for different Surveillance events. Different information is available inside callback function being called depending on the event type. Events are asynchronous, so no assumption on the order of events fired or callbacks called should be made. It is possible to set several callback functions for a single event. Data available during event should be retrieved using Nsed* functions. The table below explains what data is available during different events.

7.1.6.1.1 Functions

7.1.6.1.1.1 NsedGetFaces Function

Gets NleFace [118] from specified event details.

C++

```
NResult N_API NsedGetFaces(HNSurveillanceEventDetails hDetails, NleFace * arFaces);
```

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
NleFace * arFaces	[out] Pointer to array of NleFace [118] containing information about face location in an image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros. If NResult [1007] is greater than 0 this means that faces were found. In this case NResult [1007] value indicates the number of found faces (length of *arFaces* array).

Module

NSurveillance Module [1440]

7.1.6.1.1.2 NsedGetBestMatchesEx Function

Gets best face match results (from all frames).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NsedGetBestMatchesEx(HNSurveillanceEventDetails hDetails, NsedMatch *
arValues, NInt valuesLength);
```

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
NsedMatch * arValues	[out] Pointer to array of NsedMatch containing template Id and score.
valueLength	[in] The size of the given *arValues*.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Notes

arValues saves the score of the best matching result.

Module

NSurveillance Module [1440]

7.1.6.1.1.3 NsedGetOriginalImage Function

Gets original image from specified NSurveillanceEventDetails.

C++

```
NResult N_API NsedGetOriginalImage(HNSurveillanceEventDetails hDetails, HNImage * phValue);
```

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
HNImage * phValue	[out] Pointer to NImage [1232].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSurveillance Module [1440]

7.1.6.1.1.4 NsedGetExtractionStatus Function

Gets template extraction status form specified NSurveillanceEventDetails.

C++

```
NResult N_API NsedGetExtractionStatus(HNSurveillanceEventDetails hDetails,
NleExtractionStatus * pValue);
```

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
NleExtractionStatus * pValue	[out] Face extraction status.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NSurveillance Module [1440]

7.1.6.1.1.5 NsedGetSubjectTemplate Function

Gets generalized Subject template.

C++

```
NResult N_API NsedGetSubjectTemplate(HNSurveillanceEventDetails hDetails, HNLTemplate *
phTemplate);
```

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
HNLTemplate * phTemplate	[out] Face template.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Remarks

Subject template is a compressed generalized template of all extracted faces during subject tracking whose Id matches that of the best matched Id. If there is no best match, then it includes all templates into generalization.

The template is updated every time a new face is determined to belong to the tracked subject.

Notes

Returned handle must be freed in order to avoid memory leaks.

Module

NSurveillance Module [1440]

7.1.6.1.1.6 NsedGetTemplate Function

Gets extracted face template.

C++

```
NResult N_API NsedGetTemplate(HNSurveillanceEventDetails hDetails, HNLTemplate * phValue);
```

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
HNLTemplate * phValue	[out] Face template.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSurveillance Module [1440]

7.1.6.1.1.7 NsedGetFrameIndex Function

Gets frame index (the number of captured images prior to this event).

C++

```
NResult N_API NsedGetFrameIndex(HNSurveillanceEventDetails hDetails, NULong * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
NULong * pValue	[out] Frame index.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSurveillance Module [1440]

7.1.6.1.1.8 NsedGetGrayscaleImage Function

Gets grayscale version of originally captured image.

C++

```
NResult N_API NsedGetGrayscaleImage(HNSurveillanceEventDetails hDetails, HNGrayscaleImage *
phValue);
```

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
HNGrayscaleImage * phValue	[out] Grayscale image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSurveillance Module [1440]

7.1.6.1.1.9 NsedGetTraceIndex Function

Gets an index of the current subject being tracked. The index is zero-based and represents unique id of subject being tracked at the current moment. After subject disappears this index becomes invalid.

C++

```
NResult N_API NsedGetTraceIndex(HNSurveillanceEventDetails hDetails, NInt * pValue);
```

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
NInt * pValue	[out] Index of subject being traced.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSurveillance Module [1440]

7.1.6.1.1.10 NsedGetSourceIndex Function

Gets camera Id from specified event details.

C++

```
NResult N_API NsedGetSourceIndex(HNSurveillanceEventDetails hDetails, NInt * pValue);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
NInt * pValue	[out] Camera Id.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSurveillance Module [1440]

7.1.6.1.1.11 NsedGetDetectionDetails Function

Gets detection details of a face being processed.

C++

```
NResult N_API NsedGetDetectionDetails(HNSurveillanceEventDetails hDetails,
NleDetectionDetails * pValue);
```

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
NleDetectionDetails * pValue	[out] Face detection details.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSurveillance Module [1440]

7.1.6.1.1.12 NsedGetTimeStamp Function

Gets the date and time of the captured image.

C++

```
NResult N_API NsedGetTimeStamp(HNSurveillanceEventDetails hDetails, NDateTime_ * pValue);
```

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
NDateTime_ * pValue	[out] Timestamp of captured image.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSurveillance Module [1440]

7.1.6.1.1.13 NSurveillanceAddEventCallbackEx Function

Adds callback function to Surveillance framework.

C++

```
NResult N_API NSurveillanceAddEventCallbackEx(HNSurveillance hSurveillance,
NSurveillanceEventCallbackEx pCallback, NSurveillanceEventType eventType, void * pParam);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle to NSurveillance [1439] object.
NSurveillanceEventCallbackEx pCallback	[in] Callback function to be added. The type of the callback function is NSurveillanceEventCallbackEx.
NSurveillanceEventType eventType	[in] Event Type for which to add callback function.
void * pParam	[in] Parameter which was supplied to the callback function.

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Module

NSurveillance Module [1440]

7.1.6.1.1.14 NSurveillanceAddIsRunningChangedCallback Function

Adds user specified callback for managing Surveillance changed events.

C++

```
NResult N_API NSurveillanceAddIsRunningChangedCallback(HNSurveillance hSurveillance,
NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle to NSurveillance [1439] object.
NObjectCallback pCallback	[in] Callback function to be added. The type of the callback function is NObjectCallback.
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSurveillance Module [1440]

7.1.6.1.1.15 NSurveillanceAddSourceEx Function

Adds capturing source (handle of Camera object) to the Surveillance framework. The capturing source must not be capturing or being already in use by other library.

C++

```
NResult N_API NSurveillanceAddSourceEx(HNSurveillance hSurveillance, HCamera hCamera);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle of Surveillance object.
HCamera hCamera	[in] Handle of Camera object. The type of the handle must be HCamera [1131].

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Notes

The same capturing source must be added only once.

Surveillance must be stopped prior calling this function.

Module

NSurveillance Module [1440]

7.1.6.1.1.16 NSurveillanceAddSourceEx2 Function

Adds capturing source (handle of NCamera [1145] object) to the Surveillance framework. The capturing source must not be capturing or being already in use by other library.

C++

```
NResult N_API NSurveillanceAddSourceEx2(HNSurveillance hSurveillance, HNCamera hSource,
NInt * pIndex);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle of Surveillance object.
HNCamera hSource	[in] Handle of NCamera [1145] object. The type of the handle must be HCamera [1131].
NInt * pIndex	[out] Source index.

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Notes

The same capturing source must be added only once.

Surveillance must be stopped prior calling this function.

Module

NSurveillance Module [1440]

7.1.6.1.1.17 NSurveillanceAddTemplateEx Function

Adds serialized NLTemplate [153] to the surveillance framework's internal database.

C++

```
NResult N_API NSurveillanceAddTemplateEx(HNSurveillance hSurveillance, const void *
pTemplate, NSizeType templateSize, const void * templateId);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle of Surveillance object.
const void * pTemplate	[in] Pointer to memory location where serialized template is stored.
NSizeType templateSize	[in] The size of memory block containing serialized template.
const void * templateId	[in] ID of the template

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

Serialized template will be copied to the internal database. Only *templateId* is stored internally.

Module

NSurveillance Module [1440]

7.1.6.1.1.18 NSurveillanceClearSourcesEx Function

Removes any added capturing sources from surveillance framework.

C++

```
NResult N_API NSurveillanceClearSourcesEx(HNSurveillance hSurveillance);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle of Surveillance object from which any added capturing sources should be removed.

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Notes

Surveillance must be stopped prior calling this function.

Module

NSurveillance Module [1440]

7.1.6.1.1.19 NSurveillanceGetInfo Function

Fills NLibraryInfo structure with library information.

C++

```
NResult N_API NSurveillanceGetInfo(NLibraryInfo * pValue);
```

Parameters

Parameters	Description
NLibraryInfo * pValue	[in] Pointer to NLibraryInfo structure.

Returns

If the function succeeds, the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_FAILED [429]	Failed to complete operation.

Notes

Library exports two functions NSurveillanceGetInfoA and NSurveillanceGetInfoW for ANSI and Wide-char versions. Provided C header contains macro which selects one of these depending on configuration.

Module

NSurveillance Module [1440]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.6.1.1.20 NSurveillanceCreate Function

Creates NSurveillance [1439] object.

C++

```
NResult N_API NSurveillanceCreate(HNSurveillance * phSurveillance);
```

Parameters

Parameters	Description
pHSurveillance	[out] Pointer to handle of created Surveillance object.

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Module

NSurveillance Module [1440]

7.1.6.1.1.21 NSurveillanceGetDeviceManager Function

Gets NDeviceManager associated with specified NSurveillance [1439] handle.

C++

```
NResult N_API NSurveillanceGetDeviceManager(HNSurveillance hSurveillance, HNDeviceManager *
phValue);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle of NSurveillance [1439] object.
phDeviceManager	[out] Surveillance device manager.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSurveillance Module [1440]

7.1.6.1.1.22 NSurveillanceGetSourceCountEx Function

Retrieves the number of capturing sources previously added to the Surveillance framework.

C++

```
NResult N_API NSurveillanceGetSourceCountEx(HNSurveillance hSurveillance, NInt * pCount);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle of Surveillance object.
NInt * pCount	[out] Pointer to the count of the capturing sources. The type of the variable is NInt [1006].

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Module

NSurveillance Module [1440]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.6.1.1.23 NSurveillanceGetSourceEx Function

Retrieves a handle for previously added capturing source.

C++

```
NResult N_API NSurveillanceGetSourceEx(HNSurveillance hSurveillance, NInt index, HCamera *
phCamera);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle of Surveillance object.
NInt index	[in] Index of capturing source.
pHCamera	[out] Pointer to camera handle. The handle is of type HCamera [1131].

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Notes

If there was no capturing source set or it was removed using NSurveillanceRemoveSourceEx [1455] function, then Camera handle pointed to by pHCamera will be set to NULL [1075].

Module

NSurveillance Module [1440]

7.1.6.1.1.24 NSurveillanceGetSourceEx2 Function

Retrieves a handle for previously added capturing source.

C++

```
NResult N_API NSurveillanceGetSourceEx2(HNSurveillance hSurveillance, NInt index, HNCamera
* phValue);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle of NSurveillance [1439] object.
NInt index	[in] Index of capturing source.
phSource	[out] Pointer to NCamera [1145] handle.

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Notes

If there was no capturing source set or it was removed using NSurveillanceRemoveSourceEx [1455] function, then Camera handle pointed to by pHCamera will be set to NULL [1075].

Module

NSurveillance Module [1440]

7.1.6.1.1.25 NSurveillanceGetTemplateIdsEx Function

Retrieves an array of stored in the database template Ids. You must call this function twice: first to retrieve the number of elements in the array, and second to retrieve the array itself.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NSurveillanceGetTemplateIdsEx(HNSurveillance hSurveillance, const void * *
arTemplateIds);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle of Surveillance object.
const void * * arTemplateIds	[out] Pointer to the previously allocated array of void pointers.

Returns

If the function succeeds and supplied *arTemplateIds* is NULL [1075], the return value is the number of array elements.

If the function succeeds and supplied *arTemplateIds* is not NULL [1075], the return value is N_OK [433].

If the function fails, the return value is one of the following error codes:

Error code	Condition
N_E_FAILED [429]	Failed to complete operation.

Remarks

To retrieve the array, you must call this function twice. First call must be made with *arTemplateIds* set to NULL [1075]. The return value will be either negative error or positive value indicating the size of the array to be filled. Second call must be made with *arTemplateIds* pointing to the newly allocated array. To ensure that the number of entries in the internal database do not change between these two function calls one must never manipulate it in any way. Best is to implement a way of synchronization.

Module

NSurveillance Module [1440]

7.1.6.1.1.26 NSurveillanceIsRunning Function

Checks if NSurveillance [1439] object is running.

C++

```
NResult N_API NSurveillanceIsRunning(HNSurveillance hSurveillance, NBool * pValue);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle to NSurveillance [1439] object to check.
NBool * pValue	[out] Boolean value indicating if NSurveillance [1439] object is running.

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Module

NSurveillance Module [1440]

7.1.6.1.1.27 NSurveillanceRemoveAllTemplatesEx Function

Removed templates from internal Surveillance database.

C++

```
NResult N_API NSurveillanceRemoveAllTemplatesEx(HNSurveillance hSurveillance);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle to Surveillance object.

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Module

NSurveillance Module [1440]

7.1.6.1.1.28 NSurveillanceRemoveEventCallbackEx Function

Removes callback function from Surveillance framework.

C++

```
NResult N_API NSurveillanceRemoveEventCallbackEx(HNSurveillance hSurveillance,
NSurveillanceEventCallbackEx pCallback, NSurveillanceEventType eventType, void * pParam);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle to Surveillance object to remove callback function.
NSurveillanceEventCallbackEx pCallback	[in] Callback function to be removed. The type of the callback function is NSurveillanceEventCallback.
NSurveillanceEventType eventType	[in] Event Type for which to remove callback function.
void * pParam	[in] pParam that was supplied to the callback function.

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Remarks

If supplied *eventType* parameter is *nsetAny* then every event with such callback and *pParam* combination will be removed from callback list for every event type.

Module

NSurveillance Module [1440]

7.1.6.1.1.29 NSurveillanceRemoveIsRunningChangedCallback Function

Removes user specified callback for managing Surveillance changed events.

C++

```
NResult N_API NSurveillanceRemoveIsRunningChangedCallback(HNSurveillance hSurveillance,
NObjectCallback pCallback, void * pParam);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle to NSurveillance [1439] object.
NObjectCallback pCallback	[in] Callback function to remove. The type of the callback function is NObjectCallback.
void * pParam	[in] User data used by a callback function.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NSurveillance Module [1440]

7.1.6.1.1.30 NSurveillanceRemoveSourceEx Function

Removes capturing source from Surveillance framework.

C++

```
NResult N_API NSurveillanceRemoveSourceEx(HNSurveillance hSurveillance, NInt index);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle to Surveillance object.
NInt index	[in] Index of the capturing source to be removed.

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Notes

Surveillance must be stopped prior calling this function.

Module

NSurveillance Module [1440]

7.1.6.1.1.31 NSurveillanceRemoveTemplateEx Function

Removes template from Surveillance internal database.

C++

```
NResult N_API NSurveillanceRemoveTemplateEx(HNSurveillance hSurveillance, const void *
templateId);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle to Surveillance object.
const void * templateId	[in] Id of the template to be removed from internal database.

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Module

NSurveillance Module [1440]

7.1.6.1.1.32 NSurveillanceStartEx Function

Enables callbacks and starts Surveillance. After surveillance is no longer needed NSurveillanceStop should be called.

C++

```
NResult N_API NSurveillanceStartEx(HNSurveillance hSurveillance);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle to Surveillance object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Module

NSurveillance Module [1440]

7.1.6.1.1.33 NSurveillanceStopEx Function

Stops surveillance and disables callbacks.

C++

```
NResult N_API NSurveillanceStopEx(HNSurveillance hSurveillance);
```

Parameters

Parameters	Description
HNSurveillance hSurveillance	[in] Handle to Surveillance object.

Returns

If the function succeeds, the return value is N_OK [433]. Also NSucceeded [434] macro can be used to check if function succeeded.

Remarks

Additional synchronization is needed if any of the set callbacks synchronously calls to the main thread. A deadlock can occur if NSurveilanceStop is called in the same time as callback.

Module

NSurveillance Module [1440]

7.1.6.1.1.34 NsedGetBestMatch Function

Gets best face match results (from all frames).

C++

```
NResult N_API NsedGetBestMatch(HNSurveillanceEventDetails hDetails, NInt index, NsedMatch *
pValue);
```

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
NInt index	[in] Best match index.
NsedMatch * pValue	[out] Pointer to array of NsedMatch containing template Id and score.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSurveillance Module [1440]

7.1.6.1.1.35 NsedGetBestMatchCount Function

Gets best face match results (from all frames) count.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
NResult N_API NsedGetBestMatchCount(HNSurveillanceEventDetails hDetails, NInt * pValue);
```

Parameters

Parameters	Description
HNSurveillanceEventDetails hDetails	[in] Handle to NSurveillanceEventDetails.
NInt * pValue	[out] Best matches count.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

NSurveillance Module [1440]

7.1.6.1.2 Structs, Records, Enums

7.1.6.1.2.1 NSurveillanceEventType Enumeration

Enumeration type for different Surveillance events. Different information is available inside callback function being called depending on the event type. Events are asynchronous, so no assumption on the order of events fired or callbacks called should be made. It is possible to set several callback functions for a single event. Data available during event should be retrieved using Nsed* functions. The table below explains what data is available during different events.

	nsetImageCaptured	nsetFacesDetected	nsetSubjectAppeared	nsetSubjectTrack	nsetSubjectDisappeared
NsedGetOriginalImage [1444]	+	+	+	+	
NsedGetGrayscaleImage [1446]	+	+	+	+	
NsedGetFrameIndex [1445]	+	+	+	+	
NsedGetTimeStamp [1447]	+	+	+	+	
NsedGetFaces [1443]		+			
NsedGetDetectionDetails [1447]			+*	+*	
NsedGetTemplate [1445]			+*	+*	
NsedGetTraceIndex [1446]			+	+	+
NsedGetSubjectTemplate [1445]			+	+	+
NsedGetBestMatchCount [1456]			+	+	+
NsedGetBestMatch [1456]			+	+	+
NsedGetBestMatchesEx [1443]			+	+	+

Copyright © 2009-2014 Neurotechnology. All rights reserved.

* - This data is available for the latest processing image that the event corresponds to.

C++

```
typedef enum NSurveillanceEventType_ {
  nsetAny = -1,
  nsetImageCaptured = 1,
  nsetFacesDetected = 2,
  nsetSubjectAppeared = 3,
  nsetSubjectTrack = 4,
  nsetSubjectDisappeared = 5
} NSurveillanceEventType;
```

Members

Members	Description
nsetAny = -1	Indicates that any event can be captured by supplied callback function.
nsetImageCaptured = 1	Indicates that a new image was captured from capturing source.
nsetFacesDetected = 2	Indicates that at least one face was successfully detected in the captured image.
nsetSubjectAppeared = 3	Indicates that a new subject appeared in the field of view of the capturing device.
nsetSubjectTrack = 4	Indicates that previously appeared subject is being tracked logging it's activity.
nsetSubjectDisappeared = 5	Indicates that a previously tracked subject disappeared from the field of view of the capturing device.

Module

NSurveillance Module [1440]

7.1.6.1.3 Macros

7.1.6.1.3.1 NSP_DETECTION_DETECT_MORE_FACES Macro

Identifier used to choose different face detection procedures which result in high recall.

C++

```
#define NSP_DETECTION_DETECT_MORE_FACES 10107
```

Module

NSurveillance Module [1440]

7.1.6.1.3.2 NSP_DETECTION_FACE_DETECTION_SCALE_COUNT Macro

Identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate. Values are in range [1..16].

C++

```
#define NSP_DETECTION_FACE_DETECTION_SCALE_COUNT 10108
```

Module

NSurveillance Module [1440]

7.1.6.1.3.3 NSP_DETECTION_FACE_ROLL_ANGLE_BASE Macro

Identifier specifying a base angle for faces roll angle in the image to search. Values are in range [-180..180]. The default value is 0. Identifier is used to set extractor base angle to support mobile devices with orientation sensor.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NSP_DETECTION_FACE_ROLL_ANGLE_BASE 10109
```

Module

NSurveillance Module [1440]

7.1.6.1.3.4 NSP_DETECTION_MAX_ROLL_ANGLE_DEVIATION Macro

Defines maximum roll angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180].

C++

```
#define NSP_DETECTION_MAX_ROLL_ANGLE_DEVIATION 10105
```

Remarks

Matching speed highly depends on a roll angle deviation from frontal face.

Module

NSurveillance Module [1440]

7.1.6.1.3.5 NSP_DETECTION_MAX_YAW_ANGLE_DEVIATION Macro

Identifier specifying the maximum yaw angle deviation from frontal face in degrees. The value of this parameter can be in range [0..90]. The default value is 15. Currently supports deviations up to 45 degrees.

C++

```
#define NSP_DETECTION_MAX_YAW_ANGLE_DEVIATION 10106
```

Module

NSurveillance Module [1440]

7.1.6.1.3.6 NSP_EXTRACTION_BLINK_CONFIDENCE_THRESHOLD Macro

Closed eyes absence/presence confidence threshold.

C++

```
#define NSP_EXTRACTION_BLINK_CONFIDENCE_THRESHOLD 10215
```

Module

NSurveillance Module [1440]

7.1.6.1.3.7 NSP_EXTRACTION_DARK_GLASSES_CONFIDENCE_THRESHOLD Macro

Dark glasses on a face absence/presence confidence threshold.

C++

```
#define NSP_EXTRACTION_DARK_GLASSES_CONFIDENCE_THRESHOLD 10221
```

Module

NSurveillance Module [1440]

7.1.6.1.3.8 NSP_EXTRACTION_DETECT_BLINK Macro

Identifier used to detect closed eyes.

C++

```
#define NSP_EXTRACTION_DETECT_BLINK 10214
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NSurveillance Module [1440]

7.1.6.1.3.9 NSP_EXTRACTION_DETECT_DARK_GLASSES Macro

Identifier used to detect faces with dark glasses.

C++

```
#define NSP_EXTRACTION_DETECT_DARK_GLASSES 10220
```

Module

NSurveillance Module [1440]

7.1.6.1.3.10 NSP_EXTRACTION_DETECT_EXPRESSION Macro

Identifier used to detect face expressions.

C++

```
#define NSP_EXTRACTION_DETECT_EXPRESSION 10212
```

Module

NSurveillance Module [1440]

7.1.6.1.3.11 NSP_EXTRACTION_DETECT_GENDER Macro

Identifier indicating whether to use gender detection procedure.

C++

```
#define NSP_EXTRACTION_DETECT_GENDER 10210
```

Remarks

By enabling this parameter (setting to NTrue [1064]) you can ask to estimate person's gender from the face during facial features template extraction. Gender will be estimated correct with higher confidence if the provided face image passes all face image quality checks and facial features template is created. Additionally, gender estimator works more accurate on near-frontal face images.

Module

NSurveillance Module [1440]

7.1.6.1.3.12 NSP_EXTRACTION_DETECT_GLASSES Macro

Identifier used to detect faces with glasses.

C++

```
#define NSP_EXTRACTION_DETECT_GLASSES 10218
```

Module

NSurveillance Module [1440]

7.1.6.1.3.13 NSP_EXTRACTION_DETECT_MOUTH_OPEN Macro

Identifier used to detect faces with mouth open.

C++

```
#define NSP_EXTRACTION_DETECT_MOUTH_OPEN 10216
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

NSurveillance Module [1440]

7.1.6.1.3.14 NSP_EXTRACTION_ENROLLMENT_TEMPLATE_SIZE Macro

Specifies enrollment template size. It is used in generating a template from a tracked subject.

C++

```
#define NSP_EXTRACTION_ENROLLMENT_TEMPLATE_SIZE 10204
```

Remarks

Parameter value can be one of the NleTemplateSize [115] enumeration members. The default value is nletsMedium.

Enrollment template size cannot be larger than identification template size. If it is large, subjects will not be enrolled.

Parameter cannot be changed while surveillance is running (cannot be changed between NSurveillanceStartEx [1455] and NSurveillanceStopEx [1456] calls).

Module

NSurveillance Module [1440]

7.1.6.1.3.15 NSP_EXTRACTION_EXPRESSION_CONFIDENCE_THRESHOLD Macro

Face expressions absence/presence confidence threshold.

C++

```
#define NSP_EXTRACTION_EXPRESSION_CONFIDENCE_THRESHOLD 10213
```

Module

NSurveillance Module [1440]

7.1.6.1.3.16 NSP_EXTRACTION_GENDER_CONFIDENCE_THRESHOLD Macro

This threshold checks the gender of the captured face. The higher the value of this parameter, the more strictly the gender of the face is checked. The value of this parameter can be in range [0..100].

C++

```
#define NSP_EXTRACTION_GENDER_CONFIDENCE_THRESHOLD 10211
```

Module

NSurveillance Module [1440]

7.1.6.1.3.17 NSP_EXTRACTION_GLASSES_CONFIDENCE_THRESHOLD Macro

Glasses on a face absence/presence confidence threshold.

C++

```
#define NSP_EXTRACTION_GLASSES_CONFIDENCE_THRESHOLD 10219
```

Module

NSurveillance Module [1440]

7.1.6.1.3.18 NSP_EXTRACTION_IDENTIFICATION_TEMPLATE_SIZE Macro

Specifies identification template size.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C++

```
#define NSP_EXTRACTION_IDENTIFICATION_TEMPLATE_SIZE 10205
```

Remarks

It is used in generating a template, which is further used in identification, from each detected face. Parameter value can be one of the NleTemplateSize [115] enumeration members. The default value is nletsMedium.

Identification template size cannot be smaller than enrollment template size.

Parameter cannot be changed while surveillance is running (cannot be changed between NSurveillanceStartEx [1455] and NSurveillanceStopEx [1456] calls).

Module

NSurveillance Module [1440]

7.1.6.1.3.19 NSP_EXTRACTION_MOUTH_OPEN_CONFIDENCE_THRESHOLD Macro

Face with mouth open absence/presence confidence threshold.

C++

```
#define NSP_EXTRACTION_MOUTH_OPEN_CONFIDENCE_THRESHOLD 10217
```

Module

NSurveillance Module [1440]

7.1.6.1.3.20 NSP_MAX_THREAD_COUNT Macro

Used as ParameterId for setting the number of concurrent surveillance threads. Recommended value is "number of logical processors on CPU" plus one. Must be in range [3..8]. The default value is 5.

C++

```
#define NSP_MAX_THREAD_COUNT 10001
```

Notes

Parameter cannot be changed while surveillance is running. Change it before NSurveillanceStart or after NSurveillanceStop function is called.

Minimal thread count should be N*3 + 2 where N is running cameras count. If NSP_MAX_THREAD_COUNT is set less than this value an error should appear.

Module

NSurveillance Module [1440]

7.1.6.1.3.21 NSP_IDENTIFICATION_MAX_BEST_MATCH_COUNT Macro

Maximum length of the list which is returned by each identification. The default value is 10.

C++

```
#define NSP_IDENTIFICATION_MAX_BEST_MATCH_COUNT 10303
```

Notes

Parameter cannot be changed while surveillance is running. Change it before NSurveillanceStart or after NSurveillanceStop function is called.

Module

NSurveillance Module [1440]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.1.6.1.3.22 NSP_DETECTION_MAX_IOD Macro

Identifier of type N_TYPE_INT [577] specifying maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 4000.

C++

```
#define NSP_DETECTION_MAX_IOD 10103
```

Notes

Parameter cannot be changed while surveillance is running. Change it before NSurveillanceStart or after NSurveillanceStop function is called.

Module

NSurveillance Module [1440]

7.1.6.1.3.23 NSP_DETECTION_MIN_IOD Macro

Identifier of type N_TYPE_INT [577] specifying minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 40.

C++

```
#define NSP_DETECTION_MIN_IOD 10102
```

Notes

Parameter cannot be changed while surveillance is running. Change it before NSurveillanceStart or after NSurveillanceStop function is called.

Module

NSurveillance Module [1440]

7.1.6.1.3.24 NSP_DETECTION_FACE_CONFIDENCE_THRESHOLD Macro

Identifier of type N_TYPE_DOUBLE [576]. Specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100]. The default value is 52.

C++

```
#define NSP_DETECTION_FACE_CONFIDENCE_THRESHOLD 10104
```

Notes

Parameter cannot be changed while surveillance is running. Change it before NSurveillanceStart or after NSurveillanceStop function is called.

Module

NSurveillance Module [1440]

7.1.6.1.3.25 NSP_EXTRACTION_FACE_QUALITY_THRESHOLD Macro

Identifier of type N_TYPE_BYTE [576]. Specifies the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.

C++

```
#define NSP_EXTRACTION_FACE_QUALITY_THRESHOLD 10202
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Notes

Parameter cannot be changed while surveillance is running. Change it before NSurveillanceStart or after NSurveillanceStop function is called.

Module

NSurveillance Module [1440]

7.1.6.1.3.26 NSP_EXTRACTION_MAX_RECORDS_PER_TEMPLATE Macro

Identifier of type N_TYPE_INT [577]. Sets the maximum number of records an extraction function can return in one NLTemplate [153]. The value of this parameter can be in range [1..16]. The default value is 5.

C++

```
#define NSP_EXTRACTION_MAX_RECORDS_PER_TEMPLATE 10203
```

Notes

Parameter cannot be changed while surveillance is running. Change it before NSurveillanceStart or after NSurveillanceStop function is called.

Module

NSurveillance Module [1440]

7.1.6.1.3.27 NSP_IDENTIFICATION_MATCHING_THRESHOLD Macro

Identifier specifying identification matching threshold (biggest allowed FAR) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%). The default value is 48.

C++

```
#define NSP_IDENTIFICATION_MATCHING_THRESHOLD 10302
```

Notes

Parameter cannot be changed while surveillance is running. Change it before NSurveillanceStart or after NSurveillanceStop function is called.

Module

NSurveillance Module [1440]

7.1.6.1.4 Files

7.1.6.1.4.1 NSurveillanceEventDetails.h

A header file containing containing EventDetails functions.

Functions

	Name	Description
	NsedGetBestMatch [1456]	Gets best face match results (from all frames).
	NsedGetBestMatchCount [1456]	Gets best face match results (from all frames) count.
	NsedGetBestMatchesEx [1443]	Gets best face match results (from all frames).
	NsedGetDetectionDetails [1447]	Gets detection details of a face being processed.
	NsedGetExtractionStatus [1444]	Gets template extraction status form specified NSurveillanceEventDetails.
	NsedGetFaces [1443]	Gets NleFace [118] from specified event details.
	NsedGetFrameIndex [1445]	Gets frame index (the number of captured images prior to this event).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NsedGetGrayscaleImage [1446]	Gets grayscale version of originally captured image.
	NsedGetOriginalImage [1444]	Gets original image from specified NSurveillanceEventDetails.
	NsedGetSourceIndex [1446]	Gets camera Id from specified event details.
	NsedGetSubjectTemplate [1445]	Gets generalized Subject template.
	NsedGetTemplate [1445]	Gets extracted face template.
	NsedGetTimeStamp [1447]	Gets the date and time of the captured image.
	NsedGetTraceIndex [1446]	Gets an index of the current subject being tracked. The index is zero-based and represents unique id of subject being tracked at the current moment. After subject disappears this index becomes invalid.

Module

NSurveillance Module [1440]

7.1.6.1.4.2 NSurveillance.h

Main NSurveillance [1439] library C header file containing function declaration and event type declarations.

Enumerations

	Name	Description
	NSurveillanceEventType_ [1457]	Enumeration type for different Surveillance events. Different information is available inside callback function being called depending on the event type. Events are asynchronous, so no assumption on the order of events fired or callbacks called should be made. It is possible to set several callback functions for a single event. Data available during event should be retrieved using Nsed* functions. The table below explains what data is available during different events.
	NSurveillanceEventType [1457]	Enumeration type for different Surveillance events. Different information is available inside callback function being called depending on the event type. Events are asynchronous, so no assumption on the order of events fired or callbacks called should be made. It is possible to set several callback functions for a single event. Data available during event should be retrieved using Nsed* functions. The table below explains what data is available during different events.

Functions

	Name	Description
	NSurveillanceAddEventCallbackEx [1447]	Adds callback function to Surveillance framework.
	NSurveillanceAddIsRunningChangedCallback [1448]	Adds user specified callback for managing Surveillance changed events.
	NSurveillanceAddSourceEx [1448]	Adds capturing source (handle of Camera object) to the Surveillance framework. The capturing source must not be capturing or being already in use by other library.
	NSurveillanceAddSourceEx2 [1449]	Adds capturing source (handle of NCamera [1145] object) to the Surveillance framework. The capturing source must not be capturing or being already in use by other library.
	NSurveillanceAddTemplateEx [1449]	Adds serialized NLTemplate [153] to the surveillance framework's internal database.
	NSurveillanceClearSourcesEx [1450]	Removes any added capturing sources from surveillance framework.
	NSurveillanceCreate [1451]	Creates NSurveillance [1439] object.
	NSurveillanceGetDeviceManager [1451]	Gets NDeviceManager associated with specified NSurveillance [1439] handle.
	NSurveillanceGetInfo [1450]	Fills NLibraryInfo structure with library information.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NSurveillanceGetSourceCountEx [1451]	Retrieves the number of capturing sources previously added to the Surveillance framework.
	NSurveillanceGetSourceEx [1452]	Retrieves a handle for previously added capturing source.
	NSurveillanceGetSourceEx2 [1452]	Retrieves a handle for previously added capturing source.
	NSurveillanceGetTemplateIdsEx [1452]	Retrieves an array of stored in the database template Ids. You must call this function twice: first to retrieve the number of elements in the array, and second to retrieve the array itself.
	NSurveillanceIsRunning [1453]	Checks if NSurveillance [1439] object is running.
	NSurveillanceRemoveAllTemplatesEx [1453]	Removed templates from internal Surveillance database.
	NSurveillanceRemoveEventCallbackEx [1454]	Removes callback function from Surveillance framework.
	NSurveillanceRemoveIsRunningChangedCallback [1454]	Removes user specified callback for managing Surveillance changed events.
	NSurveillanceRemoveSourceEx [1455]	Removes capturing source from Surveillance framework.
	NSurveillanceRemoveTemplateEx [1455]	Removes template from Surveillance internal database.
	NSurveillanceStartEx [1455]	Enables callbacks and starts Surveillance. After surveillance is no longer needed NSurveillanceStop should be called.
	NSurveillanceStopEx [1456]	Stops surveillance and disables callbacks.

Module

NSurveillance Module [1440]

7.1.6.1.4.3 NSurveillanceParams.h

A header file containing NSurveillance [1439] library parameter macros.

Macros

Name	Description
NSP_DETECTION_DETECT_MORE_FACES [1458]	Identifier used to choose different face detection procedures which result in high recall.
NSP_DETECTION_FACE_CONFIDENCE_THRESHOLD [1463]	Identifier of type N_TYPE_DOUBLE [576]. Specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100]. The default value is 52.
NSP_DETECTION_FACE_DETECTION_SCALE_COUNT [1458]	Identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate. Values are in range [1..16].
NSP_DETECTION_FACE_ROLL_ANGLE_BASE [1458]	Identifier specifying a base angle for faces roll angle in the image to search. Values are in range [-180..180]. The default value is 0. Identifier is used to set extractor base angle to support mobile devices with orientation sensor.
NSP_DETECTION_MAX_IOD [1463]	Identifier of type N_TYPE_INT [577] specifying maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 4000.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSP_DETECTION_MAX_ROLL_ANGLE_DEVIATION [1459]	Defines maximum roll angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180].
NSP_DETECTION_MAX_YAW_ANGLE_DEVIATION [1459]	Identifier specifying the maximum yaw angle deviation from frontal face in degrees. The value of this parameter can be in range [0..90]. The default value is 15. Currently supports deviations up to 45 degrees.
NSP_DETECTION_MIN_IOD [1463]	Identifier of type N_TYPE_INT [577] specifying minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 40.
NSP_EXTRACTION_BLINK_CONFIDENCE_THRESHOLD [1459]	Closed eyes absence/presence confidence threshold.
NSP_EXTRACTION_DARK_GLASSES_CONFIDENCE_THRESHOLD [1459]	Dark glasses on a face absence/presence confidence threshold.
NSP_EXTRACTION_DETECT_BLINK [1459]	Identifier used to detect closed eyes.
NSP_EXTRACTION_DETECT_DARK_GLASSES [1460]	Identifier used to detect faces with dark glasses.
NSP_EXTRACTION_DETECT_EXPRESSION [1460]	Identifier used to detect face expressions.
NSP_EXTRACTION_DETECT_GENDER [1460]	Identifier indicating whether to use gender detection procedure.
NSP_EXTRACTION_DETECT_GLASSES [1460]	Identifier used to detect faces with glasses.
NSP_EXTRACTION_DETECT_MOUTH_OPEN [1460]	Identifier used to detect faces with mouth open.
NSP_EXTRACTION_ENROLLMENT_TEMPLATE_SIZE [1461]	Specifies enrollment template size. It is used in generating a template from a tracked subject.
NSP_EXTRACTION_EXPRESSION_CONFIDENCE_THRESHOLD [1461]	Face expressions absence/presence confidence threshold.
NSP_EXTRACTION_FACE_QUALITY_THRESHOLD [1463]	Identifier of type N_TYPE_BYTE [576]. Specifies the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.
NSP_EXTRACTION_GENDER_CONFIDENCE_THRESHOLD [1461]	This threshold checks the gender of the captured face. The higher the value of this parameter, the more strictly the gender of the face is checked. The value of this parameter can be in range [0..100].
NSP_EXTRACTION_GLASSES_CONFIDENCE_THRESHOLD [1461]	Glasses on a face absence/presence confidence threshold.
NSP_EXTRACTION_IDENTIFICATION_TEMPLATE_SIZE [1461]	Specifies identification template size.
NSP_EXTRACTION_MAX_RECORDS_PER_TEMPLATE [1464]	Identifier of type N_TYPE_INT [577]. Sets the maximum number of records an extraction function can return in one NLTemplate [153]. The value of this parameter can be in range [1..16]. The default value is 5.
NSP_EXTRACTION_MOUTH_OPEN_CONFIDENCE_THRESHOLD [1462]	Face with mouth open absence/presence confidence threshold.
NSP_IDENTIFICATION_MATCHING_THRESHOLD [1464]	Identifier specifying identification matching threshold (biggest allowed FAR) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%). The default value is 48.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSP_IDENTIFICATION_MAX_BEST_MATCH_COUNT [1462]	Maximum length of the list which is returned by each identification. The default value is 10.
NSP_MAX_THREAD_COUNT [1462]	Used as ParameterId for setting the number of concurrent surveillance threads. Recommended value is "number of logical processors on CPU" plus one. Must be in range [3..8]. The default value is 5.

Module

NSurveillance Module [1440]

7.2 .NET Reference

In this section *VeriLook Surveillance 2.3 SDK* .NET programming reference for API is provided.

The required libraries for namespaces are listed in the table bellow:

Namespace	Managed Dll (for .NET)	Unmanaged Dll
Neurotec	Neurotec.dll [1469]	NCore.dll [257]
Neurotec.Biometrics [1654]	Neurotec.Biometrics.dll	NBiometrics.dll
Neurotec.Devices [1792]	Neurotec.Devices.dll	NDevices.dll
Neurotec.Media	Neurotec.Media.dll	NMedia.dll
Neurotec.Licensing [1851]	Neurotec.Licensing.dll [1851]	NLicensing.dll
Neurotec.Surveillance [1883]	Neurotec.Surveillance.dll [1883]	NSurveillance.dll

Functions

	Name	Description
	NLicensingServiceGetBinPath [1909]	Gets the binary path (path to `pg.exe/pgd`) to the license.
	NLicensingServiceGetConfPath [1909]	Gets path to licensing service configuration file (`pgd.conf`).
	NLicensingServiceGetStatus [1910]	Gets the status of the licensing service.
	NLicensingServiceInstall [1910]	Installs licensing service.
	NLicensingServiceIsTrial [1910]	Checks whether the licensing service is trial.
	NLicensingServiceStart [1911]	Starts licensing service.
	NLicensingServiceStop [1911]	Stops licensing service.
	NLicensingServiceUninstall [1911]	Uninstalls licensing service.

Macros

Name	Description
NLicensingServiceInstall [1912]	Picks a Unicode (W) or ANSCII (A) version of the NLicensingServiceInstall function.

Namespaces

Name	Description
Neurotec [1469]	Contains classes that provide infrastructure for Neurotechnology components.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Neurotec.Biometrics [1654]	Class under this namespace provides methods for working with biometrical data.
Neurotec.Biometrics.Gui [1782]	Contains classes that provide infrastructure for Neurotechnology Biometrics [1654] components.
Neurotec.Devices [1792]	Provides functionality for managing devices, like cameras, fingerprint or irises scanners.
Neurotec.IO [1825]	Classes under this namespace provides infrastructure for Neurotechnology components.
Neurotec.Licensing [1851]	Provides functionality for getting, releasing licenses.
Neurotec.Plugins [1863]	Classes under this namespace provides plugins management framework for use in other Neurotechnology components.
Neurotec.Surveillance [1883]	Provides functionality for real-time faces identification.

Structs, Records, Enums

	Name	Description
	NLicensingServiceStatus_ [1911]	Enumerates licensing service status values.
	NLicensingServiceStatus [1911]	Enumerates licensing service status values.

7.2.1 Neurotec Namespace

Contains classes that provide infrastructure for Neurotechnology components.

Classes

	Name	Description
	ErrorSuppressedEventArgs [1470]	Class Description.
	NArray [1471]	Provides functionality for working with arrays.
	NComplex [1519]	Represents a complex number.
S	NCore [1521]	This class supports internal Neurotechnology libraries infrastructure and should not be used directly in your code.
A	NDisposable [1536]	Provides a method to release allocated resources.
	NExpandableObject [1537]	Provides functionality for retrieving information about specified object.
	NIndexPair [1539]	Represents pair of indexes.
A	NInfo [1541]	For internal use only.
A	NInfoObject [1541]	For internal use only.
	NLibraryInfo [1541]	Provides methods used to retrieve library information.
	NModule [1543]	Performs reflection on a module.
	NObject [1549]	Provides functionality for retrieving information about specified object.
S	NProcessorInfo [1572]	Provides functionality for getting processor information.
	NPropertyBag [1576]	Represents a property bag in which the object can persistently save its properties.
	NRange [1585]	Represents range.
	NRational [1587]	Represents a signed rational number.
	NType [1588]	Manages object types.
S	NTypes [1601]	Manages object types.
	NURational [1608]	Represents an unsigned rational number.
	NValue [1609]	Provides functionality for management and conversion of values of different types.
	NVersion [1640]	Manages version information.
	NVersionRange [1645]	Manages version ranges.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Module

.NET Reference [1468]

Structs, Records, Enums

	Name	Description
	NMemoryType [1651]	Enumerates memory type options.
	NModuleOptions [1651]	Enumerates module options.
	NProcessorVendor [1652]	Specifies the processor's vendor name.
	NTypeCode [1652]	Enumerates type codes.

Types

Name	Description
NLibraryGetInfo [1654]	This delegate supports internal Neurotechnology libraries infrastructure and should not be used directly in your code.

7.2.1.1 Classes

7.2.1.1.1 ErrorSuppressedEventArgs Class

Class Description.

C#

```
public class ErrorSuppressedEventArgs : EventArgs;
```

Methods

	Name	Description
	ErrorSuppressedEventArgs [1470]	Initializes a new instance of ErrorSuppressedEventArgs class.

ErrorSuppressedEventArgs Properties

	Name	Description
	Error [1470]	Gets the suppressed error.

7.2.1.1.1.1 ErrorSuppressedEventArgs.ErrorSuppressedEventArgs Constructor

Initializes a new instance of ErrorSuppressedEventArgs class.

C#

```
public ErrorSuppressedEventArgs(Exception error);
```

Parameters

Parameters	Description
Exception error	The supressed error.

7.2.1.1.1.2 ErrorSuppressedEventArgs Properties

7.2.1.1.1.2.1 ErrorSuppressedEventArgs.Error Property

Gets the suppressed error.

C#

```
public Exception Error;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

The suppressed error.

7.2.1.1.2 NArray Class

Provides functionality for working with arrays.

C#

```
public sealed class NArray : NValue, IEnumerable;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

Methods

NValue Class

	Name	Description
	NValue [1613]	Initializes a new instance of the NValue class.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	FromObject [1554]	Creates a new NObject [1549] from the specified object.
	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NValue Class

	Name	Description
	ChangeType [1619]	Changes the type of the specified value to the new type.
	FromObject [1619]	Creates a new value from the specified object.
	FromPointer [1621]	Creates a new value from the specified pointer.
	FromResult [1621]	Creates a new value from the specified result.
	FromString [1622]	Creates a new value from the specified string.
	FromType [1622]	Creates a new value from the specified type.
	ToBoolean [1623]	Converts the specified value into a Boolean value.
	ToByte [1624]	Converts the specified value into a byte.
	ToChar [1625]	Converts the specified value into a char.
	ToDouble [1625]	Converts the specified value into a double number.
	ToInt16 [1626]	Converts the specified value into a short integer.
	ToInt32 [1627]	Converts the specified value into an integer.
	ToInt64 [1627]	Converts the specified value into a long integer.
	ToIntPtr [1628]	Converts the specified value into a pointer.
	ToObject [1629]	Converts the specified value into an object.
	ToPointer [1630]	Converts the specified value into a pointer.
	ToResult [1631]	Converts the specified value into a result.
	ToSByte [1632]	Converts the specified value into a signed byte.
	ToSingle [1633]	Converts the specified value into a floating point value.
	ToString [1633]	Converts the specified value into a string.
	ToType [1634]	Converts the specified value into a value of specified type.
	ToUInt16 [1636]	Converts the specified value into an unsigned short integer.
	ToUInt32 [1636]	Converts the specified value into an unsigned integer.
	ToUInt64 [1637]	Converts the specified value into an unsigned long integer.
	ToUIntPtr [1638]	Converts the specified value into a pointer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NArray Class

	Name	Description
S	FromArray [1475]	Creates a new array from the specified array.
S	FromPointerArray [1485]	Creates a new array from the specified integer pointer value array.
S	FromResultArray [1486]	Creates a new array from the specified result array.
	GetEnumerator [1486]	Retrieves an enumerator for the current array.
	GetValueAs [1486]	Retrieves the of the specified value of the specified type in the array.
	GetValueAsBoolean [1488]	Retrieves the specified Boolean value in the array.
	GetValueAsByte [1489]	Retrieves the specified byte value in the array.
	GetValueAsChar [1490]	Retrieves the specified char value in the array.
	GetValueAsDouble [1491]	Retrieves the specified double value in the array.
	GetValueAsInt16 [1492]	Retrieves the specified short integer value in the array.
	GetValueAsInt32 [1492]	Retrieves the specified integer value in the array.
	GetValueAsInt64 [1493]	Retrieves the specified long integer value in the array.
	GetValueAsIntPtr [1494]	Retrieves the specified integer pointer value in the array.
	GetValueAsObject [1495]	Retrieves the specified object in the array.
	GetValueAsPointer [1496]	Retrieves the specified integer pointer value in the array.
	GetValueAsResult [1497]	Retrieves the specified result value in the array.
	GetValueAsSByte [1498]	Retrieves the specified signed byte value in the array.
	GetValueAsSingle [1498]	Retrieves the specified floating point value in the array.
	GetValueAsString [1499]	Retrieves the specified string value in the array.
	GetValueAsUInt16 [1500]	Retrieves the specified unsigned short integer value in the array.
	GetValueAsUInt32 [1501]	Retrieves the specified unsigned integer value in the array.
	GetValueAsUInt64 [1502]	Retrieves the specified unsigned long integer value in the array.
	GetValueAsUIntPtr [1503]	Retrieves the specified unsigned integer pointer value in the array.
	ToArray [1504]	Copies the current array into another array of the specified type.
	ToBooleanArray [1506]	Copies the current array into a Boolean value array.
	ToByteArray [1506]	Copies the current array into a byte array.
	ToCharArray [1507]	Copies the current array into a char array.
	ToDoubleArray [1508]	Copies the current array into a double value array.
	ToInt16Array [1509]	Copies the current array into a short integer array.
	ToInt32Array [1509]	Copies the current array into an integer array.
	ToInt64Array [1510]	Copies the current array into a long integer array.
	ToIntPtrArray [1511]	Copies the current array into an integer pointer array.
	ToObjectArray [1512]	Copies the current array into an object array.
	ToPointerArray [1512]	Copies the current array into a pointer array.
	ToResultArray [1513]	Copies the current array into a result array.
	ToSByteArray [1514]	Copies the current array into a signed byte array.
	ToSingleArray [1515]	Copies the current array into a floating point value array.
	ToStringArray [1515]	Copies the current array into a string array.
	ToUInt16Array [1516]	Copies the current array into an unsigned short integer array.
	ToUInt32Array [1517]	Copies the current array into an unsigned integer array.
	ToUInt64Array [1518]	Copies the current unsigned long integer array into another array.
	ToUIntPtrArray [1518]	Copies the current array into an unsigned integer pointer array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NValue Class

	Name	Description
	Attributes [1639]	Gets the value attributes.
	IsWrapper [1639]	Checks whether the value is a wrapper.
S	KeyValuePairNativeType [1639]	Defines the native type of the key value pair structure.
	LongSize [1639]	Gets the size of the value (64-bit).
S	NameValuePairNativeType [1639]	Defines the native type of the name value pair structure.
	NativeSize [1639]	Gets the native size of the value.
S	NativeType [1639]	Defines the native type of the object.
	Ptr [1639]	Gets a poitner to the value.
	Size [1639]	Gets the size of the value.
	TypeCode [1640]	Gets the type code of the value.
	ValueType [1640]	Gets the type of the value.

NArray Class

	Name	Description
	Length [1519]	Retrieves the length of the array.
S	NativeType [1519]	Defines the native type of the object.

7.2.1.1.2.1 NArray Methods

7.2.1.1.2.1.1 FromArray Method

7.2.1.1.2.1.1.1 NArray.FromArray Method (Array)

Creates a new array from the specified array.

C#

```
public static NArray FromArray(Array value);
```

Parameters

Parameters	Description
Array value	The original array.

Returns

The new array.

7.2.1.1.2.1.1.2 NArray.FromArray Method (Array, NAttributes)

Creates a new array from the specified array with specified attributes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static NArray FromArray(Array value, NAttributes attributes);
```

Parameters

Parameters	Description
Array value	The original array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.3 NArray.FromArray Method (IntPtr[])

Creates a new array from the specified integer pointer value array.

C#

```
public static NArray FromArray(IntPtr[] value);
```

Parameters

Parameters	Description
IntPtr[] value	The original integer pointer array.

Returns

The new array.

7.2.1.1.2.1.1.4 NArray.FromArray Method (IntPtr[], NAttributes)

Creates a new array from the specified integer pointer value array.

C#

```
public static NArray FromArray(IntPtr[] value, NAttributes attributes);
```

Parameters

Parameters	Description
IntPtr[] value	The original integer pointer array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.5 NArray.FromArray Method (NType, NObject[])

Creates a new array from the specified object array.

C#

```
public static NArray FromArray(NType type, NObject[] value);
```

Parameters

Parameters	Description
NType type	Type of the objects the array.
NObject[] value	The original object array.

Returns

The new array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.1.6 NArray.FromArray Method (NType, NObject[], NAttributes)

Creates a new array from the specified object array.

C#

```
public static NArray FromArray(NType type, NObject[] value, NAttributes attributes);
```

Parameters

Parameters	Description
NType type	Type of the objects in the array.
NObject[] value	The original object array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.7 NArray.FromArray Method (T[])

Creates a new array from the specified array of the specified type.

C#

```
public static NArray FromArray<T>(T[] value);
```

Parameters

Parameters	Description
T[] value	The original array.

Type Parameters

Type Parameters	Description
T	Array type.

Returns

The new array.

7.2.1.1.2.1.1.8 NArray.FromArray Method (T[], NAttributes)

Creates a new array from the specified array of the specified type.

C#

```
public static NArray FromArray<T>(T[] value, NAttributes attributes);
```

Parameters

Parameters	Description
T[] value	The original array.
NAttributes attributes	Array attributes.

Type Parameters

Type Parameters	Description
T	Array type.

Returns

The new array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.1.9 NArray.FromArray Method (UIntPtr[])

Creates a new array from the specified unsigned integer pointer array.

C#

```
public static NArray FromArray(UIntPtr[] value);
```

Parameters

Parameters	Description
UIntPtr[] value	The original unsigned integer pointer array.

Returns

The new array.

7.2.1.1.2.1.1.10 NArray.FromArray Method (UIntPtr[], NAttributes)

Creates a new array from the specified unsigned integer pointer array.

C#

```
public static NArray FromArray(UIntPtr[] value, NAttributes attributes);
```

Parameters

Parameters	Description
UIntPtr[] value	The original unsigned integer pointer array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.11 NArray.FromArray Method (bool[])

Creates a new array from the specified Boolean value array.

C#

```
public static NArray FromArray(bool[] value);
```

Parameters

Parameters	Description
bool[] value	The original Boolean value array.

Returns

The new array.

7.2.1.1.2.1.1.12 NArray.FromArray Method (bool[], NAttributes)

Creates a new array from the specified Boolean value array.

C#

```
public static NArray FromArray(bool[] value, NAttributes attributes);
```

Parameters

Parameters	Description
bool[] value	The original Boolean value array.
NAttributes attributes	Array attributes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The new array.

7.2.1.1.2.1.1.13 NArray.FromArray Method (byte[])

Creates a new array from the specified byte array.

C#

```
public static NArray FromArray(byte[] value);
```

Parameters

Parameters	Description
byte[] value	The original byte array.

Returns

The new array.

7.2.1.1.2.1.1.14 NArray.FromArray Method (byte[], NAttributes)

Creates a new array from the specified byte array.

C#

```
public static NArray FromArray(byte[] value, NAttributes attributes);
```

Parameters

Parameters	Description
byte[] value	The original byte array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.15 NArray.FromArray Method (char[])

Creates a new array from the specified char array.

C#

```
public static NArray FromArray(char[] value);
```

Parameters

Parameters	Description
char[] value	The original char array.

Returns

The new array.

7.2.1.1.2.1.1.16 NArray.FromArray Method (char[], NAttributes)

Creates a new array from the specified char array.

C#

```
public static NArray FromArray(char[] value, NAttributes attributes);
```

Parameters

Parameters	Description
char[] value	The original char array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.17 NArray.FromArray Method (double[])

Creates a new array from the specified double value array.

C#

```
public static NArray FromArray(double[] value);
```

Parameters

Parameters	Description
double[] value	The original double value array.

Returns

The new array.

7.2.1.1.2.1.1.18 NArray.FromArray Method (double[], NAttributes)

Creates a new array from the specified double value array.

C#

```
public static NArray FromArray(double[] value, NAttributes attributes);
```

Parameters

Parameters	Description
double[] value	The original double value array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.19 NArray.FromArray Method (float[])

Creates a new array from the specified floating point value array.

C#

```
public static NArray FromArray(float[] value);
```

Parameters

Parameters	Description
float[] value	The original floating point value array.

Returns

The new array.

7.2.1.1.2.1.1.20 NArray.FromArray Method (float[], NAttributes)

Creates a new array from the specified floating point value array.

C#

```
public static NArray FromArray(float[] value, NAttributes attributes);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
float[] value	The original floating point value array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.21 NArray.FromArray Method (int[])

Creates a new array from the specified integer array.

C#

```
public static NArray FromArray(int[] value);
```

Parameters

Parameters	Description
int[] value	The original integer array.

Returns

The new array.

7.2.1.1.2.1.1.22 NArray.FromArray Method (int[], NAttributes)

Creates a new array from the specified integer array.

C#

```
public static NArray FromArray(int[] value, NAttributes attributes);
```

Parameters

Parameters	Description
int[] value	The original integer array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.23 NArray.FromArray Method (long[])

Creates a new array from the specified long integer array.

C#

```
public static NArray FromArray(long[] value);
```

Parameters

Parameters	Description
long[] value	The original long integer array.

Returns

The new array.

7.2.1.1.2.1.1.24 NArray.FromArray Method (long[], NAttributes)

Creates a new array from the specified long integer array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static NArray FromArray(long[] value, NAttributes attributes);
```

Parameters

Parameters	Description
long[] value	The original long integer array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.25 NArray.FromArray Method (sbyte[])

Creates a new array from the specified signed byte array.

C#

```
public static NArray FromArray(sbyte[] value);
```

Parameters

Parameters	Description
sbyte[] value	The original signed byte array.

Returns

The new array.

7.2.1.1.2.1.1.26 NArray.FromArray Method (sbyte[], NAttributes)

Creates a new array from the specified signed byte array.

C#

```
public static NArray FromArray(sbyte[] value, NAttributes attributes);
```

Parameters

Parameters	Description
sbyte[] value	The original signed byte array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.27 NArray.FromArray Method (short[])

Creates a new array from the specified short integer array.

C#

```
public static NArray FromArray(short[] value);
```

Parameters

Parameters	Description
short[] value	The original short integer array.

Returns

The new array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.1.28 NArray.FromArray Method (short[], NAttributes)

Creates a new array from the specified short integer array.

C#

```
public static NArray FromArray(short[] value, NAttributes attributes);
```

Parameters

Parameters	Description
short[] value	The original short integer array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.29 NArray.FromArray Method (string[])

Creates a new array from the specified string array.

C#

```
public static NArray FromArray(string[] value);
```

Parameters

Parameters	Description
string[] value	The original string array.

Returns

The new array.

7.2.1.1.2.1.1.30 NArray.FromArray Method (string[], NAttributes)

Creates a new array from the specified string array.

C#

```
public static NArray FromArray(string[] value, NAttributes attributes);
```

Parameters

Parameters	Description
string[] value	The original string array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.31 NArray.FromArray Method (uint[])

Creates a new array from the specified unsigned integer array.

C#

```
public static NArray FromArray(uint[] value);
```

Parameters

Parameters	Description
uint[] value	The original unsigned integer array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The new array.

7.2.1.1.2.1.1.32 NArray.FromArray Method (uint[], NAttributes)

Creates a new array from the specified unsigned integer array.

C#

```
public static NArray FromArray(uint[] value, NAttributes attributes);
```

Parameters

Parameters	Description
uint[] value	The original unsigned integer array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.33 NArray.FromArray Method (ulong[])

Creates a new array from the specified unsigned long integer array.

C#

```
public static NArray FromArray(ulong[] value);
```

Parameters

Parameters	Description
ulong[] value	The original unsigned long integer array.

Returns

The new array.

7.2.1.1.2.1.1.34 NArray.FromArray Method (ulong[], NAttributes)

Creates a new array from the specified unsigned long integer array.

C#

```
public static NArray FromArray(ulong[] value, NAttributes attributes);
```

Parameters

Parameters	Description
ulong[] value	The original unsigned long integer array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.1.35 NArray.FromArray Method (ushort[])

Creates a new array from the specified unsigned short integer array.

C#

```
public static NArray FromArray(ushort[] value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
ushort[] value	The original unsigned short integer array.

Returns

The new array.

7.2.1.1.2.1.1.36 NArray.FromArray Method (ushort[], NAttributes)

Creates a new array from the specified unsigned short integer array.

C#

```
public static NArray FromArray(ushort[] value, NAttributes attributes);
```

Parameters

Parameters	Description
ushort[] value	The original unsigned short integer array.
NAttributes attributes	Array attributes.

Returns

The new array.

7.2.1.1.2.1.2 FromPointerArray Method

7.2.1.1.2.1.2.1 NArray.FromPointerArray Method (IntPtr[])

Creates a new array from the specified integer pointer value array.

C#

```
public static NArray FromPointerArray(IntPtr[] value);
```

Parameters

Parameters	Description
IntPtr[] value	The original integer pointer array.

Returns

The new pointer array.

7.2.1.1.2.1.2.2 NArray.FromPointerArray Method (IntPtr[], NAttributes)

Creates a new array from the specified integer pointer value array.

C#

```
public static NArray FromPointerArray(IntPtr[] value, NAttributes attributes);
```

Parameters

Parameters	Description
IntPtr[] value	The original integer pointer array.
NAttributes attributes	Array attributes.

Returns

The new pointer array.

7.2.1.1.2.1.3 FromResultArray Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.3.1 NArray.FromResultArray Method (int[])

Creates a new array from the specified result array.

C#

```
public static NArray FromResultArray(int[] value);
```

Parameters

Parameters	Description
int[] value	The original result array.

Returns

The new result array.

7.2.1.1.2.1.3.2 NArray.FromResultArray Method (int[], NAttributes)

Creates a new array from the specified result array.

C#

```
public static NArray FromResultArray(int[] value, NAttributes attributes);
```

Parameters

Parameters	Description
int[] value	The original result array.
NAttributes attributes	Array attributes.

Returns

The new result array.

7.2.1.1.2.1.4 NArray.GetEnumerator Method

Retrieves an enumerator for the current array.

C#

```
public IEnumerator GetEnumerator();
```

Returns

The enumerator.

7.2.1.1.2.1.5 GetValueAs Method

7.2.1.1.2.1.5.1 NArray.GetValueAs Method (int)

Retrieves the of the specified value of the specified type in the array.

C#

```
public T GetValueAs<T>(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Type Parameters

Type Parameters	Description
T	Type of the value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The value.

7.2.1.1.2.1.5.2 NArray.GetValueAs Method (int, NAttributes)

Retrieves the of the specified value of the specified type in the array.

C#

```
public T GetValueAs<T>(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Type Parameters

Type Parameters	Description
T	Type of the value.

Returns

The value.

7.2.1.1.2.1.5.3 NArray.GetValueAs Method (int, NAttributes, string)

Retrieves the of the specified value of the specified type in the array.

C#

```
public T GetValueAs<T>(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Type Parameters

Type Parameters	Description
T	Type of the value.

Returns

The value.

7.2.1.1.2.1.5.4 NArray.GetValueAs Method (int, Type)

C#

```
public object GetValueAs(int index, Type type);
```

Parameters

Parameters	Description
int index	Index of the value.
Type type	Type of the value.

Returns

The value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.5.5 NArray.GetValueAs Method (int, Type, NAttributes)

Retrieves the specified value in the array.

C#

```
public object GetValueAs(int index, Type type, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
Type type	Type of the value.
NAttributes attributes	Array attributes.

Returns

The value.

7.2.1.1.2.1.5.6 NArray.GetValueAs Method (int, Type, NAttributes, string)

Retrieves the specified value in the array.

C#

```
public object GetValueAs(int index, Type type, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
Type type	Type of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The value.

7.2.1.1.2.1.6 GetValueAsBoolean Method

7.2.1.1.2.1.6.1 NArray.GetValueAsBoolean Method (int)

Retrieves the specified Boolean value in the array.

C#

```
public bool GetValueAsBoolean(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The Boolean value.

7.2.1.1.2.1.6.2 NArray.GetValueAsBoolean Method (int, NAttributes)

Retrieves the specified Boolean value in the array.

C#

```
public bool GetValueAsBoolean(int index, NAttributes attributes);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The Boolean value.

7.2.1.1.2.1.6.3 NArray.GetValueAsBoolean Method (int, NAttributes, string)

Retrieves the specified Boolean value in the array.

C#

```
public bool GetValueAsBoolean(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The Boolean value.

7.2.1.1.2.1.7 GetValueAsByte Method

7.2.1.1.2.1.7.1 NArray.GetValueAsByte Method (int)

Retrieves the specified byte value in the array.

C#

```
public byte GetValueAsByte(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The byte value.

7.2.1.1.2.1.7.2 NArray.GetValueAsByte Method (int, NAttributes)

Retrieves the specified byte value in the array.

C#

```
public byte GetValueAsByte(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The byte value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.7.3 NArray.GetValueAsByte Method (int, NAttributes, string)

Retrieves the specified byte value in the array.

C#

```
public byte GetValueAsByte(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The byte value.

7.2.1.1.2.1.8 GetValueAsChar Method

7.2.1.1.2.1.8.1 NArray.GetValueAsChar Method (int)

Retrieves the specified char value in the array.

C#

```
public char GetValueAsChar(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The string value.

7.2.1.1.2.1.8.2 NArray.GetValueAsChar Method (int, NAttributes)

Retrieves the specified char value in the array.

C#

```
public char GetValueAsChar(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The char value.

7.2.1.1.2.1.8.3 NArray.GetValueAsChar Method (int, NAttributes, string)

Retrieves the specified char value in the array.

C#

```
public char GetValueAsChar(int index, NAttributes attributes, string format);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The char value.

7.2.1.1.2.1.9 GetValueAsDouble Method

7.2.1.1.2.1.9.1 NArray.GetValueAsDouble Method (int)

Retrieves the specified double value in the array.

C#

```
public double GetValueAsDouble(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The double value.

7.2.1.1.2.1.9.2 NArray.GetValueAsDouble Method (int, NAttributes)

Retrieves the specified double value in the array.

C#

```
public double GetValueAsDouble(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The double value.

7.2.1.1.2.1.9.3 NArray.GetValueAsDouble Method (int, NAttributes, string)

Retrieves the specified double value in the array.

C#

```
public double GetValueAsDouble(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The double value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.10 GetValueAsInt16 Method

7.2.1.1.2.1.10.1 NArray.GetValueAsInt16 Method (int)

Retrieves the specified short integer value in the array.

C#

```
public short GetValueAsInt16(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The short integer value.

7.2.1.1.2.1.10.2 NArray.GetValueAsInt16 Method (int, NAttributes)

Retrieves the specified short integer value in the array.

C#

```
public short GetValueAsInt16(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The short integer value.

7.2.1.1.2.1.10.3 NArray.GetValueAsInt16 Method (int, NAttributes, string)

Retrieves the specified short integer value in the array.

C#

```
public short GetValueAsInt16(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The short integer value.

7.2.1.1.2.1.11 GetValueAsInt32 Method

7.2.1.1.2.1.11.1 NArray.GetValueAsInt32 Method (int)

Retrieves the specified integer value in the array.

C#

```
public int GetValueAsInt32(int index);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
int index	Index of the value.

Returns

The integer value.

7.2.1.1.2.1.11.2 NArray.GetValueAsInt32 Method (int, NAttributes)

Retrieves the specified integer value in the array.

C#

```
public int GetValueAsInt32(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The integer value.

7.2.1.1.2.1.11.3 NArray.GetValueAsInt32 Method (int, NAttributes, string)

Retrieves the specified integer value in the array.

C#

```
public int GetValueAsInt32(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The integer value.

7.2.1.1.2.1.12 GetValueAsInt64 Method

7.2.1.1.2.1.12.1 NArray.GetValueAsInt64 Method (int)

Retrieves the specified long integer value in the array.

C#

```
public long GetValueAsInt64(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The long integer value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.12.2 NArray.GetValueAsInt64 Method (int, NAttributes)

Retrieves the specified long integer value in the array.

C#

```
public long GetValueAsInt64(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The long integer value.

7.2.1.1.2.1.12.3 NArray.GetValueAsInt64 Method (int, NAttributes, string)

Retrieves the specified long integer value in the array.

C#

```
public long GetValueAsInt64(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The long integer value.

7.2.1.1.2.1.13 GetValueAsIntPtr Method

7.2.1.1.2.1.13.1 NArray.GetValueAsIntPtr Method (int)

Retrieves the specified integer pointer value in the array.

C#

```
public IntPtr GetValueAsIntPtr(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The integer pointer value.

7.2.1.1.2.1.13.2 NArray.GetValueAsIntPtr Method (int, NAttributes)

Retrieves the specified integer pointer value in the array.

C#

```
public IntPtr GetValueAsIntPtr(int index, NAttributes attributes);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The integer pointer value.

7.2.1.1.2.1.13.3 NArray.GetValueAsIntPtr Method (int, NAttributes, string)

Retrieves the specified integer pointer value in the array.

C#

```
public IntPtr GetValueAsIntPtr(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The integer pointer value.

7.2.1.1.2.1.14 GetValueAsObject Method

7.2.1.1.2.1.14.1 NArray.GetValueAsObject Method (int, NType)

Retrieves the specified object in the array.

C#

```
public NObject GetValueAsObject(int index, NType type);
```

Parameters

Parameters	Description
int index	Index of the value.
NType type	Type of the object.

Returns

The object.

7.2.1.1.2.1.14.2 NArray.GetValueAsObject Method (int, NType, NAttributes)

Retrieves the specified object in the array.

C#

```
public NObject GetValueAsObject(int index, NType type, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NType type	Type of the object.
NAttributes attributes	Array attributes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The object.

7.2.1.1.2.1.14.3 NArray.GetValueAsObject Method (int, NType, NAttributes, string)

Retrieves the specified object in the array.

C#

```
public NObject GetValueAsObject(int index, NType type, NAttributes attributes, string
format);
```

Parameters

Parameters	Description
int index	Index of the value.
NType type	Type of the object.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The object.

7.2.1.1.2.1.15 GetValueAsPointer Method

7.2.1.1.2.1.15.1 NArray.GetValueAsPointer Method (int)

Retrieves the specified integer pointer value in the array.

C#

```
public IntPtr GetValueAsPointer(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The pointer value.

7.2.1.1.2.1.15.2 NArray.GetValueAsPointer Method (int, NAttributes)

Retrieves the specified integer pointer value in the array.

C#

```
public IntPtr GetValueAsPointer(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The pointer value.

7.2.1.1.2.1.15.3 NArray.GetValueAsPointer Method (int, NAttributes, string)

Retrieves the specified integer pointer value in the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public IntPtr GetValueAsPointer(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The pointer value.

7.2.1.1.2.1.16 GetValueAsResult Method

7.2.1.1.2.1.16.1 NArray.GetValueAsResult Method (int)

Retrieves the specified result value in the array.

C#

```
public int GetValueAsResult(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The result value.

7.2.1.1.2.1.16.2 NArray.GetValueAsResult Method (int, NAttributes)

Retrieves the specified result value in the array.

C#

```
public int GetValueAsResult(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The result value.

7.2.1.1.2.1.16.3 NArray.GetValueAsResult Method (int, NAttributes, string)

Retrieves the specified result value in the array.

C#

```
public int GetValueAsResult(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The result value.

7.2.1.1.2.1.17 GetValueAsSByte Method

7.2.1.1.2.1.17.1 NArray.GetValueAsSByte Method (int)

Retrieves the specified signed byte value in the array.

C#

```
public sbyte GetValueAsSByte(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The signed byte value.

7.2.1.1.2.1.17.2 NArray.GetValueAsSByte Method (int, NAttributes)

Retrieves the specified signed byte value in the array.

C#

```
public sbyte GetValueAsSByte(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The signed byte value.

7.2.1.1.2.1.17.3 NArray.GetValueAsSByte Method (int, NAttributes, string)

Retrieves the specified signed byte value in the array.

C#

```
public sbyte GetValueAsSByte(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The signed byte value.

7.2.1.1.2.1.18 GetValueAsSingle Method

7.2.1.1.2.1.18.1 NArray.GetValueAsSingle Method (int)

Retrieves the specified floating point value in the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public float GetValueAsSingle(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The floating point value.

7.2.1.1.2.1.18.2 NArray.GetValueAsSingle Method (int, NAttributes)

Retrieves the specified floating point value in the array.

C#

```
public float GetValueAsSingle(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The floating point value.

7.2.1.1.2.1.18.3 NArray.GetValueAsSingle Method (int, NAttributes, string)

Retrieves the specified floating point value in the array.

C#

```
public float GetValueAsSingle(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The floating point value.

7.2.1.1.2.1.19 GetValueAsString Method

7.2.1.1.2.1.19.1 NArray.GetValueAsString Method (int)

Retrieves the specified string value in the array.

C#

```
public string GetValueAsString(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The string value.

7.2.1.1.2.1.19.2 NArray.GetValueAsString Method (int, NAttributes)

Retrieves the specified string value in the array.

C#

```
public string GetValueAsString(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

7.2.1.1.2.1.19.3 NArray.GetValueAsString Method (int, NAttributes, string)

Retrieves the specified string value in the array.

C#

```
public string GetValueAsString(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The string value.

7.2.1.1.2.1.20 GetValueAsUInt16 Method

7.2.1.1.2.1.20.1 NArray.GetValueAsUInt16 Method (int)

Retrieves the specified unsigned short integer value in the array.

C#

```
public ushort GetValueAsUInt16(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The unsigned short integer value.

7.2.1.1.2.1.20.2 NArray.GetValueAsUInt16 Method (int, NAttributes)

Retrieves the specified unsigned short integer value in the array.

C#

```
public ushort GetValueAsUInt16(int index, NAttributes attributes);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The unsigned short integer value.

7.2.1.1.2.1.20.3 NArray.GetValueAsUInt16 Method (int, NAttributes, string)

Retrieves the specified unsigned short integer value in the array.

C#

```
public ushort GetValueAsUInt16(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The unsigned short integer value.

7.2.1.1.2.1.21 GetValueAsUInt32 Method

7.2.1.1.2.1.21.1 NArray.GetValueAsUInt32 Method (int)

Retrieves the specified unsigned integer value in the array.

C#

```
public uint GetValueAsUInt32(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The unsigned integer value.

7.2.1.1.2.1.21.2 NArray.GetValueAsUInt32 Method (int, NAttributes)

Retrieves the specified unsigned integer value in the array.

C#

```
public uint GetValueAsUInt32(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The unsigned integer value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.21.3 NArray.GetValueAsUInt32 Method (int, NAttributes, string)

Retrieves the specified unsigned integer value in the array.

C#

```
public uint GetValueAsUInt32(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The unsigned integer value.

7.2.1.1.2.1.22 GetValueAsUInt64 Method

7.2.1.1.2.1.22.1 NArray.GetValueAsUInt64 Method (int)

Retrieves the specified unsigned long integer value in the array.

C#

```
public ulong GetValueAsUInt64(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The unsigned long integer value.

7.2.1.1.2.1.22.2 NArray.GetValueAsUInt64 Method (int, NAttributes)

Retrieves the specified unsigned long integer value in the array.

C#

```
public ulong GetValueAsUInt64(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The unsigned long integer value.

7.2.1.1.2.1.22.3 NArray.GetValueAsUInt64 Method (int, NAttributes, string)

Retrieves the specified unsigned long integer value in the array.

C#

```
public ulong GetValueAsUInt64(int index, NAttributes attributes, string format);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The unsigned long integer value.

7.2.1.1.2.1.23 GetValueAsUIntPtr Method

7.2.1.1.2.1.23.1 NArray.GetValueAsUIntPtr Method (int)

Retrieves the specified unsigned integer pointer value in the array.

C#

```
public UIntPtr GetValueAsUIntPtr(int index);
```

Parameters

Parameters	Description
int index	Index of the value.

Returns

The unsigned integer pointer value.

7.2.1.1.2.1.23.2 NArray.GetValueAsUIntPtr Method (int, NAttributes)

Retrieves the specified unsigned integer pointer value in the array.

C#

```
public UIntPtr GetValueAsUIntPtr(int index, NAttributes attributes);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.

Returns

The unsigned integer pointer value.

7.2.1.1.2.1.23.3 NArray.GetValueAsUIntPtr Method (int, NAttributes, string)

Retrieves the specified unsigned integer pointer value in the array.

C#

```
public UIntPtr GetValueAsUIntPtr(int index, NAttributes attributes, string format);
```

Parameters

Parameters	Description
int index	Index of the value.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

The unsigned integer pointer value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.24 ToArray Method

7.2.1.1.2.1.24.1 NArray.ToArray Method ()

Copies the current array into another array of the specified type.

C#

```
public T[] ToArray<T>();
```

Type Parameters

Type Parameters	Description
T	Type of the array.

Returns

Destination array.

7.2.1.1.2.1.24.2 NArray.ToArray Method (NArray)

Creates a new array from the specified NArray [1471].

C#

```
public static Array ToArray(NArray value);
```

Parameters

Parameters	Description
NArray value	The original NArray [1471]

Returns

The new array.

7.2.1.1.2.1.24.3 NArray.ToArray Method (NAttributes)

Copies the current array into another array of the specified type.

C#

```
public T[] ToArray<T>(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Type Parameters

Type Parameters	Description
T	Type of the array.

Returns

Destination array.

7.2.1.1.2.1.24.4 NArray.ToArray Method (NAttributes, string)

Copies the current array into another array of the specified type.

C#

```
public T[] ToArray<T>(NAttributes attributes, string format);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Type Parameters

Type Parameters	Description
T	Type of the array.

Returns

Destination array.

7.2.1.1.2.1.24.5 NArray.ToArray Method (Type)

Copies the current array of the specified type into another array.

C#

```
public Array ToArray(Type type);
```

Parameters

Parameters	Description
Type type	Type of the array.

Returns

Destination array.

7.2.1.1.2.1.24.6 NArray.ToArray Method (Type, NAttributes)

Copies the current array of the specified type into another array.

C#

```
public Array ToArray(Type type, NAttributes attributes);
```

Parameters

Parameters	Description
Type type	Type of the array.
NAttributes attributes	Array attributes.

Returns

Destination array.

7.2.1.1.2.1.24.7 NArray.ToArray Method (Type, NAttributes, string)

Copies the current array of the specified type into another array.

C#

```
public Array ToArray(Type type, NAttributes attributes, string format);
```

Parameters

Parameters	Description
Type type	Type of the array.
NAttributes attributes	Array attributes.
string format	Value format.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Destination array.

7.2.1.1.2.1.25 ToBooleanArray Method

7.2.1.1.2.1.25.1 NArray.ToBooleanArray Method ()

Copies the current array into a Boolean value array.

C#

```
public bool[] ToBooleanArray();
```

Returns

Destination Boolean value array.

7.2.1.1.2.1.25.2 NArray.ToBooleanArray Method (NAttributes)

Copies the current array into a Boolean value array.

C#

```
public bool[] ToBooleanArray(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination Boolean value array.

7.2.1.1.2.1.25.3 NArray.ToBooleanArray Method (NAttributes, string)

Copies the current array into a Boolean value array.

C#

```
public bool[] ToBooleanArray(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination Boolean value array.

7.2.1.1.2.1.26 ToByteArray Method

7.2.1.1.2.1.26.1 NArray.ToByteArray Method ()

Copies the current array into a byte array.

C#

```
public byte[] ToByteArray();
```

Returns

Destination byte array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.26.2 NArray.ToByteArray Method (NAttributes)

Copies the current array into a byte array.

C#

```
public byte[] ToByteArray(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination byte array.

7.2.1.1.2.1.26.3 NArray.ToByteArray Method (NAttributes, string)

Copies the current array into a byte array.

C#

```
public byte[] ToByteArray(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination byte array.

7.2.1.1.2.1.27 ToCharArray Method

7.2.1.1.2.1.27.1 NArray.ToCharArray Method ()

Copies the current array into a char array.

C#

```
public char[] ToCharArray();
```

Returns

Destination char array.

7.2.1.1.2.1.27.2 NArray.ToCharArray Method (NAttributes)

Copies the current array into a char array.

C#

```
public char[] ToCharArray(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination char array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.27.3 NArray.ToCharArray Method (NAttributes, string)

Copies the current array into a char array.

C#

```
public char[] ToCharArray(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination char array.

7.2.1.1.2.1.28 ToDoubleArray Method

7.2.1.1.2.1.28.1 NArray.ToDoubleArray Method ()

Copies the current array into a double value array.

C#

```
public double[] ToDoubleArray();
```

Returns

Destination double value array.

7.2.1.1.2.1.28.2 NArray.ToDoubleArray Method (NAttributes)

Copies the current array into a double value array.

C#

```
public double[] ToDoubleArray(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination double value array.

7.2.1.1.2.1.28.3 NArray.ToDoubleArray Method (NAttributes, string)

Copies the current array into a double value array.

C#

```
public double[] ToDoubleArray(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination double value array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.29 ToInt16Array Method

7.2.1.1.2.1.29.1 NArray.ToInt16Array Method ()

Copies the current array into a short integer array.

C#

```
public short[] ToInt16Array();
```

Returns

Destination short integer array.

7.2.1.1.2.1.29.2 NArray.ToInt16Array Method (NAttributes)

Copies the current array into a short integer array.

C#

```
public short[] ToInt16Array(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination short integer array.

7.2.1.1.2.1.29.3 NArray.ToInt16Array Method (NAttributes, string)

Copies the current array into a short integer array.

C#

```
public short[] ToInt16Array(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination short integer array.

7.2.1.1.2.1.30 ToInt32Array Method

7.2.1.1.2.1.30.1 NArray.ToInt32Array Method ()

Copies the current array into an integer array.

C#

```
public int[] ToInt32Array();
```

Returns

Destination integer array.

7.2.1.1.2.1.30.2 NArray.ToInt32Array Method (NAttributes)

Copies the current array into an integer array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public int[] ToInt32Array(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination integer array.

7.2.1.1.2.1.30.3 NArray.ToInt32Array Method (NAttributes, string)

Copies the current array into an integer array.

C#

```
public int[] ToInt32Array(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination unsigned integer array.

7.2.1.1.2.1.31 ToInt64Array Method

7.2.1.1.2.1.31.1 NArray.ToInt64Array Method ()

Copies the current array into a long integer array.

C#

```
public long[] ToInt64Array();
```

Returns

Destination long integer array.

7.2.1.1.2.1.31.2 NArray.ToInt64Array Method (NAttributes)

Copies the current array into a long integer array.

C#

```
public long[] ToInt64Array(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination long integer array.

7.2.1.1.2.1.31.3 NArray.ToInt64Array Method (NAttributes, string)

Copies the current array into a long integer array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public long[] ToInt64Array(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination long integer array.

7.2.1.1.2.1.32 ToIntPtrArray Method

7.2.1.1.2.1.32.1 NArray.ToIntPtrArray Method ()

Copies the current array into an integer pointer array.

C#

```
public IntPtr[] ToIntPtrArray();
```

Returns

Destination integer pointer array.

7.2.1.1.2.1.32.2 NArray.ToIntPtrArray Method (NAttributes)

Copies the current array into an integer pointer array.

C#

```
public IntPtr[] ToIntPtrArray(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination integer pointer array.

7.2.1.1.2.1.32.3 NArray.ToIntPtrArray Method (NAttributes, string)

Copies the current array into an integer pointer array.

C#

```
public IntPtr[] ToIntPtrArray(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination integer pointer array.

7.2.1.1.2.1.33 ToObjectArray Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.33.1 NArray.ToObjectArray Method (NType)

Copies the current array into an object array.

C#

```
public NObject[] ToObjectArray(NType type);
```

Parameters

Parameters	Description
NType type	Object type.

Returns

Destination object array.

7.2.1.1.2.1.33.2 NArray.ToObjectArray Method (NType, NAttributes)

Copies the current array into an object array.

C#

```
public NObject[] ToObjectArray(NType type, NAttributes attributes);
```

Parameters

Parameters	Description
NType type	Object type.
NAttributes attributes	Array attributes.

Returns

Destination object array.

7.2.1.1.2.1.33.3 NArray.ToObjectArray Method (NType, NAttributes, string)

Copies the current array into an object array.

C#

```
public NObject[] ToObjectArray(NType type, NAttributes attributes, string format);
```

Parameters

Parameters	Description
NType type	Object type.
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination object array.

7.2.1.1.2.1.34 ToPointerArray Method

7.2.1.1.2.1.34.1 NArray.ToPointerArray Method ()

Copies the current array into a pointer array.

C#

```
public IntPtr[] ToPointerArray();
```

Returns

Destination pointer array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.34.2 NArray.ToPointerArray Method (NAttributes)

Copies the current array into a pointer array.

C#

```
public IntPtr[] ToPointerArray(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination pointer array.

7.2.1.1.2.1.34.3 NArray.ToPointerArray Method (NAttributes, string)

Copies the current array into a pointer array.

C#

```
public IntPtr[] ToPointerArray(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination pointer array.

7.2.1.1.2.1.35 ToResultArray Method

7.2.1.1.2.1.35.1 NArray.ToResultArray Method ()

Copies the current array into a result array.

C#

```
public int[] ToResultArray();
```

Returns

Destination result array.

7.2.1.1.2.1.35.2 NArray.ToResultArray Method (NAttributes)

Copies the current array into a result array.

C#

```
public int[] ToResultArray(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination result array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.35.3 NArray.ToResultArray Method (NAttributes, string)

Copies the current array into a result array.

C#

```
public int[] ToResultArray(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination result array.

7.2.1.1.2.1.36 ToSByteArray Method

7.2.1.1.2.1.36.1 NArray.ToSByteArray Method ()

Copies the current array into a signed byte array.

C#

```
public sbyte[] ToSByteArray();
```

Returns

Destination signed byte array.

7.2.1.1.2.1.36.2 NArray.ToSByteArray Method (NAttributes)

Copies the current array into a signed byte array.

C#

```
public sbyte[] ToSByteArray(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination signed byte array.

7.2.1.1.2.1.36.3 NArray.ToSByteArray Method (NAttributes, string)

Copies the current array into a signed byte array.

C#

```
public sbyte[] ToSByteArray(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination signed byte array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.37 ToSingleArray Method

7.2.1.1.2.1.37.1 NArray.ToSingleArray Method ()

Copies the current array into a floating point value array.

C#

```
public float[] ToSingleArray();
```

Returns

Destination floating point value array.

7.2.1.1.2.1.37.2 NArray.ToSingleArray Method (NAttributes)

Copies the current array into a floating point value array.

C#

```
public float[] ToSingleArray(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination floating point value array.

7.2.1.1.2.1.37.3 NArray.ToSingleArray Method (NAttributes, string)

Copies the current array into a floating point value array.

C#

```
public float[] ToSingleArray(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination floating point value array.

7.2.1.1.2.1.38 ToStringArray Method

7.2.1.1.2.1.38.1 NArray.ToStringArray Method ()

Copies the current array into a string array.

C#

```
public string[] ToStringArray();
```

Returns

Destination string array.

7.2.1.1.2.1.38.2 NArray.ToStringArray Method (NAttributes)

Copies the current array into a string array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public string[] ToStringArray(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination string array.

7.2.1.1.2.1.38.3 NArray.ToStringArray Method (NAttributes, string)

Copies the current array into a string array.

C#

```
public string[] ToStringArray(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination string array.

7.2.1.1.2.1.39 ToUInt16Array Method

7.2.1.1.2.1.39.1 NArray.ToUInt16Array Method ()

Copies the current array into an unsigned short integer array.

C#

```
public ushort[] ToUInt16Array();
```

Returns

Destination unsigned short integer array.

7.2.1.1.2.1.39.2 NArray.ToUInt16Array Method (NAttributes)

Copies the current array into an unsigned short integer array.

C#

```
public ushort[] ToUInt16Array(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination unsigned short integer array.

7.2.1.1.2.1.39.3 NArray.ToUInt16Array Method (NAttributes, string)

Copies the current array into an unsigned short integer array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public ushort[] ToUInt16Array(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination unsigned short integer array.

7.2.1.1.2.1.40 ToUInt32Array Method

7.2.1.1.2.1.40.1 NArray.ToUInt32Array Method ()

Copies the current array into an unsigned integer array.

C#

```
public uint[] ToUInt32Array();
```

Returns

Destination unsigned integer array.

7.2.1.1.2.1.40.2 NArray.ToUInt32Array Method (NAttributes)

Copies the current array into an unsigned integer array.

C#

```
public uint[] ToUInt32Array(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination unsigned integer array.

7.2.1.1.2.1.40.3 NArray.ToUInt32Array Method (NAttributes, string)

Copies the current array into an unsigned integer array.

C#

```
public uint[] ToUInt32Array(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination unsigned integer array.

7.2.1.1.2.1.41 ToUInt64Array Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.2.1.41.1 NArray.ToUInt64Array Method ()

Copies the current unsigned long integer array into another array.

C#

```
public ulong[] ToUInt64Array();
```

Returns

Destination unsigned long integer array.

7.2.1.1.2.1.41.2 NArray.ToUInt64Array Method (NAttributes)

Copies the current array into an unsigned long integer array.

C#

```
public ulong[] ToUInt64Array(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination unsigned long integer array.

7.2.1.1.2.1.41.3 NArray.ToUInt64Array Method (NAttributes, string)

Copies the current array into an unsigned long integer array.

C#

```
public ulong[] ToUInt64Array(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination unsigned long integer array.

7.2.1.1.2.1.42 ToUIntPtrArray Method

7.2.1.1.2.1.42.1 NArray.ToUIntPtrArray Method ()

Copies the current array into an unsigned integer pointer array.

C#

```
public UIntPtr[] ToUIntPtrArray();
```

Returns

Destination unsigned integer pointer array.

7.2.1.1.2.1.42.2 NArray.ToUIntPtrArray Method (NAttributes)

Copies the current array into an unsigned integer pointer array.

C#

```
public UIntPtr[] ToUIntPtrArray(NAttributes attributes);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NAttributes attributes	Array attributes.

Returns

Destination unsigned integer pointer array.

7.2.1.1.2.1.42.3 NArray.ToUIntPtrArray Method (NAttributes, string)

Copies the current array into an unsigned integer pointer array.

C#

```
public UIntPtr[] ToUIntPtrArray(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Array attributes.
string format	Value format.

Returns

Destination unsigned integer pointer array.

7.2.1.1.2.2 NArray Properties

7.2.1.1.2.2.1 NArray.Length Property

Retrieves the length of the array.

C#

```
public int Length;
```

7.2.1.1.2.2.2 NArray.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.1.1.3 NComplex Structure

Represents a complex number.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NComplex : IFormattable {
  public static readonly NComplex Zero = new NComplex(0, 0);
  public static readonly NComplex One = new NComplex(1, 0);
  public static readonly NComplex ImaginaryOne = new NComplex(0, 1);
}
```

NComplex Fields

	Name	Description
S R	ImaginaryOne [1520]	Represents a NComplex with real part equal to zero and imaginary part equal to one.
S R	One [1520]	Represents NComplex with real part equal to one and imaginary part equal to zero.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S R	Zero [1520]	Represents a NComplex with real part equal to zero and imaginary part equal to zero.

Methods

	Name	Description
	NComplex [1520]	Initializes a new instance of the NComplex structure.

NComplex Methods

	Name	Description
	ToString [1521]	Retrieves a string representation of the object.

NComplex Properties

	Name	Description
	Imaginary [1521]	Sets or retrieves the imaginary part of NComplex value.
S	NativeType [1521]	Defines the native type of the type.
	Real [1521]	Sets or retrieves the real part of NComplex value.

7.2.1.1.3.1 NComplex.NComplex Constructor

Initializes a new instance of the NComplex structure.

C#

```
public NComplex(double real, double imaginary);
```

Parameters

Parameters	Description
double real	Real [1521] part of this NComplex.
double imaginary	Imaginary [1521] part of this NComplex.

7.2.1.1.3.2 NComplex Fields

7.2.1.1.3.2.1 NComplex.ImaginaryOne Field

Represents a NComplex [1519] with real part equal to zero and imaginary part equal to one.

C#

```
public static readonly NComplex ImaginaryOne = new NComplex(0, 1);
```

7.2.1.1.3.2.2 NComplex.One Field

Represents NComplex [1519] with real part equal to one and imaginary part equal to zero.

C#

```
public static readonly NComplex One = new NComplex(1, 0);
```

7.2.1.1.3.2.3 NComplex.Zero Field

Represents a NComplex [1519] with real part equal to zero and imaginary part equal to zero.

C#

```
public static readonly NComplex Zero = new NComplex(0, 0);
```

7.2.1.1.3.3 NComplex Methods

7.2.1.1.3.3.1 ToString Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.3.3.1.1 NComplex.ToString Method ()

Retrieves a string representation of the object.

C#

```
public override string ToString();
```

Returns

A string representation of an object.

7.2.1.1.3.3.1.2 NComplex.ToString Method (string)

Retrieves a string representation of the object.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	Format string.

Returns

A string representation of an object.

7.2.1.1.3.4 NComplex Properties

7.2.1.1.3.4.1 NComplex.Imaginary Property

Sets or retrieves the imaginary part of NComplex [1519] value.

C#

```
public double Imaginary;
```

7.2.1.1.3.4.2 NComplex.NativeType Property

Defines the native type of the type.

C#

```
public static NType NativeType;
```

7.2.1.1.3.4.3 NComplex.Real Property

Sets or retrieves the real part of NComplex [1519] value.

C#

```
public double Real;
```

7.2.1.1.4 NCore Class

This class supports internal Neurotechnology libraries infrastructure and should not be used directly in your code.

C#

```
public static class NCore;
```

NCore Events

	Name	Description
	ErrorSuppressed [1536]	Occurs when an error is supressed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCore Fields

	Name	Description
	DllName [1522]	NCore [257] library name.

NCore Methods

	Name	Description
	AlignedAlloc [1522]	Allocates a block of memory and makes sure that the returned pointer can be divided evenly by alignment.
	AlignedCAlloc [1524]	Allocates an array in memory with elements initialized to 0 and makes sure that the returned pointer can be divided evenly by alignment.
	AlignedFree [1526]	Frees a specified aligned memory block.
	AlignedReAlloc [1526]	Reallocate memory blocks, represented by a pointer that can be divided evenly by the alignment.
	Alloc [1528]	Allocates a block of memory.
	CAlloc [1529]	Allocates an array in memory with elements initialized to 0.
	Clear [1530]	Clears all elements in the specified array.
	Compare [1530]	Compares two memory blocks.
	Copy [1531]	Copies data from one memory block to other.
	Fill [1532]	Sets memory block to a specified value.
	Free [1533]	Frees a specified memory block.
	Move [1534]	Moves one memory block to another.
	ReAlloc [1535]	Reallocate memory blocks.
	Shutdown [1535]	Disables sends and receives on a socket.
	ShutdownThread [1535]	Disables sends and receives on a thread.

NCore Properties

	Name	Description
	NativeModule [1536]	Defines the native module of the object.

7.2.1.1.4.1 NCore Fields

7.2.1.1.4.1.1 NCore.DllName Field

NCore [257] library name.

C#

```
public const string DllName = "NCore";
```

7.2.1.1.4.2 NCore Methods

7.2.1.1.4.2.1 AlignedAlloc Method

7.2.1.1.4.2.1.1 NCore.AlignedAlloc Method (UIntPtr, UIntPtr)

Allocates a block of memory and makes sure that the returned pointer can be divided evenly by alignment.

C#

```
public static IntPtr AlignedAlloc(UIntPtr size, UIntPtr alignment);
```

Parameters

Parameters	Description
UIntPtr size	Size in bytes of memory to be allocated.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

UIntPtr alignment	The alignment.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block. Can be divided evenly by the alignment.

7.2.1.1.4.2.1.2 NCore.AlignedAlloc Method (UIntPtr, UIntPtr, UIntPtr)

Allocates a block of memory and makes sure that the sum of returned pointer and the offset can be divided evenly by alignment.

C#

```
public static IntPtr AlignedAlloc(UIntPtr size, UIntPtr alignment, UIntPtr offset);
```

Parameters

Parameters	Description
UIntPtr size	Size in bytes of memory to be allocated.
UIntPtr alignment	The alignment.
UIntPtr offset	The offset.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block. Its sum with offset can be divided evenly by the alignment.

7.2.1.1.4.2.1.3 NCore.AlignedAlloc Method (int, int)

Allocates a block of memory and makes sure that the returned pointer can be divided evenly by alignment.

C#

```
public static IntPtr AlignedAlloc(int size, int alignment);
```

Parameters

Parameters	Description
int size	Size in bytes of memory to be allocated.
int alignment	The alignment.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block. Can be divided evenly by the alignment.

7.2.1.1.4.2.1.4 NCore.AlignedAlloc Method (int, int, int)

Allocates a block of memory and makes sure that the sum of returned pointer and the offset can be divided evenly by alignment.

C#

```
public static IntPtr AlignedAlloc(int size, int alignment, int offset);
```

Parameters

Parameters	Description
int size	Size in bytes of memory to be allocated.
int alignment	The alignment.
int offset	The offset.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block. Its sum with offset can be divided evenly by the alignment.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.4.2.1.5 NCore.AlignedAlloc Method (long, long)

Allocates a block of memory and makes sure that the returned pointer can be divided evenly by alignment.

C#

```
public static IntPtr AlignedAlloc(long size, long alignment);
```

Parameters

Parameters	Description
long size	Size in bytes of memory to be allocated.
long alignment	The alignment.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block. Can be divided evenly by the alignment.

7.2.1.1.4.2.1.6 NCore.AlignedAlloc Method (long, long, long)

Allocates a block of memory and makes sure that the sum of returned pointer and the offset can be divided evenly by alignment.

C#

```
public static IntPtr AlignedAlloc(long size, long alignment, long offset);
```

Parameters

Parameters	Description
long size	Size in bytes of memory to be allocated.
long alignment	The alignment.
long offset	The offset.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block. Its sum with offset can be divided evenly by the alignment.

7.2.1.1.4.2.2 AlignedCAlloc Method

7.2.1.1.4.2.2.1 NCore.AlignedCAlloc Method (UIntPtr, UIntPtr)

Allocates an array in memory with elements initialized to 0 and makes sure that the returned pointer can be divided evenly by alignment.

C#

```
public static IntPtr AlignedCAlloc(UIntPtr size, UIntPtr alignment);
```

Parameters

Parameters	Description
UIntPtr size	Size in bytes of memory to be allocated.
UIntPtr alignment	The alignment.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block. Can be divided evenly by the alignment.

7.2.1.1.4.2.2.2 NCore.AlignedCAlloc Method (UIntPtr, UIntPtr, UIntPtr)

Allocates an array in memory with elements initialized to 0 and makes sure that the sum of returned pointer and the offset can be divided evenly by alignment.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static IntPtr AlignedCAlloc(UIntPtr size, UIntPtr alignment, UIntPtr offset);
```

Parameters

Parameters	Description
UIntPtr size	Size in bytes of memory to be allocated.
UIntPtr alignment	The alignment.
UIntPtr offset	The offset.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block. Its sum with offset can be divided evenly by the alignment.

7.2.1.1.4.2.2.3 NCore.AlignedCAlloc Method (int, int)

Allocates an array in memory with elements initialized to 0 and makes sure that the returned pointer can be divided evenly by alignment.

C#

```
public static IntPtr AlignedCAlloc(int size, int alignment);
```

Parameters

Parameters	Description
int size	Size in bytes of memory to be allocated.
int alignment	The alignment.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block. Can be divided evenly by the alignment.

7.2.1.1.4.2.2.4 NCore.AlignedCAlloc Method (int, int, int)

Allocates an array in memory with elements initialized to 0 and makes sure that the sum of returned pointer and the offset can be divided evenly by alignment.

C#

```
public static IntPtr AlignedCAlloc(int size, int alignment, int offset);
```

Parameters

Parameters	Description
int size	Size in bytes of memory to be allocated.
int alignment	The alignment.
int offset	The offset.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block. Its sum with offset can be divided evenly by the alignment.

7.2.1.1.4.2.2.5 NCore.AlignedCAlloc Method (long, long)

Allocates an array in memory with elements initialized to 0 and makes sure that the returned pointer can be divided evenly by alignment.

C#

```
public static IntPtr AlignedCAlloc(long size, long alignment);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
long size	Size in bytes of memory to be allocated.
long alignment	The alignment.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block. Can be divided evenly by the alignment.

7.2.1.1.4.2.2.6 NCore.AlignedCAlloc Method (long, long, long)

Allocates an array in memory with elements initialized to 0 and makes sure that the sum of returned pointer and the offset can be divided evenly by alignment.

C#

```
public static IntPtr AlignedCAlloc(long size, long alignment, long offset);
```

Parameters

Parameters	Description
long size	Size in bytes of memory to be allocated.
long alignment	The alignment.
long offset	The offset.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block. Its sum with offset can be divided evenly by the alignment.

7.2.1.1.4.2.3 NCore.AlignedFree Method

Frees a specified aligned memory block.

C#

```
public static void AlignedFree(IntPtr pBlock);
```

Parameters

Parameters	Description
IntPtr pBlock	A representation of a pointer (IntPtr type) to the memory block. Its sum with the offset an be divided evenly by the memory block's alignment.

7.2.1.1.4.2.4 AlignedReAlloc Method

7.2.1.1.4.2.4.1 NCore.AlignedReAlloc Method (IntPtr, UIntPtr, UIntPtr)

Reallocate memory blocks, represented by a pointer that can be divided evenly by the alignment.

C#

```
public static void AlignedReAlloc(ref IntPtr pBlock, UIntPtr size, UIntPtr alignment);
```

Parameters

Parameters	Description
ref IntPtr pBlock	A representation of a pointer (IntPtr type) to the allocated memory block. Can be divided evenly by the alignment.
UIntPtr size	Size in bytes of memory to be allocated.
UIntPtr alignment	The alignment.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.4.2.4.2 NCore.AlignedReAlloc Method (IntPtr, UIntPtr, UIntPtr, UIntPtr)

Reallocate memory blocks, represented by a pointer. A sum of the pointer and an offset can be divided evenly by the alignment.

C#

```
public static void AlignedReAlloc(ref IntPtr pBlock, UIntPtr size, UIntPtr alignment,
UIntPtr offset);
```

Parameters

Parameters	Description
ref IntPtr pBlock	A representation of a pointer (IntPtr type) to the allocated memory block. Its sum with the offset an be divided evenly by the alignment.
UIntPtr size	Size in bytes of memory to be allocated.
UIntPtr alignment	The alignment.
UIntPtr offset	The offset.

7.2.1.1.4.2.4.3 NCore.AlignedReAlloc Method (IntPtr, int, int)

Reallocate memory blocks, represented by a pointer that can be divided evenly by the alignment.

C#

```
public static void AlignedReAlloc(ref IntPtr pBlock, int size, int alignment);
```

Parameters

Parameters	Description
ref IntPtr pBlock	A representation of a pointer (IntPtr type) to the allocated memory block. Can be divided evenly by the alignment.
int size	Size in bytes of memory to be allocated.
int alignment	The alignment.

7.2.1.1.4.2.4.4 NCore.AlignedReAlloc Method (IntPtr, int, int, int)

Reallocate memory blocks, represented by a pointer. A sum of the pointer and an offset can be divided evenly by the alignment.

C#

```
public static void AlignedReAlloc(ref IntPtr pBlock, int size, int alignment, int offset);
```

Parameters

Parameters	Description
ref IntPtr pBlock	A representation of a pointer (IntPtr type) to the allocated memory block. Its sum with the offset an be divided evenly by the alignment.
int size	Size in bytes of memory to be allocated.
int alignment	The alignment.
int offset	The offset.

7.2.1.1.4.2.4.5 NCore.AlignedReAlloc Method (IntPtr, long, long)

Reallocate memory blocks, represented by a pointer that can be divided evenly by the alignment.

C#

```
public static void AlignedReAlloc(ref IntPtr pBlock, long size, long alignment);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
ref IntPtr pBlock	A representation of a pointer (IntPtr type) to the allocated memory block. Can be divided evenly by the alignment.
long size	Size in bytes of memory to be allocated.
long alignment	The alignment.

7.2.1.1.4.2.4.6 NCore.AlignedReAlloc Method (IntPtr, long, long, long)

Reallocate memory blocks, represented by a pointer. A sum of the pointer and an offset can be divided evenly by the alignment.

C#

```
public static void AlignedReAlloc(ref IntPtr pBlock, long size, long alignment, long
offset);
```

Parameters

Parameters	Description
ref IntPtr pBlock	A representation of a pointer (IntPtr type) to the allocated memory block. Its sum with the offset an be divided evenly by the alignment.
long size	Size in bytes of memory to be allocated.
long alignment	The alignment.
long offset	The offset.

7.2.1.1.4.2.5 Alloc Method

7.2.1.1.4.2.5.1 NCore.Alloc Method (UIntPtr)

Allocates a block of memory.

C#

```
public static IntPtr Alloc(UIntPtr size);
```

Parameters

Parameters	Description
UIntPtr size	Size in bytes of memory to be allocated.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block.

7.2.1.1.4.2.5.2 NCore.Alloc Method (int)

Allocates a block of memory.

C#

```
public static IntPtr Alloc(int size);
```

Parameters

Parameters	Description
int size	Size in bytes of memory to be allocated.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.4.2.5.3 NCore.Alloc Method (long)

Allocates a block of memory.

C#

```
public static IntPtr Alloc(long size);
```

Parameters

Parameters	Description
long size	Size in bytes of memory to be allocated.

Returns

A representation of a pointer (IntPtr type) to the allocated memory block.

7.2.1.1.4.2.6 CAlloc Method

7.2.1.1.4.2.6.1 NCore.CAlloc Method (UIntPtr)

Allocates an array in memory with elements initialized to 0.

C#

```
public static IntPtr CAlloc(UIntPtr size);
```

Parameters

Parameters	Description
UIntPtr size	Size in bytes of memory to be allocated.

Returns

Pointer to allocated memory block.

7.2.1.1.4.2.6.2 NCore.CAlloc Method (int)

Allocates an array in memory with elements initialized to 0.

C#

```
public static IntPtr CAlloc(int size);
```

Parameters

Parameters	Description
int size	Size in bytes of memory to be allocated.

Returns

Pointer to allocated memory block.

7.2.1.1.4.2.6.3 NCore.CAlloc Method (long)

Allocates an array in memory with elements initialized to 0.

C#

```
public static IntPtr CAlloc(long size);
```

Parameters

Parameters	Description
long size	Size in bytes of memory to be allocated.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Pointer to allocated memory block.

7.2.1.1.4.2.7 Clear Method

7.2.1.1.4.2.7.1 NCore.Clear Method (IntPtr, UIntPtr)

Clears all elements in the specified array.

C#

```
public static void Clear(IntPtr pBlock, UIntPtr size);
```

Parameters

Parameters	Description
IntPtr pBlock	A pointer to memory block that contains an array.
UIntPtr size	The size in bytes of an array.

7.2.1.1.4.2.7.2 NCore.Clear Method (IntPtr, int)

Clears all elements in the specified array.

C#

```
public static void Clear(IntPtr pBlock, int size);
```

Parameters

Parameters	Description
IntPtr pBlock	A pointer to memory block that contains an array.
int size	The size in bytes of an array.

7.2.1.1.4.2.7.3 NCore.Clear Method (IntPtr, long)

Clears all elements in the specified array.

C#

```
public static void Clear(IntPtr pBlock, long size);
```

Parameters

Parameters	Description
IntPtr pBlock	A pointer to memory block that contains an array.
long size	The size in bytes of an array.

7.2.1.1.4.2.8 Compare Method

7.2.1.1.4.2.8.1 NCore.Compare Method (IntPtr, IntPtr, UIntPtr)

Compares two memory blocks.

C#

```
public static int Compare(IntPtr pBlock1, IntPtr pBlock2, UIntPtr size);
```

Parameters

Parameters	Description
IntPtr pBlock1	A pointer to the first memory block.
IntPtr pBlock2	A pointer to the second memory block.
UIntPtr size	Number of characters to be compared.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The value indicating the relationship between the memory blocks.

If return value is less than zero a block specified by pBlock1 is less than a block specified by pBlock2.

if return value is zero a block specified by pBlock1 is identical to a block specified by pBlock2.

If return value is greater than zero a block specified by pBlock1 is greater than a block specified by pBlock2.

7.2.1.1.4.2.8.2 NCore.Compare Method (IntPtr, IntPtr, int)

Compares two memory blocks.

C#

```
public static int Compare(IntPtr pBlock1, IntPtr pBlock2, int size);
```

Parameters

Parameters	Description
IntPtr pBlock1	A pointer to the first memory block.
IntPtr pBlock2	A pointer to the second memory block.
int size	Number of characters to be compared.

Returns

The value indicating the relationship between the memory blocks.

If return value is less than zero a block specified by pBlock1 is less than a block specified by pBlock2.

if return value is zero a block specified by pBlock1 is identical to a block specified by pBlock2.

If return value is greater than zero a block specified by pBlock1 is greater than a block specified by pBlock2.

7.2.1.1.4.2.8.3 NCore.Compare Method (IntPtr, IntPtr, long)

Compares two memory blocks.

C#

```
public static int Compare(IntPtr pBlock1, IntPtr pBlock2, long size);
```

Parameters

Parameters	Description
IntPtr pBlock1	A pointer to the first memory block.
IntPtr pBlock2	A pointer to the second memory block.
long size	Number of characters to be compared.

Returns

The value indicating the relationship between the memory blocks.

If return value is less than zero a block specified by pBlock1 is less than a block specified by pBlock2.

if return value is zero a block specified by pBlock1 is identical to a block specified by pBlock2.

If return value is greater than zero a block specified by pBlock1 is greater than a block specified by pBlock2.

7.2.1.1.4.2.9 Copy Method

7.2.1.1.4.2.9.1 NCore.Copy Method (IntPtr, IntPtr, UIntPtr)

Copies data from one memory block to other.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static void Copy(IntPtr pDstBlock, IntPtr pSrcBlock, UIntPtr size);
```

Parameters

Parameters	Description
IntPtr pDstBlock	A pointer to destination memory block (a memory block where the data from source memory block will be copied).
IntPtr pSrcBlock	A pointer to memory block to be copied to another memory block.
UIntPtr size	The size of memory block to be copied (the size is in bytes).

7.2.1.1.4.2.9.2 NCore.Copy Method (IntPtr, IntPtr, int)

Copies data from one memory block to other.

C#

```
public static void Copy(IntPtr pDstBlock, IntPtr pSrcBlock, int size);
```

Parameters

Parameters	Description
IntPtr pDstBlock	A pointer to destination memory block (a memory block where the data from source memory block will be copied).
IntPtr pSrcBlock	A pointer to memory block to be copied to another memory block.
int size	The size of memory block to be copied (the size is in bytes).

7.2.1.1.4.2.9.3 NCore.Copy Method (IntPtr, IntPtr, long)

Copies data from one memory block to other.

C#

```
public static void Copy(IntPtr pDstBlock, IntPtr pSrcBlock, long size);
```

Parameters

Parameters	Description
IntPtr pDstBlock	A pointer to destination memory block (a memory block where the data from source memory block will be copied).
IntPtr pSrcBlock	A pointer to memory block to be copied to another memory block.
long size	The size of memory block to be copied (the size is in bytes).

7.2.1.1.4.2.10 Fill Method

7.2.1.1.4.2.10.1 NCore.Fill Method (IntPtr, byte, UIntPtr)

Sets memory block to a specified value.

C#

```
public static void Fill(IntPtr pBlock, byte value, UIntPtr size);
```

Parameters

Parameters	Description
IntPtr pBlock	A pointer to a destination memory block that contains data.
byte value	A value to set.
UIntPtr size	The size in bytes of memory block to be set.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.4.2.10.2 NCore.Fill Method (IntPtr, byte, int)

Sets memory block to a specified value.

C#

```
public static void Fill(IntPtr pBlock, byte value, int size);
```

Parameters

Parameters	Description
IntPtr pBlock	A pointer to a destination memory block that contains data.
byte value	A value to set.
int size	The size in bytes of memory block to be set.

7.2.1.1.4.2.10.3 NCore.Fill Method (IntPtr, byte, long)

Sets memory block to a specified value.

C#

```
public static void Fill(IntPtr pBlock, byte value, long size);
```

Parameters

Parameters	Description
IntPtr pBlock	A pointer to a destination memory block that contains data.
byte value	A value to set.
long size	The size in bytes of memory block to be set.

7.2.1.1.4.2.11 Free Method

7.2.1.1.4.2.11.1 NCore.Free Method (IntPtr)

Frees a specified memory block.

C#

```
public static void Free(IntPtr pBlock);
```

Parameters

Parameters	Description
IntPtr pBlock	A pointer to previously allocated memory block to be freed.

Remarks

The Free method frees a memory block that was previously allocated by a call to Alloc [1528], CAlloc [1529] or ReAlloc [1535] methods. The number of freed bytes is equivalent to the number of bytes requested when the block was allocated (or reallocated, in the case of ReAlloc [1535]).

7.2.1.1.4.2.11.2 NCore.Free Method (IntPtr, NMemoryType)

Frees a specified memory block.

C#

```
public static void Free(IntPtr pBlock, NMemoryType memoryType);
```

Parameters

Parameters	Description
IntPtr pBlock	A pointer to previously allocated memory block to be freed.
NMemoryType memoryType	Type of the memory block to be freed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

The Free method frees a memory block that was previously allocated by a call to Alloc [1528], CAlloc [1529] or ReAlloc [1535] methods. The number of freed bytes is equivalent to the number of bytes requested when the block was allocated (or reallocated, in the case of ReAlloc [1535]).

7.2.1.1.4.2.12 Move Method

7.2.1.1.4.2.12.1 NCore.Move Method (IntPtr, IntPtr, UIntPtr)

Moves one memory block to another.

C#

```
public static void Move(IntPtr pDstBlock, IntPtr pSrcBlock, UIntPtr size);
```

Parameters

Parameters	Description
IntPtr pDstBlock	A pointer to destination memory block.
IntPtr pSrcBlock	A pointer to source memory block.
UIntPtr size	Size in bytes to be copied.

Remarks

Copies size bytes from pDstBlock to pSrcBlock.

Make sure that the destination buffer is the same size or larger than the source buffer.

7.2.1.1.4.2.12.2 NCore.Move Method (IntPtr, IntPtr, int)

Moves one memory block to another.

C#

```
public static void Move(IntPtr pDstBlock, IntPtr pSrcBlock, int size);
```

Parameters

Parameters	Description
IntPtr pDstBlock	A pointer to destination memory block.
IntPtr pSrcBlock	A pointer to source memory block.
int size	Size in bytes to be copied.

Remarks

Copies size bytes from pDstBlock to pSrcBlock.

Make sure that the destination buffer is the same size or larger than the source buffer.

7.2.1.1.4.2.12.3 NCore.Move Method (IntPtr, IntPtr, long)

Moves one memory block to another.

C#

```
public static void Move(IntPtr pDstBlock, IntPtr pSrcBlock, long size);
```

Parameters

Parameters	Description
IntPtr pDstBlock	A pointer to destination memory block.
IntPtr pSrcBlock	A pointer to source memory block.
long size	Size in bytes to be copied.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

Copies size bytes from pDstBlock to pSrcBlock.

Make sure that the destination buffer is the same size or larger than the source buffer.

7.2.1.1.4.2.13 ReAlloc Method

7.2.1.1.4.2.13.1 NCore.ReAlloc Method (IntPtr, UIntPtr)

Reallocate memory blocks.

C#

```
public static void ReAlloc(ref IntPtr pBlock, UIntPtr size);
```

Parameters

Parameters	Description
ref IntPtr pBlock	Pointer to memory block to be allocated.
UIntPtr size	Size in bytes of new memory block.

7.2.1.1.4.2.13.2 NCore.ReAlloc Method (IntPtr, int)

Reallocate memory blocks.

C#

```
public static void ReAlloc(ref IntPtr pBlock, int size);
```

Parameters

Parameters	Description
ref IntPtr pBlock	Pointer to memory block to be allocated.
int size	Size in bytes of new memory block.

7.2.1.1.4.2.13.3 NCore.ReAlloc Method (IntPtr, long)

Reallocate memory blocks.

C#

```
public static void ReAlloc(ref IntPtr pBlock, long size);
```

Parameters

Parameters	Description
ref IntPtr pBlock	Pointer to memory block to be allocated.
long size	Size in bytes of new memory block.

7.2.1.1.4.2.14 NCore.Shutdown Method

Disables sends and receives on a socket.

C#

```
public static void Shutdown();
```

7.2.1.1.4.2.15 NCore.ShutdownThread Method

Disables sends and receives on a thread.

C#

```
public static void ShutdownThread();
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.4.3 NCore Properties

7.2.1.1.4.3.1 NCore.NativeModule Property

Defines the native module of the object.

C#

```
public static NModule NativeModule;
```

7.2.1.1.4.4 NCore Events

7.2.1.1.4.4.1 NCore.ErrorSuppressed Event

Occurs when an error is supressed.

C#

```
public static event EventHandler<ErrorSuppressedEventArgs> ErrorSuppressed;
```

7.2.1.1.5 NDisposable Class

Provides a method to release allocated resources.

C#

```
public abstract class NDisposable : IDisposable;
```

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NDisposable Properties

	Name	Description
	IsDisposed [1536]	Gets whether allocated resources were released previously.

7.2.1.1.5.1 NDisposable Methods

7.2.1.1.5.1.1 NDisposable.Dispose Method

Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

C#

```
public virtual void Dispose();
```

Remarks

This method is used to close or release unmanaged resources. By convention, this method is used for all tasks associated with freeing resources held by an object.

7.2.1.1.5.2 NDisposable Properties

7.2.1.1.5.2.1 NDisposable.IsDisposed Property

Gets whether allocated resources were released previously.

C#

```
public virtual bool IsDisposed;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.6 NExpandableObject Class

Provides functionality for retrieving information about specified object.

C#

```
public class NExpandableObject : NObject;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	FromObject [1554]	Creates a new NObject [1549] from the specified object.
	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NExpandableObject Class

	Name	Description
S	NativeType [1539]	Defines the native type of the object.
	Properties [1539]	Gets the property bag of the expandable object.

7.2.1.1.6.1 NExpandableObject Properties

7.2.1.1.6.1.1 NExpandableObject.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.1.1.6.1.2 NExpandableObject.Properties Property

Gets the property bag of the expandable object.

C#

```
public NPropertyBag Properties;
```

7.2.1.1.7 NIndexPair Structure

Represents pair of indexes.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NIndexPair : IFormattable {
}
```

Methods

	Name	Description
	NIndexPair [1540]	Initializes a new instance of the NIndexPair structure.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NIndexPair Methods

	Name	Description
	ToString [1540]	Retrieves a string representation of the object.

NIndexPair Properties

	Name	Description
	Index1 [1541]	Gets or sets first index of this NIndexPair.
	Index2 [1541]	Gets or sets second index of this NIndexPair.
S	NativeType [1541]	Defines the native type of the type.

7.2.1.1.7.1 NIndexPair.NIndexPair Constructor

Initializes a new instance of the NIndexPair structure.

C#

```
public NIndexPair(int index1, int index2);
```

Parameters

Parameters	Description
int index1	First index of this NIndexPair.
int index2	Second index of this NIndexPair.

7.2.1.1.7.2 NIndexPair Methods

7.2.1.1.7.2.1 ToString Method

7.2.1.1.7.2.1.1 NIndexPair.ToString Method ()

Retrieves a string representation of the object.

C#

```
public override string ToString();
```

Returns

A string representation of an object.

7.2.1.1.7.2.1.2 NIndexPair.ToString Method (string)

Retrieves a string representation of the object.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	Reserved.

Returns

A string representation of an object.

7.2.1.1.7.3 NIndexPair Properties

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.7.3.1 NIndexPair.Index1 Property

Gets or sets first index of this NIndexPair [1539].

C#

```
public int Index1;
```

7.2.1.1.7.3.2 NIndexPair.Index2 Property

Gets or sets second index of this NIndexPair [1539].

C#

```
public int Index2;
```

7.2.1.1.7.3.3 NIndexPair.NativeType Property

Defines the native type of the type.

C#

```
public static NType NativeType;
```

7.2.1.1.8 NInfo Class

For internal use only.

C#

```
public abstract class NInfo : NDisposable;
```

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NDisposable Properties

	Name	Description
	IsDisposed [1536]	Gets whether allocated resources were released previously.

7.2.1.1.9 NInfoObject Class

For internal use only.

C#

```
public abstract class NInfoObject : NDisposable;
```

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NDisposable Properties

	Name	Description
	IsDisposed [1536]	Gets whether allocated resources were released previously.

7.2.1.1.10 NLibraryInfo Class

Provides methods used to retrieve library information.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public sealed class NLibraryInfo;
```

NLibraryInfo Methods

	Name	Description
S	GetString [1542]	Retrieves a string from an array of chars.
S	Retrieve [1542]	Retrieves information about this library.

NLibraryInfo Properties

	Name	Description
	Activated [1542]	Checks if the library is registered.
	Company [1543]	Retrieves company name.
	Copyright [1543]	Retrieves copyright notice of this library.
	Product [1543]	Retrieves product name.
	Title [1543]	Retrieves library's title.
	Version [1543]	Retrieves library version number.

7.2.1.1.10.1 NLibraryInfo Methods

7.2.1.1.10.1.1 NLibraryInfo.GetString Method

Retrieves a string from an array of chars.

C#

```
public static string GetString(char[] chars);
```

Parameters

Parameters	Description
char[] chars	An array of chars that form a string.

Returns

A string value.

7.2.1.1.10.1.2 NLibraryInfo.Retrieve Method

Retrieves information about this library.

C#

```
public static NLibraryInfo Retrieve(NLibraryGetInfo getInfo);
```

Parameters

Parameters	Description
NLibraryGetInfo getInfo	NLibraryGetInfo [1654] object.

Returns

NLibraryInfo [1541] object which contains information about this library.

7.2.1.1.10.2 NLibraryInfo Properties

7.2.1.1.10.2.1 NLibraryInfo.Activated Property

Checks if the library is registered.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public string Activated;
```

7.2.1.1.10.2.2 NLibraryInfo.Company Property

Retrieves company name.

C#

```
public string Company;
```

7.2.1.1.10.2.3 NLibraryInfo.Copyright Property

Retrieves copyright notice of this library.

C#

```
public string Copyright;
```

7.2.1.1.10.2.4 NLibraryInfo.Product Property

Retrieves product name.

C#

```
public string Product;
```

7.2.1.1.10.2.5 NLibraryInfo.Title Property

Retrieves library's title.

C#

```
public string Title;
```

7.2.1.1.10.2.6 NLibraryInfo.Version Property

Retrieves library version number.

C#

```
public Version Version;
```

7.2.1.1.11 NModule Class

Performs reflection on a module.

C#

```
public class NModule : NObject;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

Methods

NModule Class

	Name	Description
	NModule [1546]	Initializes a new instance of NModule class.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	FromObject [1554]	Creates a new NObject [1549] from the specified object.
	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NModule Class

	Name	Description
	CheckInit [1546]	Checks whether the current module is initialized and throws and exception if it is not.
	GetDependences [1546]	Retrieves dependencies of the module.
	GetLoadedModules [1547]	Gets all the loaded modules that are part of this assembly.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NModule Class

	Name	Description
	Activated [1547]	Checks if the module is registered.
	Company [1547]	Retrieves company name.
	Copyright [1547]	Retrieves copyright notice of this module.
	Name [1547]	Retrieves module name.
S	NativeType [1547]	Defines the native type of the object.
S	NModuleOptionsNativeType [1548]	Defines the native type of the module options.
	Options [1548]	Retrieves module options.
	Product [1548]	Retrieves product name.
	Title [1548]	Retrieves module's title.
	Version [1548]	Retrieves module version number.
	VersionBuild [1548]	Retrieves the build of the current version.
	VersionMajor [1548]	Retrieves the major version value.
	VersionMinor [1548]	Retrieves the minor version value.
	VersionRevision [1548]	Retrieves revision of the current version.

7.2.1.1.11.1 NModule.NModule Constructor

Initializes a new instance of NModule class.

C#

```
public NModule();
```

7.2.1.1.11.2 NModule Methods

7.2.1.1.11.2.1 NModule.CheckInit Method

Checks whether the current module is initialized and throws and exception if it is not.

C#

```
public void CheckInit(bool lazy);
```

Parameters

Parameters	Description
bool lazy	true if the initialization is lazy; else false.

7.2.1.1.11.2.2 NModule.GetDependences Method

Retrieves dependencies of the module.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NModule[] GetDependences();
```

Returns

A pointer to an array of dependencies.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

7.2.1.1.11.2.3 NModule.GetLoadedModules Method

Gets all the loaded modules that are part of this assembly.

C#

```
public static NModule[] GetLoadedModules();
```

Returns

An array of modules.

Remarks

NObjectUnref [560] shall be called on returned objects when they are no longer used to prevent memory leaks.

7.2.1.1.11.3 NModule Properties

7.2.1.1.11.3.1 NModule.Activated Property

Checks if the module is registered.

C#

```
public string Activated;
```

7.2.1.1.11.3.2 NModule.Company Property

Retrieves company name.

C#

```
public string Company;
```

7.2.1.1.11.3.3 NModule.Copyright Property

Retrieves copyright notice of this module.

C#

```
public string Copyright;
```

7.2.1.1.11.3.4 NModule.Name Property

Retrieves module name.

C#

```
public string Name;
```

7.2.1.1.11.3.5 NModule.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.11.3.6 NModule.NModuleOptionsNativeType Property

Defines the native type of the module options.

C#

```
public static NType NModuleOptionsNativeType;
```

7.2.1.1.11.3.7 NModule.Options Property

Retrieves module options.

C#

```
public NModuleOptions Options;
```

7.2.1.1.11.3.8 NModule.Product Property

Retrieves product name.

C#

```
public string Product;
```

7.2.1.1.11.3.9 NModule.Title Property

Retrieves module's title.

C#

```
public string Title;
```

7.2.1.1.11.3.10 NModule.Version Property

Retrieves module version number.

C#

```
public Version Version;
```

7.2.1.1.11.3.11 NModule.VersionBuild Property

Retrieves the build of the current version.

C#

```
public int VersionBuild;
```

7.2.1.1.11.3.12 NModule.VersionMajor Property

Retrieves the major version value.

C#

```
public int VersionMajor;
```

7.2.1.1.11.3.13 NModule.VersionMinor Property

Retrieves the minor version value.

C#

```
public int VersionMinor;
```

7.2.1.1.11.3.14 NModule.VersionRevision Property

Retrieves revision of the current version.

C#

```
public int VersionRevision;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.12 NObject Class

Provides functionality for retrieving information about specified object.

C#

```
public class NObject : NDisposable, IFormattable, INotifyPropertyChanged;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	CopyParameters [1552]	Copies parameters values from one NObject object to another.
	CopyProperties [1552]	Copies proeprty values from one NObject object to another.
	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	FromObject [1554]	Creates a new NObject from the specified object.
	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject.
	ToString [1566]	Returns string representation of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

7.2.1.1.12.1 NObject Fields

7.2.1.1.12.1.1 NObject.FlagRefRet Field

This flag indicates that an object should be returned with incremented ref count.

C#

```
public const uint FlagRefRet = 0x00000010;
```

7.2.1.1.12.2 NObject Methods

7.2.1.1.12.2.1 NObject.Clone Method

Creates a copy of specified object.

C#

```
public object Clone();
```

Returns

Reference to cloned object.

7.2.1.1.12.2.2 NObject.CompareTo Method

Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.

C#

```
public int CompareTo(object obj);
```

Parameters

Parameters	Description
object obj	The other object.

Returns

Integer that indicates their relative position in the sort order.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.12.2.3 NObject.CopyParameters Method

Copies parameters values from one NObject [1549] object to another.

C#

```
public static void CopyParameters(NObject srcObj, NObject dstObj);
```

Parameters

Parameters	Description
NObject srcObj	Source NObject [1549] object which should be copied.
NObject dstObj	Destination NObject [1549] object where source object will be copied.

7.2.1.1.12.2.4 NObject.CopyProperties Method

Copies proeprty values from one NObject [1549] object to another.

C#

```
public static void CopyProperties(NObject srcObj, NObject dstObj);
```

Parameters

Parameters	Description
NObject srcObj	Source NObject [1549] object which should be copied.
NObject dstObj	Destination NObject [1549] object where source object will be copied.

7.2.1.1.12.2.5 NObject.DisposeArray Method

Frees an array of objects.

C#

```
public static void DisposeArray<T>(T[] objects) where T : NObject;
```

Parameters

Parameters	Description
T[] objects	The array to be freed.

7.2.1.1.12.2.6 NObject.Equals Method

Compares the objects.

C#

```
public override bool Equals(object obj);
```

Parameters

Parameters	Description
object obj	Other object to be compared with.

Returns

true if objects are equal.

7.2.1.1.12.2.7 FromHandle Method

7.2.1.1.12.2.7.1 NObject.FromHandle Method (HNObject)

Create C# object from native API objects handle.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static T FromHandle<T>(HNObject handle) where T : NObject;
```

Parameters

Parameters	Description
HNObject handle	Handle of the object in native API.

Returns

C# object associated with native API object.

7.2.1.1.12.2.7.2 NObject.FromHandle Method (HNObject, bool)

Create C# object from native API objects handle.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static T FromHandle<T>(HNObject handle, bool cache) where T : NObject;
```

Parameters

Parameters	Description
HNObject handle	Handle of the object in native API.
bool cache	Will the created C# object use cache.

Returns

C# object associated with native API object.

7.2.1.1.12.2.7.3 NObject.FromHandle Method (IntPtr)

Create C# object from native API objects handle.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static T FromHandle<T>(IntPtr hObject) where T : NObject;
```

Parameters

Parameters	Description
IntPtr hObject	Handle of the object in native API.

Returns

C# object associated with native API object.

7.2.1.1.12.2.7.4 NObject.FromHandle Method (IntPtr, bool, bool)

Create C# object from native API objects handle.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static T FromHandle<T>(IntPtr hObject, bool ownsHandle, bool cache) where T :
NObject;
```

Parameters

Parameters	Description
IntPtr hObject	Handle of the object in native API.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

bool ownsHandle	Will the created C# object become the owner of the handle.
bool cache	Whether the object is created from a cache.

Returns

C# object associated with native API object.

7.2.1.1.12.2.8 NObject.FromObject Method

Creates a new NObject [1549] from the specified object.

C#

```
public static NObject FromObject(object obj);
```

Parameters

Parameters	Description
object obj	The original object.

Returns

The new NObject [1549].

7.2.1.1.12.2.9 GetArray Method

7.2.1.1.12.2.9.1 NObject.GetArray Method (GetArrayPlain)

Helper function for getting data from the native object.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static T[] GetArray<T>(GetArrayPlain getArray) where T : struct;
```

Parameters

Parameters	Description
GetArrayPlain getArray	The getter function of the native object.

Type Parameters

Type Parameters	Description
T	Type of the array's element.

Returns

The array of elements returned by delegate specified in getArray argument.

7.2.1.1.12.2.9.2 NObject.GetArray Method (GetOutArrayPlain)

Helper function for getting data from the native object.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static T[] GetArray<T>(GetOutArrayPlain getArray) where T : struct;
```

Parameters

Parameters	Description
GetOutArrayPlain getArray	The getter function of the native object.

Returns

The array of elements returned by delegate specified in getArray argument.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.12.2.10 NObject.GetHashCode Method

Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.

C#

```
public override int GetHashCode();
```

Returns

Hash code for the object.

7.2.1.1.12.2.11 NObject.GetNativeType Method

Retrieves native type of object.

C#

```
public NType GetNativeType();
```

Returns

NativeType [1568] value.

7.2.1.1.12.2.12 GetObject Method

7.2.1.1.12.2.12.1 NObject.GetObject Method (GetObjectPlain)

Helper function for getting data from the native object.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static T GetObject<T>(GetObjectPlain getObject) where T : NObject;
```

Parameters

Parameters	Description
GetObjectPlain getObject	The getter function of the native object.

Returns

The object returned by delegate specified in getObject argument.

7.2.1.1.12.2.12.2 NObject.GetObject Method (GetObjectPlain, bool)

Helper function for getting data from the native object.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static T GetObject<T>(GetObjectPlain getObject, bool cache) where T : NObject;
```

Parameters

Parameters	Description
GetObjectPlain getObject	The getter function of the native object.
bool cache	Whether a cache is used.

7.2.1.1.12.2.13 GetObjects Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.12.2.13.1 NObject.GetObjects Method (GetArrayPlain)

Helper function for getting data from the native object.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static T[] GetObjects<T>(GetArrayPlain getArray) where T : NObject;
```

Parameters

Parameters	Description
GetArrayPlain getArray	The getter function of the native object.

Returns

The array of elements returned by delegate specified in getArray argument.

7.2.1.1.12.2.13.2 NObject.GetObjects Method (GetArrayPlain, bool, bool)

Helper function for getting data from the native object.

This method supports the .NET Framework infrastructure and is not intended to be used directly.

C#

```
public static T[] GetObjects<T>(GetArrayPlain getArray, bool ownsHandles, bool cache) where
T : NObject;
```

Parameters

Parameters	Description
GetArrayPlain getArray	The getter function of the native object.
bool ownsHandles	Whether the objects own handles.
bool cache	Whether a cache is used.

Returns

The array of objects returned by delegate specified in getObjects argument.

7.2.1.1.12.2.13.3 NObject.GetObjects Method (GetOutArrayPlain)

Helper function for getting data from the native object.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static T[] GetObjects<T>(GetOutArrayPlain getArray) where T : NObject;
```

Parameters

Parameters	Description
GetOutArrayPlain getArray	The getter function of the native object.

Returns

The array of objects returned by delegate specified in getObjects argument.

7.2.1.1.12.2.13.4 NObject.GetObjects Method (GetOutArrayPlain, bool, bool)

Helper function for getting data from the native object.

This method supports the .NET Framework infrastructure and is not intended to be used directly.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static T[] GetObjects<T>(GetOutArrayPlain getArray, bool ownsHandles, bool cache)
where T : NObject;
```

Parameters

Parameters	Description
GetOutArrayPlain getArray	The getter function of the native object.
bool ownsHandles	Whether the objects own handles.
bool cache	Whether a cache is used.

Returns

The array of objects returned by delegate specified in getObjects argument.

7.2.1.1.12.2.14 GetParameter Method

7.2.1.1.12.2.14.1 NObject.GetParameter Method (ushort)

Gets value of parameter by parameter id.

C#

```
public T GetParameter<T>(ushort parameterId);
```

Parameters

Parameters	Description
ushort parameterId	The parameter identifier.

Type Parameters

Type Parameters	Description
T	Generic parameter type.

Returns

The parameter value.

Remarks

- NObject [1549] doesn't currently have any configurable parameters. The method can be used to change behavior of derived classes. The list of supported parameters can be found in list of the fields in API reference of appropriate class. Note the naming convention, that names of parameter fields start with "Parameter".
- The use of this method is not recommended. Instead access the parameters in more natural to C# way, by using properties of appropriate class.

7.2.1.1.12.2.14.2 NObject.GetParameter Method (ushort, ushort)

Gets value of parameter by parameter id.

C#

```
public T GetParameter<T>(ushort partId, ushort parameterId);
```

Parameters

Parameters	Description
ushort partId	Specifies the part of the specified NObject [1549].
ushort parameterId	The parameter identifier.

Type Parameters

Type Parameters	Description
T	Generic parameter type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The parameter value.

Remarks

- NObject [1549] doesn't currently have any configurable parameters. The method can be used to change behavior of derived classes. The list of supported parameters can be found in list of the fields in API reference of appropriate class. Note the naming convention, that names of parameter fields start with "Parameter".
- The use of this method is not recommended. Instead access the parameters in more natural to C# way, by using properties of appropriate class.

7.2.1.1.12.2.15 NObject.GetProperty Method

Retrieves the specified property of the current object.

C#

```
public T GetProperty<T>(string name);
```

Parameters

Parameters	Description
string name	Name of the property.

Returns

The specified property.

7.2.1.1.12.2.16 GetSize Method

7.2.1.1.12.2.16.1 NObject.GetSize Method ()

Returns the size of the current object.

C#

```
public int GetSize();
```

Returns

The object's size.

7.2.1.1.12.2.16.2 NObject.GetSize Method (uint)

Returns the size of the current object.

C#

```
public int GetSize(uint flags);
```

Parameters

Parameters	Description
uint flags	Object's flags.

Returns

The object's size.

7.2.1.1.12.2.17 GetSizeMany Method

7.2.1.1.12.2.17.1 NObject.GetSizeMany Method (T[])

Returns the size of objects in the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static int GetSizeMany<T>(T[] objects) where T : NObject;
```

Parameters

Parameters	Description
T[] objects	An array of objects.

Returns

Objects' size.

7.2.1.1.12.2.17.2 NObject.GetSizeMany Method (T[], uint)

Returns the size of objects in the array.

C#

```
public static int GetSizeMany<T>(T[] objects, uint flags) where T: NObject;
```

Parameters

Parameters	Description
T[] objects	An array of objects.
uint flags	Objects' flags.

Returns

Objects' size.

7.2.1.1.12.2.18 NObject.GetString Method

Helper function for getting data from the native object.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static string GetString(GetStringPlainN getString);
```

Parameters

Parameters	Description
GetStringPlainN getString	The getter function of the native object.

Returns

The text returned by delegate specified in getString argument.

7.2.1.1.12.2.19 Invoke Method

7.2.1.1.12.2.19.1 NObject.Invoke Method (EventHandler<T>, NObject, T)

Invoke Event.

C#

```
public static void Invoke<T>(EventHandler<T> eventHandler, NObject sender, T ea) where T :
EventArgs;
```

7.2.1.1.12.2.19.2 NObject.Invoke Method (PropertyChangedEventHandler, NObject, PropertyChangedEventArgs)

Invoke Event.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static void Invoke(PropertyChangedEventHandler eventHandler, NObject sender,
PropertyChangedEventArgs ea);
```

7.2.1.1.12.2.20 NObject.Ref Method

Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.

C#

```
public static void Ref(IntPtr hObject);
```

Parameters

Parameters	Description
IntPtr hObject	Handle to an object to increment its reference count.

7.2.1.1.12.2.21 NObject.Reset Method

Resets all NObject [1549] parameters to default values.

C#

```
public void Reset();
```

7.2.1.1.12.2.22 NObject.ResetProperty Method

Resets the value of the specified property.

C#

```
public void ResetProperty(string name);
```

Parameters

Parameters	Description
string name	Name of the property to be reset.

7.2.1.1.12.2.23 Save Method

7.2.1.1.12.2.23.1 NObject.Save Method ()

Saves the current object into a buffer.

C#

```
public NBuffer Save();
```

Returns

The destination NBuffer [1825].

7.2.1.1.12.2.23.2 NObject.Save Method (NBuffer)

Saves the current object into a buffer.

C#

```
public int Save(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	The destination buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The buffer size.

7.2.1.1.12.2.23.3 NObject.Save Method (NBuffer, uint)

Saves the current object into a buffer.

C#

```
public int Save(NBuffer buffer, uint flags);
```

Parameters

Parameters	Description
NBuffer buffer	The destination buffer.
uint flags	Must be zero.

Returns

The buffer size.

7.2.1.1.12.2.23.4 NObject.Save Method (Stream)

Saves the current object into a stream.

C#

```
public void Save(Stream stream);
```

Parameters

Parameters	Description
Stream stream	The destination stream.

7.2.1.1.12.2.23.5 NObject.Save Method (Stream, uint)

Saves the current object into a stream.

C#

```
public void Save(Stream stream, uint flags);
```

Parameters

Parameters	Description
Stream stream	The destination stream.
uint flags	Must be zero.

7.2.1.1.12.2.23.6 NObject.Save Method (uint)

Saves the current object into a buffer.

C#

```
public NBuffer Save(uint flags);
```

Parameters

Parameters	Description
uint flags	Must be zero.

Returns

The destination NBuffer [1825].

7.2.1.1.12.2.24 SaveMany Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.12.2.24.1 NObject.SaveMany Method (T[])

Saves an array of objects into a buffer.

C#

```
public static NBuffer SaveMany<T>(T[] objects) where T : NObject;
```

Parameters

Parameters	Description
T[] objects	An object array.

Returns

The destination buufer.

7.2.1.1.12.2.24.2 NObject.SaveMany Method (T[], NBuffer)

Saves an array of objects into a buffer.

C#

```
public static int SaveMany<T>(T[] objects, NBuffer buffer) where T : NObject;
```

Parameters

Parameters	Description
T[] objects	An object array.
NBuffer buffer	Destination buffer.

Returns

Buffer size.

7.2.1.1.12.2.24.3 NObject.SaveMany Method (T[], NBuffer, uint)

Saves an array of objects into a buffer.

C#

```
public static int SaveMany<T>(T[] objects, NBuffer buffer, uint flags) where T : NObject;
```

Parameters

Parameters	Description
T[] objects	An object array.
NBuffer buffer	Destination buffer.
uint flags	Must be zero.

Returns

Buffer size.

7.2.1.1.12.2.24.4 NObject.SaveMany Method (T[], uint)

Saves an array of objects into a buffer.

C#

```
public static NBuffer SaveMany<T>(T[] objects, uint flags) where T : NObject;
```

Parameters

Parameters	Description
T[] objects	An object array.
uint flags	Must be zero.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The destination buffer.

7.2.1.1.12.2.25 NObject.SetArray Method

Helper function for setting data of the native object.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static void SetArray<T>(SetArrayPlain setArray, T[] value) where T : struct;
```

Parameters

Parameters	Description
SetArrayPlain setArray	The setter function of the native object.
T[] value	The new value to set.

Type Parameters

Type Parameters	Description
T	Type of the array's element.

7.2.1.1.12.2.26 NObject.SetObject Method

Helper function for setting data of the native object.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static void SetObject<T>(SetObjectPlain setObject, T value) where T : NObject;
```

Parameters

Parameters	Description
SetObjectPlain setObject	The setter function of the native object.
T value	The new value to set.

7.2.1.1.12.2.27 SetParameter Method

7.2.1.1.12.2.27.1 NObject.SetParameter Method (ushort, T)

Sets value of parameter by parameter id.

C#

```
public void SetParameter<T>(ushort parameterId, T value);
```

Parameters

Parameters	Description
ushort parameterId	The parameter identifier.
T value	The new value.

Type Parameters

Type Parameters	Description
T	Generic parameter type.

Remarks

- NObject [1549] doesn't currently has any configurable parameters. The method can be used to change behavior of derived classes. The list of supported parameters can be found in list of the fields in API reference of appropriate class.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Note the naming convention, that names of parameter fields start with "Parameter".

- The use of this method is not recommended. Instead access the parameters in more natural to C# way, by using properties of appropriate class.

7.2.1.1.12.2.27.2 NObject.SetParameter Method (ushort, object)

Sets value of parameter by parameter id.

C#

```
public void SetParameter(ushort parameterId, object value);
```

Parameters

Parameters	Description
ushort parameterId	The parameter identifier.
object value	The parameter value.

Remarks

- NObject [1549] doesn't currently have any configurable parameters. The method can be used to change behavior of derived classes. The list of supported parameters can be found in list of the fields in API reference of appropriate class. Note the naming convention, that names of parameter fields start with "Parameter".
- The use of this method is not recommended. Instead access the parameters in more natural to C# way, by using properties of appropriate class.

7.2.1.1.12.2.27.3 NObject.SetParameter Method (ushort, ushort, T)

Sets value of parameter by parameter id.

C#

```
public void SetParameter<T>(ushort partId, ushort parameterId, T value);
```

Parameters

Parameters	Description
ushort partId	Specifies the part of the specified NObject [1549].
ushort parameterId	The parameter identifier.
T value	The parameter value.

Type Parameters

Type Parameters	Description
T	Generic parameter type.

Remarks

- NObject [1549] doesn't currently have any configurable parameters. The method can be used to change behavior of derived classes. The list of supported parameters can be found in list of the fields in API reference of appropriate class. Note the naming convention, that names of parameter fields start with "Parameter".
- The use of this method is not recommended. Instead access the parameters in more natural to C# way, by using properties of appropriate class.

7.2.1.1.12.2.27.4 NObject.SetParameter Method (ushort, ushort, object)

Sets value of parameter by parameter id.

C#

```
public void SetParameter(ushort partId, ushort parameterId, object value);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
ushort partId	Specifies the part of the specified NObject [1549].
ushort parameterId	The parameter identifier.
object value	The parameter value.

Remarks

- NObject [1549] doesn't currently have any configurable parameters. The method can be used to change behavior of derived classes. The list of supported parameters can be found in list of the fields in API reference of appropriate class. Note the naming convention, that names of parameter fields start with "Parameter".
- The use of this method is not recommended. Instead access the parameters in more natural to C# way, by using properties of appropriate class.

7.2.1.1.12.2.28 SetProperty Method

7.2.1.1.12.2.28.1 NObject.SetProperty Method (string, T)

Sets the value of the specified property.

C#

```
public void SetProperty<T>(string name, T value);
```

Parameters

Parameters	Description
string name	Name of the property to be set.
T value	The new value of the property.

Type Parameters

Type Parameters	Description
T	Type of the property.

7.2.1.1.12.2.28.2 NObject.SetProperty Method (string, object)

Sets the value of the specified property.

C#

```
public void SetProperty(string name, object value);
```

Parameters

Parameters	Description
string name	Name of the property to be set.
object value	The new value of the property.

7.2.1.1.12.2.29 NObject.SetString Method

Helper function for setting data of the native object.

This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.

C#

```
public static void SetString(SetStringPlainN setString, string value);
```

Parameters

Parameters	Description
SetStringPlainN setString	The setter function of the native object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

string value	The new value to set.

7.2.1.1.12.2.30 NObject.ToHandle Method

Gets the representation of native API handle corresponding to the C# object.

C#

```
public static HNObject ToHandle(NObject obj);
```

Parameters

Parameters	Description
NObject obj	C# level object.

Returns

C# wrapper of native API handle.

7.2.1.1.12.2.31 NObject.ToObject Method

Creates a new object from the specified NObject [1549].

C#

```
public static object ToObject(NObject obj);
```

Parameters

Parameters	Description
NObject obj	The original NObject [1549].

Returns

The new object.

7.2.1.1.12.2.32 ToString Method

7.2.1.1.12.2.32.1 NObject.ToString Method ()

Returns string representation of the object.

C#

```
public sealed override string ToString();
```

Returns

String representation of the object.

7.2.1.1.12.2.32.2 NObject.ToString Method (string)

Retrieves a string representation of the object.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	Reserved.

Returns

A string representation of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.12.2.33 NObject.Unref Method

Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.

C#

```
public static void Unref(IntPtr hObject);
```

Parameters

Parameters	Description
IntPtr hObject	Handle to an object to decrement its reference count.

7.2.1.1.12.2.34 NObject.UnrefArray Method

Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

C#

```
public static void UnrefArray(IntPtr arhValues, int count);
```

Parameters

Parameters	Description
IntPtr arhValues	Handle to array of objects which reference count should be decremented.
int count	Objects in arhObjects count.

7.2.1.1.12.2.35 NObject.UnrefElements Method

Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

C#

```
public static void UnrefElements(IntPtr arhValues, int count);
```

Parameters

Parameters	Description
IntPtr arhValues	Handle to array of objects which reference count should be decremented.
int count	Objects in arhObjects count.

7.2.1.1.12.3 NObject Properties

7.2.1.1.12.3.1 NObject.Flags Property

Gets flags from object.

C#

```
public uint Flags;
```

Property value

Object flags.

7.2.1.1.12.3.2 NObject.IsDisposed Property

Gets whether allocated resources were released previously.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public sealed override bool IsDisposed;
```

7.2.1.1.12.3.3 NObject.NativeType Property

Neurotechnology framework type of the object.

C#

```
public static NType NativeType;
```

7.2.1.1.12.3.4 NObject.Owner Property

Gets owner of the object.

C#

```
public NObject Owner;
```

Property value

Object (owner) of NObject [1549] type.

7.2.1.1.12.4 NObject Delegates

7.2.1.1.12.4.1 NObject.EventInvoker Delegate

Creates a delegate to an event invoker function.

C#

```
public delegate void EventInvoker<T>(T ea) where T : EventArgs;
```

Parameters

Parameters	Description
ea	The event to be invoked.

7.2.1.1.12.4.2 NObject.FromHandlePlain Delegate

Creates a delegate to FromHandlePlain function.

C#

```
public delegate NObject FromHandlePlain(HNObject handle);
```

Parameters

Parameters	Description
handle	Object handle.

Returns

NObject [1549] value.

7.2.1.1.12.4.3 NObject.GetArrayPlain Delegate

Creates a delegate to a function, which gets a plain array.

C#

```
public delegate int GetArrayPlain(IntPtr arValues, int valuesLength);
```

Parameters

Parameters	Description
arValues	Pointer to an array of values.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

valuesLength	Length of values.

Returns

NResult [1007].

7.2.1.1.12.4.4 NObject.GetArrayWithHandle Delegate

Creates a delegate to a function, which gets an array with handle.

C#

```
public delegate int GetArrayWithHandle(HNObject handle, IntPtr arValues, int valuesLength);
```

Parameters

Parameters	Description
handle	Array handle.
arValues	Pointer to an array of values.
valuesLength	Length of values.

Returns

NResult [1007].

7.2.1.1.12.4.5 NObject.GetObjectPlain Delegate

Creates a delegate to a function which returns a plain object.

C#

```
public delegate int GetObjectPlain(out HNObject hValue);
```

Parameters

Parameters	Description
hValue	[out] Object value.

Returns

NResult [1007].

7.2.1.1.12.4.6 NObject.GetObjectWithHandle Delegate

Creates a delegate to a function, which returns an object with handle.

C#

```
public delegate int GetObjectWithHandle(HNObject handle, out HNObject hValue);
```

Parameters

Parameters	Description
handle	Object handle.
hValue	[out] Object value.

Returns

NResult [1007].

7.2.1.1.12.4.7 NObject.GetOutArrayPlain Delegate

Creates a delegate to a function, which gets a plain out array.

C#

```
public delegate int GetOutArrayPlain(out IntPtr arValues, out int count);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
arValues	[out] Pointer to an array of values.
count	[out] Number of elements in the array.

Returns

NResult [1007].

7.2.1.1.12.4.8 NObject.GetOutArrayWithHandle Delegate

Creates a delegate to a function, which gets an out array with a handle.

C#

```
public delegate int GetOutArrayWithHandle(HNObject handle, out IntPtr arValues, out int
count);
```

Parameters

Parameters	Description
handle	Array handle.
arValues	Pointer to an array of values.
count	[out] Number of elements in the array.

Returns

NResult [1007].

7.2.1.1.12.4.9 NObject.GetStringPlainN Delegate

Creates a delegate to a function, which gets a plain string.

C#

```
public delegate int GetStringPlainN(out HNString hValue);
```

Parameters

Parameters	Description
hValue	[out] String value.

Returns

NResult [1007].

7.2.1.1.12.4.10 NObject.GetStringWithHandleN Delegate

Creates a delegate to a function, which gets a string with handle.

C#

```
public delegate int GetStringWithHandleN(HNObject handle, out HNString hValue);
```

Parameters

Parameters	Description
handle	String handle.
hValue	[out] String value.

Returns

NResult [1007].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.12.4.11 NObject.SetArrayPlain Delegate

Creates a delegate to a function, which sets values of a plain array.

C#

```
public delegate int SetArrayPlain(IntPtr arValues, int valueCount);
```

Parameters

Parameters	Description
arValues	Pointer to an array of values.
valueCount	Number of values.

Returns

NResult [1007].

7.2.1.1.12.4.12 NObject.SetArrayWithHandle Delegate

Creates a delegate to a function, which sets values of an array with handle.

C#

```
public delegate int SetArrayWithHandle(HNObject handle, IntPtr arValues, int valueCount);
```

Parameters

Parameters	Description
handle	Array handle.
arValues	Pointer to an array of values.
valueCount	Number of values.

Returns

NResult [1007].

7.2.1.1.12.4.13 NObject.SetObjectPlain Delegate

Creates a delegate to a function, which sets a value of a plain object.

C#

```
public delegate int SetObjectPlain(HNObject hValue);
```

Parameters

Parameters	Description
hValue	Object value.

Returns

NResult [1007].

7.2.1.1.12.4.14 NObject.SetObjectWithHandle Delegate

Creates a delegate to a function, which sets a value of an object with handle.

C#

```
public delegate int SetObjectWithHandle(HNObject handle, HNObject hValue);
```

Parameters

Parameters	Description
handle	Object handle.
hValue	Object value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NResult [1007].

7.2.1.1.12.4.15 NObject.SetStringPlainN Delegate

Creates a delegate to a function, which sets a plain string.

C#

```
public delegate int SetStringPlainN(HNString value);
```

Parameters

Parameters	Description
value	String value.

Returns

NResult [1007].

7.2.1.1.12.4.16 NObject.SetStringWithHandleN Delegate

Creates a delegate to a function, which sets a string with handle.

C#

```
public delegate int SetStringWithHandleN(HNObject handle, HNString value);
```

Parameters

Parameters	Description
handle	String handle.
value	String value.

Returns

NResult [1007].

7.2.1.1.12.5 NObject Events

7.2.1.1.12.5.1 NObject.PropertyChanged Event

Adds or removes a "Property Changed" event.

C#

```
public event PropertyChangedEventHandler PropertyChanged;
```

7.2.1.1.13 NProcessorInfo Class

Provides functionality for getting processor information.

C#

```
public static class NProcessorInfo;
```

NProcessorInfo Methods

	Name	Description
S	GetModelInfo [1573]	Retrieves processor model information.

NProcessorInfo Properties

	Name	Description
S	Count [1573]	Retrieves the number of logical processors in the current group.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	Is3DNowSupported [1574]	Gets a boolean representing if processor supports 3DNow instructions set.
	IsLZCntSupported [1574]	Gets a boolean representing if processor supports LZCnt instruction.
	IsMmxSupported [1574]	Gets a boolean representing if processor supports MMX instructions set.
	IsNeonSupported [1574]	Gets a boolean representing if processor supports NEON instructions set.
	IsPopCntSupported [1574]	Gets a boolean representing if processor supports PopCnt instruction.
	IsSse2Supported [1574]	Gets a boolean representing if processor supports SSE2 instructions set.
	IsSse3Supported [1574]	Gets a boolean representing if processor supports SSE3 instructions set.
	IsSse41Supported [1574]	Gets a boolean representing if processor supports SSE 4.1 instructions set.
	IsSse42Supported [1574]	Gets a boolean representing if processor supports SSE 4.2 instructions set.
	IsSse4aSupported [1575]	Gets a boolean representing if processor supports SSE 4a instructions set.
	IsSse5Supported [1575]	Gets a boolean representing if processor supports SSE5 instructions set.
	IsSseSupported [1575]	Gets a boolean representing if processor supports SSE instructions.
	IsSsse3Supported [1575]	Gets a boolean representing if processor supports SSSE3 instructions set.
	ModelName [1575]	Gets a string representing processor model's name.
	NativeType [1575]	Defines the native type of the object.
	NProcessorVendorNativeType [1575]	Defines the native type of the processor vendor.
	Vendor [1575]	Gets current processor's vendor value in vendor's enumeration.
	VendorName [1575]	Gets a string representing processor vendor's name.

7.2.1.1.13.1 NProcessorInfo Methods

7.2.1.1.13.1.1 NProcessorInfo.GetModelInfo Method

Retrieves processor model information.

C#

```
public static void GetModelInfo(out int family, out int model, out int stepping);
```

Parameters

Parameters	Description
out int family	Integer reresenting processor's family.
out int model	Integer representing processor's model name.
out int stepping	Integer representing processor's stepping.

7.2.1.1.13.2 NProcessorInfo Properties

7.2.1.1.13.2.1 NProcessorInfo.Count Property

Retrieves the number of logical processors in the current group.

C#

```
public static int Count;
```

Property value

The number of logical processors in the current group.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.13.2.2 NProcessorInfo.Is3DNowSupported Property

Gets a boolean representing if processor supports 3DNow instructions set.

C#

```
public static bool Is3DNowSupported;
```

7.2.1.1.13.2.3 NProcessorInfo.IsLZCntSupported Property

Gets a boolean representing if processor supports LZCnt instruction.

C#

```
public static bool IsLZCntSupported;
```

7.2.1.1.13.2.4 NProcessorInfo.IsMmxSupported Property

Gets a boolean representing if processor supports MMX instructions set.

C#

```
public static bool IsMmxSupported;
```

7.2.1.1.13.2.5 NProcessorInfo.IsNeonSupported Property

Gets a boolean representing if processor supports NEON instructions set.

C#

```
public static bool IsNeonSupported;
```

7.2.1.1.13.2.6 NProcessorInfo.IsPopCntSupported Property

Gets a boolean representing if processor supports PopCnt instruction.

C#

```
public static bool IsPopCntSupported;
```

7.2.1.1.13.2.7 NProcessorInfo.IsSse2Supported Property

Gets a boolean representing if processor supports SSE2 instructions set.

C#

```
public static bool IsSse2Supported;
```

7.2.1.1.13.2.8 NProcessorInfo.IsSse3Supported Property

Gets a boolean representing if processor supports SSE3 instructions set.

C#

```
public static bool IsSse3Supported;
```

7.2.1.1.13.2.9 NProcessorInfo.IsSse41Supported Property

Gets a boolean representing if processor supports SSE 4.1 instructions set.

C#

```
public static bool IsSse41Supported;
```

7.2.1.1.13.2.10 NProcessorInfo.IsSse42Supported Property

Gets a boolean representing if processor supports SSE 4.2 instructions set.

C#

```
public static bool IsSse42Supported;
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.13.2.11 NProcessorInfo.IsSse4aSupported Property

Gets a boolean representing if processor supports SSE 4a instructions set.

C#

```
public static bool IsSse4aSupported;
```

7.2.1.1.13.2.12 NProcessorInfo.IsSse5Supported Property

Gets a boolean representing if processor supports SSE5 instructions set.

C#

```
public static bool IsSse5Supported;
```

7.2.1.1.13.2.13 NProcessorInfo.IsSseSupported Property

Gets a boolean representing if processor supports SSE instructions.

C#

```
public static bool IsSseSupported;
```

7.2.1.1.13.2.14 NProcessorInfo.IsSsse3Supported Property

Gets a boolean representing if processor supports SSSE3 instructions set.

C#

```
public static bool IsSsse3Supported;
```

7.2.1.1.13.2.15 NProcessorInfo.ModelName Property

Gets a string representing processor model's name.

C#

```
public static string ModelName;
```

7.2.1.1.13.2.16 NProcessorInfo.NativeType Property

Defines the native type of the object.

C#

```
public static NType NativeType;
```

7.2.1.1.13.2.17 NProcessorInfo.NProcessorVendorNativeType Property

Defines the native type of the processor vendor.

C#

```
public static NType NProcessorVendorNativeType;
```

7.2.1.1.13.2.18 NProcessorInfo.Vendor Property

Gets current processor's vendor value in vendor's enumeration.

C#

```
public static NProcessorVendor Vendor;
```

7.2.1.1.13.2.19 NProcessorInfo.VendorName Property

Gets a string representing processor vendor's name.

C#

```
public static string VendorName;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.14 NPropertyBag Class

Represents a property bag in which the object can persistently save its properties.

C#

```
public sealed class NPropertyBag : NObject, IDictionary<string, object>,
IList<KeyValuePair<string, object>>, ICollection<KeyValuePair<string, object>>, IEnumerable;
```

NPropertyBag Classes

NPropertyBag Class

	Name	Description
	KeyCollection [1579]	Represents the collection of key parts of the key value pair.
	ValueCollection [1580]	Represents the collection of value parts of the key value pair.

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Methods

NPropertyBag Class

	Name	Description
	NPropertyBag [1579]	Creates a new instance of NPropertyBag class.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.
S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.
S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NPropertyBag Class

	Name	Description
	Add [1580]	Adds a key value pair to the current property bag.
	AddRange [1580]	Adds a range of key value pair values to the current property bag.
	ApplyTo [1580]	Applies the current property bag to the specified object.
	Clear [1581]	Clears the current property bag.
	Contains [1581]	Checks whether the current property bag contains the specified key value pair.
	ContainsKey [1581]	Checks whether the property bag contains a property specified by key.
	CopyTo [1581]	Copies the current property bag into a key value pair array.
	GetEnumerator [1581]	Gets the enumerator for the current key value pair.
S	Parse [1582]	Converts the specified string to a property bag.
	Remove [1582]	Removes the specified key value pair from the current property bag.
	RemoveAt [1583]	Removes the property specified by index.
	this [1583]	Gets the specfied key vale pair.
	TryGetValue [1583]	Retrieves the value of a property specified by the key. Returns a Boolean value indicating whether the operation succeeded.
S	TryParse [1584]	Converts the specified string to a property bag. Returns a Boolean value indicating whether the conversion succeeded.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NPropertyBag Class

	Name	Description
	Count [1584]	Retrieves the number of properties in the property bag.
	IsReadOnly [1585]	Checks whether the property bag is read-only.
	Keys [1585]	Gets the collection of key values.
S	NativeType [1585]	Defines the native type of the object.
	Values [1585]	Gets the collection of values.

7.2.1.1.14.1 NPropertyBag Constructor

7.2.1.1.14.1.1 NPropertyBag.NPropertyBag Constructor ()

Creates a new instance of NPropertyBag class.

C#

```
public NPropertyBag();
```

7.2.1.1.14.1.2 NPropertyBag.NPropertyBag Constructor (IEnumerable<KeyValuePair<string, object>>)

Creates a new instance of NPropertyBag class.

C#

```
public NPropertyBag(IEnumerable<KeyValuePair<string, object>> values);
```

Parameters

Parameters	Description
IEnumerable<KeyValuePair<string, object>> values	Property values.

7.2.1.1.14.2 NPropertyBag Classes

7.2.1.1.14.2.1 NPropertyBag.KeyCollection Class

Represents the collection of key parts of the key value pair.

C#

```
public sealed class KeyCollection : StringReadOnlyCollection;
```

KeyCollection Methods

	Name	Description
	Contains [1579]	Checks wether the key collection contains the specified key.

7.2.1.1.14.2.1.1 KeyCollection Methods

7.2.1.1.14.2.1.1.1 NPropertyBag.KeyCollection.Contains Method

Checks wether the key collection contains the specified key.

C#

```
public override bool Contains(string value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
string value	Key string.

Returns

true if the collection contains the key; else false.

7.2.1.1.14.2.2 NPropertyBag.ValueCollection Class

Represents the collection of value parts of the key value pair.

C#

```
public sealed class ValueCollection : ObjectReadOnlyCollection;
```

7.2.1.1.14.3 NPropertyBag Methods

7.2.1.1.14.3.1 Add Method

7.2.1.1.14.3.1.1 NPropertyBag.Add Method (KeyValuePair<string, object>)

Adds a key value pair to the current property bag.

C#

```
public void Add(KeyValuePair<string, object> item);
```

Parameters

Parameters	Description
KeyValuePair<string, object> item	The key value pair to add.

7.2.1.1.14.3.1.2 NPropertyBag.Add Method (string, object)

Adds a new key value pair.

C#

```
public void Add(string key, object value);
```

Parameters

Parameters	Description
string key	Key part of the pair.
object value	Value part of the pair.

7.2.1.1.14.3.2 NPropertyBag.AddRange Method

Adds a range of key value pair values to the current property bag.

C#

```
public void AddRange(IEnumerable<KeyValuePair<string, object>> values);
```

Parameters

Parameters	Description
IEnumerable<KeyValuePair<string, object>> values	Key valye pair values to be added.

7.2.1.1.14.3.3 NPropertyBag.ApplyTo Method

Applies the current property bag to the specified object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public void ApplyTo(NObject @object);
```

Parameters

Parameters	Description
object	The object to apply the property bag to.

7.2.1.1.14.3.4 NPropertyBag.Clear Method

Clears the current property bag.

C#

```
public void Clear();
```

7.2.1.1.14.3.5 NPropertyBag.Contains Method

Checks whether the current property bag contains the specified key value pair.

C#

```
public bool Contains(KeyValuePair<string, object> item);
```

Parameters

Parameters	Description
KeyValuePair<string, object> item	The key value pair.

Returns

true if the property bag contains the key value pair; else false.

7.2.1.1.14.3.6 NPropertyBag.ContainsKey Method

Checks whether the property bag contains a property specified by key.

C#

```
public bool ContainsKey(string key);
```

Parameters

Parameters	Description
string key	Key of the property.

Returns

true if the property bag contains the property, else false.

7.2.1.1.14.3.7 NPropertyBag.CopyTo Method

Copies the current property bag into a key value pair array.

C#

```
public void CopyTo(KeyValuePair<string, object>[] array, int arrayIndex);
```

Parameters

Parameters	Description
KeyValuePair<string, object>[] array	The key value pair array.
int arrayIndex	Index in the array, at which the copying starts.

7.2.1.1.14.3.8 NPropertyBag.GetEnumerator Method

Gets the enumerator for the current key value pair.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public IEnumerator<KeyValuePair<string, object>> GetEnumerator();
```

Returns

The new enumerator.

7.2.1.1.14.3.9 Parse Method

7.2.1.1.14.3.9.1 NPropertyBag.Parse Method (string)

Converts the specified string to a property bag.

C#

```
public static NPropertyBag Parse(string value);
```

Parameters

Parameters	Description
string value	The string to parse.

Returns

The parsed property bag.

7.2.1.1.14.3.9.2 NPropertyBag.Parse Method (string, string)

Converts the specified string to a property bag.

C#

```
public static NPropertyBag Parse(string value, string format);
```

Parameters

Parameters	Description
string value	The string to parse.
string format	The format string.

Returns

The parsed property bag.

7.2.1.1.14.3.10 Remove Method

7.2.1.1.14.3.10.1 NPropertyBag.Remove Method (KeyValuePair<string, object>)

Removes the specified key value pair from the current property bag.

C#

```
public bool Remove(KeyValuePair<string, object> item);
```

Parameters

Parameters	Description
KeyValuePair<string, object> item	The key value pair to be removed.

Returns

true if the operation was successful, else false.

7.2.1.1.14.3.10.2 NPropertyBag.Remove Method (string)

Removes the property specified by key.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public bool Remove(string key);
```

Parameters

Parameters	Description
string key	Key of the property.

Returns

true if the operation was successful, else false.

7.2.1.1.14.3.11 NPropertyBag.RemoveAt Method

Removes the property specified by index.

C#

```
public void RemoveAt(int index);
```

Parameters

Parameters	Description
int index	Index of the property to be removed.

7.2.1.1.14.3.12 this Indexer

7.2.1.1.14.3.12.1 NPropertyBag.this Indexer (int)

Gets the specfied key vale pair.

C#

```
public KeyValuePair<string, object> this[int index];
```

Parameters

Parameters	Description
int index	Index of the key value pair.

Returns

The key value pair.

7.2.1.1.14.3.12.2 NPropertyBag.this Indexer (string)

Gets or sets the value specified by key.

C#

```
public object this[string key];
```

Parameters

Parameters	Description
string key	Key of the property.

Returns

The value.

7.2.1.1.14.3.13 NPropertyBag.TryGetValue Method

Retrieves the value of a property specified by the key. Returns a Boolean value indicating whether the operation succeeded.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public bool TryGetValue(string key, out object value);
```

Parameters

Parameters	Description
string key	Key to the property to be retrieved.
out object value	[out] The property.

Returns

true if the operation was successful, else false.

7.2.1.1.14.3.14 TryParse Method

7.2.1.1.14.3.14.1 NPropertyBag.TryParse Method (string, out NPropertyBag)

Converts the specified string to a property bag. Returns a Boolean value indicating whether the conversion succeeded.

C#

```
public static bool TryParse(string value, out NPropertyBag result);
```

Parameters

Parameters	Description
string value	The string to parse.
out NPropertyBag result	[out] The parsed property bag.

Returns

true if the operation was successful, else false.

7.2.1.1.14.3.14.2 NPropertyBag.TryParse Method (string, string, out NPropertyBag)

Converts the specified string to a property bag. Returns a Boolean value indicating whether the conversion succeeded.

C#

```
public static bool TryParse(string value, string format, out NPropertyBag result);
```

Parameters

Parameters	Description
string value	The string to parse.
string format	The format string.
out NPropertyBag result	[out] The parsed property bag.

Returns

true if the operation was successful, else false.

7.2.1.1.14.4 NPropertyBag Properties

7.2.1.1.14.4.1 NPropertyBag.Count Property

Retrieves the number of properties in the property bag.

C#

```
public int Count;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.14.4.2 NPropertyBag.IsReadOnly Property

Checks whether the property bag is read-only.

C#

```
public bool IsReadOnly;
```

7.2.1.1.14.4.3 NPropertyBag.Keys Property

Gets the collection of key values.

C#

```
public KeyCollection Keys;
```

7.2.1.1.14.4.4 NPropertyBag.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.1.1.14.4.5 NPropertyBag.Values Property

Gets the collection of values.

C#

```
public ValueCollection Values;
```

7.2.1.1.15 NRange Structure

Represents range.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NRange : IFormattable {
}
```

Methods

	Name	Description
	NRange [1585]	Initializes a new instance of the NRange structure.

NRange Methods

	Name	Description
	ToString [1586]	Retrieves a string representation of the range object.

NRange Properties

	Name	Description
	From [1586]	Gets or sets lower bound of a range.
S	NativeType [1586]	Defines the native type of the type.
	To [1586]	Gets or sets upper bound of a range.

7.2.1.1.15.1 NRange.NRange Constructor

Initializes a new instance of the NRange structure.

C#

```
public NRange(int from, int to);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
int from	lower bound of a range to be created.
int to	upper bound of a range to be created.

7.2.1.1.15.2 NRange Methods

7.2.1.1.15.2.1 ToString Method

7.2.1.1.15.2.1.1 NRange.ToString Method ()

Retrieves a string representation of the range object.

C#

```
public override string ToString();
```

Returns

String representation of the range.

7.2.1.1.15.2.1.2 NRange.ToString Method (string)

Retrieves a string representation of the range object.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	Reserved.

Returns

A string representation of the range.

7.2.1.1.15.3 NRange Properties

7.2.1.1.15.3.1 NRange.From Property

Gets or sets lower bound of a range.

C#

```
public int From;
```

7.2.1.1.15.3.2 NRange.NativeType Property

Defines the native type of the type.

C#

```
public static NType NativeType;
```

7.2.1.1.15.3.3 NRange.To Property

Gets or sets upper bound of a range.

C#

```
public int To;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.16 NRational Structure

Represents a signed rational number.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NRational : IFormattable {
  public static readonly NRational Empty = new NRational(0, 0);
}
```

NRational Fields

	Name	Description
S R	Empty [1587]	Represents that NRational is a null reference.

Methods

	Name	Description
	NRational [1587]	Initializes a new instance of the NRational structure.

NRational Methods

	Name	Description
	ToString [1588]	Retrieves a string representation of the object.

NRational Properties

	Name	Description
	Denominator [1588]	Sets or retrieves the NRational value Denominator.
S	NativeType [1588]	Defines the native type of the type.
	Numerator [1588]	Sets or retrieves the NRational value Numerator.

7.2.1.1.16.1 NRational.NRational Constructor

Initializes a new instance of the NRational structure.

C#

```
public NRational(int numerator, int denominator);
```

Parameters

Parameters	Description
int numerator	Numerator [1588] of this NRational.
int denominator	Denominator [1588] of this NRational.

7.2.1.1.16.2 NRational Fields

7.2.1.1.16.2.1 NRational.Empty Field

Represents that NRational [1587] is a null reference.

C#

```
public static readonly NRational Empty = new NRational(0, 0);
```

7.2.1.1.16.3 NRational Methods

7.2.1.1.16.3.1 ToString Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.16.3.1.1 NRational.ToString Method ()

Retrieves a string representation of the object.

C#

```
public override string ToString();
```

Returns

A string representation of an object.

7.2.1.1.16.3.1.2 NRational.ToString Method (string)

Retrieves a string representation of the object.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	Format string.

Returns

A string representation of an object.

7.2.1.1.16.4 NRational Properties

7.2.1.1.16.4.1 NRational.Denominator Property

Sets or retrieves the NRational [1587] value Denominator.

C#

```
public int Denominator;
```

7.2.1.1.16.4.2 NRational.NativeType Property

Defines the native type of the type.

C#

```
public static NType NativeType;
```

7.2.1.1.16.4.3 NRational.Numerator Property

Sets or retrieves the NRational [1587] value Numerator.

C#

```
public int Numerator;
```

7.2.1.1.17 NType Class

Manages object types.

C#

```
public sealed class NType : NObject;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	FromObject [1554]	Creates a new NObject [1549] from the specified object.
	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NType Class

	Name	Description
	CopyParameters [1592]	Copies parameters from one object to another object.
	CopyPropertyValues [1592]	Copies properties from one object to another object.
	GetParameter [1592]	Gets value of parameter by parameter id.
	GetPropertyValue [1593]	Retrieves the specified property of the specified object.
S	IdentifierToString [1593]	Converts the specified identifier into a string.
	IsAssignableFrom [1594]	Checks whether the current type is assignable from the specified type.
	IsInstanceOfType [1594]	Checks whether the current type is an instance of the specified object.
	IsSubclassOf [1594]	Checks whether the current type is a sybclass of the specified type.
	Reset [1594]	Resets the specified object.
	ResetPropertyValue [1595]	Resets the specified property.
	SetParameter [1595]	Sets value of parameter by parameter id.
	SetPropertyValue [1595]	Sets the value of the specified property.
S	ToNativeType [1596]	Converts the specified type to a native type.
	ToType [1596]	Converts the current value into a type.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NType Class

	Name	Description
	BaseType [1597]	Gets the base type of the current type.
	HasOwnerType [1597]	Checks whether the current type has an owner type.
	IsAbstract [1597]	Checks whether the current type is abstract.
	IsBasic [1597]	Checks whether the current type is a basic type.
	IsCallback [1597]	Checks whether the current type is a callback type.
	IsCloneable [1598]	Checks whether the current type is cloneable.
	IsComparable [1598]	Checks whether the current type is comparable.
	IsDeprecated [1598]	Checks whether the current type is deprecated.
	IsDisposable [1598]	Checks whether the current type is disposable.
	IsEnum [1598]	Checks whether the current type is an enum type.
	IsEquatable [1598]	Checks whether the current type is equatable.
	IsFlagsEnum [1598]	Checks whether the current type is a flags enum.
	IsHandle [1598]	Checks whether the current type is a handle type.
	IsMemorySerializable [1598]	Checks whether the current type is memory serializable.
	IsObject [1599]	Checks whether the current type is an object type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	IsParsable [1599]	Checks whether the current type is parsable.
	IsPrimitive [1599]	Checks whether the current type is a primitive type.
	IsPublic [1599]	Checks whether the current type is public.
	IsSealed [1599]	Checks whether the current type is sealed.
	IsSerializable [1599]	Checks whether the current type is serializable.
	IsSignNeutral [1599]	Checks whether the current type is sign-neutral.
	IsStatic [1599]	Checks whether the current type is static.
	IsStruct [1599]	Checks whether the current type is a struct type.
	Module [1600]	Gets the module of the current type.
	Name [1600]	Gets the name of the current type.
S	NativeType [1600]	Defines the native type of the object.
S	NTypeCodeNativeType [1600]	Defines the native type of the type code.
	OwnerType [1600]	Gets the owner type of the current type.
	RootType [1600]	Gets the root type of the current type.
	RootTypeCode [1600]	Returns Neurotec.IO [1825]NTypeCode [1652]for the root type of the current type.
	TypeCode [1600]	Returns Neurotec.IO [1825]NTypeCode [1652]for the current type.
	UseInsteadType [1600]	Gets the type to be used instead of current type.
	ValueSize [1601]	Gets the size of the current value.

7.2.1.1.17.1 NType Methods

7.2.1.1.17.1.1 NType.CopyParameters Method

Copies parameters from one object to another object.

C#

```
public new void CopyParameters(NObject dstObject, NObject srcObject);
```

Parameters

Parameters	Description
NObject dstObject	The destination object.
NObject srcObject	The source object.

7.2.1.1.17.1.2 NType.CopyPropertyValues Method

Copies properties from one object to another object.

C#

```
public void CopyPropertyValues(NObject dstObject, NObject srcObject);
```

Parameters

Parameters	Description
NObject dstObject	The destination object.
NObject srcObject	The source object.

7.2.1.1.17.1.3 NType.GetParameter Method

Gets value of parameter by parameter id.

C#

```
public T GetParameter<T>(NObject obj, ushort partId, ushort parameterId);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NObject obj	The object.
ushort partId	Specifies the part of the specified NObject [1549].
ushort parameterId	The parameter identifier.

Returns

The parameter value. Can be null.

7.2.1.1.17.1.4 NType.GetPropertyValue Method

Retrieves the specified property of the specified object.

C#

```
public T GetPropertyValue<T>(NObject obj, string name);
```

Parameters

Parameters	Description
NObject obj	The object to retrieve a property of.
string name	Name [1600] of the property.

Type Parameters

Type Parameters	Description
T	Type of the property.

Returns

The property value.

7.2.1.1.17.1.5 IdentifierToString Method

7.2.1.1.17.1.5.1 NType.IdentifierToString Method (string)

Converts the specified identifier into a string.

C#

```
public static string IdentifierToString(string value);
```

Parameters

Parameters	Description
string value	The indentifier.

Returns

String, representing the identifier.

7.2.1.1.17.1.5.2 NType.IdentifierToString Method (string, string)

Converts the specified identifier into a string.

C#

```
public static string IdentifierToString(string value, string format);
```

Parameters

Parameters	Description
string value	The indentifier.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

string format	Format string.

Returns

String, representing the identifier.

7.2.1.1.17.1.6 NType.IsAssignableFrom Method

Checks whether the current type is assignable from the specified type.

C#

```
public bool IsAssignableFrom(NType type);
```

Parameters

Parameters	Description
NType type	The specified type.

Returns

NTrue [1064] if the current type is assignable from another type; else false.

7.2.1.1.17.1.7 NType.IsInstanceOfType Method

Checks whether the current type is an instance of the specified object.

C#

```
public bool IsInstanceOfType(NObject obj);
```

Parameters

Parameters	Description
NObject obj	The specified object.

Returns

true if the current type is an instance of the object; else false

7.2.1.1.17.1.8 NType.IsSubclassOf Method

Checks whether the current type is a sybclass of the specified type.

C#

```
public bool IsSubclassOf(NType type);
```

Parameters

Parameters	Description
NType type	The specified type.

Returns

NTrue [1064] if the current type is a subclass of another type; else NFalse [1041].

7.2.1.1.17.1.9 NType.Reset Method

Resets the specified object.

C#

```
public void Reset(NObject obj);
```

Parameters

Parameters	Description
NObject obj	The specified object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.17.1.10 NType.ResetPropertyValue Method

Resets the specified property.

C#

```
public void ResetPropertyValue(NObject obj, string name);
```

Parameters

Parameters	Description
NObject obj	The object to reset the property of.
string name	Name [1600] of the property.

7.2.1.1.17.1.11 SetParameter Method

7.2.1.1.17.1.11.1 NType.SetParameter Method (NObject, ushort, ushort, T)

Sets value of parameter by parameter id.

C#

```
public void SetParameter<T>(NObject obj, ushort partId, ushort parameterId, T value);
```

Parameters

Parameters	Description
NObject obj	The object.
ushort partId	Specifies the part of the specified NObject [1549].
ushort parameterId	The parameter identifier.
T value	The parameter value. Can be null.

7.2.1.1.17.1.11.2 NType.SetParameter Method (NObject, ushort, ushort, object)

Sets value of parameter by parameter id.

C#

```
public void SetParameter(NObject obj, ushort partId, ushort parameterId, object value);
```

Parameters

Parameters	Description
NObject obj	The object.
ushort partId	Specifies the part of the specified NObject [1549].
ushort parameterId	The parameter identifier.
object value	The parameter value.

7.2.1.1.17.1.12 SetPropertyValue Method

7.2.1.1.17.1.12.1 NType.SetPropertyValue Method (NObject, string, T)

Sets the value of the specified property.

C#

```
public void SetPropertyValue<T>(NObject obj, string name, T value);
```

Parameters

Parameters	Description
NObject obj	The object to set the property of.
string name	Name [1600] of the property.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

T value	New value of the property.

Type Parameters

Type Parameters	Description
T	Type of the property.

7.2.1.1.17.1.12.2 NType.SetPropertyValue Method (NObject, string, object)

Sets the value of the specified property.

C#

```
public void SetPropertyValue(NObject obj, string name, object value);
```

Parameters

Parameters	Description
NObject obj	The object to set the property of.
string name	Name [1600] of the property.
object value	New value of the property.

7.2.1.1.17.1.13 NType.ToNativeType Method

Converts the specified type to a native type.

C#

```
public static NType ToNativeType(Type value, NAttributes attributes, bool deepMap);
```

Parameters

Parameters	Description
Type value	The type to be converted.
NAttributes attributes	Type attributes.
bool deepMap	true if deep map is to be used; else false

Returns

The native type.

7.2.1.1.17.1.14 ToType Method

7.2.1.1.17.1.14.1 NType.ToType Method ()

Converts the current value into a type.

C#

```
public Type ToType();
```

Returns

The type value.

7.2.1.1.17.1.14.2 NType.ToType Method (NAttributes, bool)

Converts the current value into a type.

C#

```
public Type ToType(NAttributes attributes, bool deepMap);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NAttributes attributes	Type attributes.
bool deepMap	true if deep map is to be used; else false

Returns

The type value.

7.2.1.1.17.1.14.3 NType.ToType Method (NType, NAttributes, bool)

Converts the native type to a regular type.

C#

```
public static Type ToType(NType value, NAttributes attributes, bool deepMap);
```

Parameters

Parameters	Description
NType value	The native type to be convereted.
NAttributes attributes	Type attributes.
bool deepMap	true if deep map is to be used; else false

Returns

Type value.

7.2.1.1.17.2 NType Properties

7.2.1.1.17.2.1 NType.BaseType Property

Gets the base type of the current type.

C#

```
public NType BaseType;
```

7.2.1.1.17.2.2 NType.HasOwnerType Property

Checks whether the current type has an owner type.

C#

```
public bool HasOwnerType;
```

7.2.1.1.17.2.3 NType.IsAbstract Property

Checks whether the current type is abstract.

C#

```
public bool IsAbstract;
```

7.2.1.1.17.2.4 NType.IsBasic Property

Checks whether the current type is a basic type.

C#

```
public bool IsBasic;
```

7.2.1.1.17.2.5 NType.IsCallback Property

Checks whether the current type is a callback type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public bool IsCallback;
```

7.2.1.1.17.2.6 NType.IsCloneable Property

Checks whether the current type is cloneable.

C#

```
public bool IsCloneable;
```

7.2.1.1.17.2.7 NType.IsComparable Property

Checks whether the current type is comparable.

C#

```
public bool IsComparable;
```

7.2.1.1.17.2.8 NType.IsDeprecated Property

Checks whether the current type is deprecated.

C#

```
public bool IsDeprecated;
```

7.2.1.1.17.2.9 NType.IsDisposable Property

Checks whether the current type is disposable.

C#

```
public bool IsDisposable;
```

7.2.1.1.17.2.10 NType.IsEnum Property

Checks whether the current type is an enum type.

C#

```
public bool IsEnum;
```

7.2.1.1.17.2.11 NType.IsEquatable Property

Checks whether the current type is equatable.

C#

```
public bool IsEquatable;
```

7.2.1.1.17.2.12 NType.IsFlagsEnum Property

Checks whether the current type is a flags enum.

C#

```
public bool IsFlagsEnum;
```

7.2.1.1.17.2.13 NType.IsHandle Property

Checks whether the current type is a handle type.

C#

```
public bool IsHandle;
```

7.2.1.1.17.2.14 NType.IsMemorySerializable Property

Checks whether the current type is memory serializable.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public bool IsMemorySerializable;
```

7.2.1.1.17.2.15 NType.IsObject Property

Checks whether the current type is an object type.

C#

```
public bool IsObject;
```

7.2.1.1.17.2.16 NType.IsParsable Property

Checks whether the current type is parsable.

C#

```
public bool IsParsable;
```

7.2.1.1.17.2.17 NType.IsPrimitive Property

Checks whether the current type is a primitive type.

C#

```
public bool IsPrimitive;
```

7.2.1.1.17.2.18 NType.IsPublic Property

Checks whether the current type is public.

C#

```
public bool IsPublic;
```

7.2.1.1.17.2.19 NType.IsSealed Property

Checks whether the current type is sealed.

C#

```
public bool IsSealed;
```

7.2.1.1.17.2.20 NType.IsSerializable Property

Checks whether the current type is serializable.

C#

```
public bool IsSerializable;
```

7.2.1.1.17.2.21 NType.IsSignNeutral Property

Checks whether the current type is sign-neutral.

C#

```
public bool IsSignNeutral;
```

7.2.1.1.17.2.22 NType.IsStatic Property

Checks whether the current type is static.

C#

```
public bool IsStatic;
```

7.2.1.1.17.2.23 NType.IsStruct Property

Checks whether the current type is a struct type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public bool IsStruct;
```

7.2.1.1.17.2.24 NType.Module Property

Gets the module of the current type.

C#

```
public NModule Module;
```

7.2.1.1.17.2.25 NType.Name Property

Gets the name of the current type.

C#

```
public string Name;
```

7.2.1.1.17.2.26 NType.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.1.1.17.2.27 NType.NTypeCodeNativeType Property

Defines the native type of the type code.

C#

```
public static NType NTypeCodeNativeType;
```

7.2.1.1.17.2.28 NType.OwnerType Property

Gets the owner type of the current type.

C#

```
public NType OwnerType;
```

7.2.1.1.17.2.29 NType.RootType Property

Gets the root type of the current type.

C#

```
public NType RootType;
```

7.2.1.1.17.2.30 NType.RootTypeCode Property

Returns Neurotec.IO [1825]NTypeCode [1652]for the root type of the current type.

C#

```
public NTypeCode RootTypeCode;
```

7.2.1.1.17.2.31 NType.TypeCode Property

Returns Neurotec.IO [1825]NTypeCode [1652]for the current type.

C#

```
public NTypeCode TypeCode;
```

7.2.1.1.17.2.32 NType.UseInsteadType Property

Gets the type to be used instead of current type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NType UseInsteadType;
```

7.2.1.1.17.2.33 NType.ValueSize Property

Gets the size of the current value.

C#

```
public int ValueSize;
```

7.2.1.1.18 NTypes Class

Manages object types.

C#

```
public static class NTypes;
```

NTypes Fields

	Name	Description
	CharSet [1602]	Char set constant.
S R	StringEncoding [1602]	The string encoding.
S R	WCharApi [1602]	Whether the Unicode char API is supported.

NTypes Methods

	Name	Description
S	CreateCallback [1602]	Creates a new callback.
S	FreeObjectPointer [1603]	Decrements the reference count for a specified pointer.
S	FreeStringArray [1603]	Frees specified strings.
S	FreeStringElements [1603]	Frees specified string elements.
S	GetObjectFromPointer [1603]	Retrieves an object from a pointer.
S	GetPointerForObject [1604]	Increments the reference count for a pointer to a specified object.
S	IsMemoryTypeValid [1604]	Checks whether the memory type is valid.
S	IsMemoryTypeValidForFree [1604]	Checks whether the memory type is valid for freeing.

NTypes Properties

	Name	Description
S	ACharNativeType [1604]	Gets the native type of the current ASCII char.
S	BooleanNativeType [1605]	Gets the native type of the current Boolean value.
S	CharNativeType [1605]	Gets the native type of the current char.
S	DateTimeNativeType [1605]	Gets the native type of the current NDateTime [361] object.
S	DoubleNativeType [1605]	Gets the native type of the current double number.
S	GuidNativeType [1605]	Gets the native type of the current global unique identifier.
S	Int16NativeType [1605]	Gets the native type of the current 16-bit integer.
S	Int32NativeType [1605]	Gets the native type of the current 32-bit integer.
S	Int64NativeType [1605]	Gets the native type of the current 64-bit integer.
S	Int8NativeType [1605]	Gets the native type of the current 8-bit integer.
S	IntPtrNativeType [1606]	Gets the native type of the current integer pointer.
S	NameStringPairNativeType [1606]	Gets the native type of the current name string pair.
S	NativeType [1606]	Gets the native type of the current object.
S	NAttributesNativeType [1606]	Gets the native type of the current NAttributes [1002] value.
S	NMemoryTypeNativeType [1606]	Gets the native type of the current memory type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NOSFamilyNativeType [1606]	Gets the native type of the current OS family.
	OSFamilyCurrent [1606]	Gets the current OS family.
	PointerNativeType [1606]	Gets the native type of the current pointer.
	ResultNativeType [1606]	Gets the native type of the current NResult [1007] value.
	SingleNativeType [1607]	Gets the native type of the current floating point value.
	StringNativeType [1607]	Gets the native type of the current string.
	TimeSpanNativeType [1607]	Gets the native type of the current NTimeSpan [855] value.
	UInt16NativeType [1607]	Gets the native type of the current unsigned 16-bit integer.
	UInt32NativeType [1607]	Gets the native type of the current unsigned 32-bit integer.
	UInt64NativeType [1607]	Gets the native type of the current unsigned 64-bit integer.
	UInt8NativeType [1607]	Gets the native type of the current unsigned 8-bit integer.
	UIntPtrNativeType [1607]	Gets the native type of the current unsigned integer pointer.
	WCharNativeType [1607]	Gets the native type of the current Unicode char.

7.2.1.1.18.1 NTypes Fields

7.2.1.1.18.1.1 NTypes.CharSet Field

Char set constant.

C#

```
public const CharSet CharSet = System.Runtime.InteropServices.CharSet.Auto;
```

7.2.1.1.18.1.2 NTypes.StringEncoding Field

The string encoding.

C#

```
public static readonly NEncoding StringEncoding;
```

7.2.1.1.18.1.3 NTypes.WCharApi Field

Whether the Unicode char API is supported.

C#

```
public static readonly bool WCharApi;
```

7.2.1.1.18.2 NTypes Methods

7.2.1.1.18.2.1 CreateCallback Method

7.2.1.1.18.2.1.1 NTypes.CreateCallback Method (Delegate, Delegate)

Creates a new callback.

C#

```
public static HNCallback CreateCallback(Delegate callback, Delegate del);
```

Parameters

Parameters	Description
Delegate callback	Native callback delegate.
Delegate del	Managed delegate.

Returns

The new callback.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.18.2.1.2 NTypes.CreateCallback Method (Delegate, object)

Creates a new callback.

C#

```
public static HNCallback CreateCallback(Delegate callback, object param);
```

Parameters

Parameters	Description
Delegate callback	Native callback delegate.
object param	Callback parameters.

Returns

The new callback.

7.2.1.1.18.2.2 NTypes.FreeObjectPointer Method

Decrements the reference count for a specified pointer.

C#

```
public static void FreeObjectPointer(IntPtr ptr);
```

Parameters

Parameters	Description
IntPtr ptr	The pointer.

7.2.1.1.18.2.3 NTypes.FreeStringArray Method

Frees specified strings.

C#

```
public static void FreeStringArray(IntPtr arhStrings, int count);
```

Parameters

Parameters	Description
IntPtr arhStrings	Pointer to an array of strings.
int count	Number of strings in the array.

7.2.1.1.18.2.4 NTypes.FreeStringElements Method

Frees specified string elements.

C#

```
public static void FreeStringElements(IntPtr arhStrings, int count);
```

Parameters

Parameters	Description
IntPtr arhStrings	Pointer to an array of string elements.
int count	Number of string elements in the array.

7.2.1.1.18.2.5 NTypes.GetObjectFromPointer Method

Retrieves an object from a pointer.

C#

```
public static object GetObjectFromPointer(IntPtr ptr);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
IntPtr ptr	A pointer indicating the object.

Returns

A pointer to the object.

7.2.1.1.18.2.6 NTypes.GetPointerForObject Method

Increments the reference count for a pointer to a specified object.

C#

```
public static IntPtr GetPointerForObject(object obj);
```

Parameters

Parameters	Description
object obj	The object.

Returns

The pointer to the object.

7.2.1.1.18.2.7 NTypes.IsMemoryTypeValid Method

Checks whether the memory type is valid.

C#

```
public static bool IsMemoryTypeValid(NMemoryType value);
```

Parameters

Parameters	Description
NMemoryType value	The memory type.

Returns

NTrue [1064] if the memory type is valid; else NFalse [1041].

7.2.1.1.18.2.8 NTypes.IsMemoryTypeValidForFree Method

Checks whether the memory type is valid for freeing.

C#

```
public static bool IsMemoryTypeValidForFree(NMemoryType value);
```

Parameters

Parameters	Description
NMemoryType value	The memory type.

Returns

NTrue [1064] if the memory type is valid; else NFalse [1041].

7.2.1.1.18.3 NTypes Properties

7.2.1.1.18.3.1 NTypes.ACharNativeType Property

Gets the native type of the current ASCII char.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static NType ACharNativeType;
```

7.2.1.1.18.3.2 NTypes.BooleanNativeType Property

Gets the native type of the current Boolean value.

C#

```
public static NType BooleanNativeType;
```

7.2.1.1.18.3.3 NTypes.CharNativeType Property

Gets the native type of the current char.

C#

```
public static NType CharNativeType;
```

7.2.1.1.18.3.4 NTypes.DateTimeNativeType Property

Gets the native type of the current NDateTime [361] object.

C#

```
public static NType DateTimeNativeType;
```

7.2.1.1.18.3.5 NTypes.DoubleNativeType Property

Gets the native type of the current double number.

C#

```
public static NType DoubleNativeType;
```

7.2.1.1.18.3.6 NTypes.GuidNativeType Property

Gets the native type of the current global unique identifier.

C#

```
public static NType GuidNativeType;
```

7.2.1.1.18.3.7 NTypes.Int16NativeType Property

Gets the native type of the current 16-bit integer.

C#

```
public static NType Int16NativeType;
```

7.2.1.1.18.3.8 NTypes.Int32NativeType Property

Gets the native type of the current 32-bit integer.

C#

```
public static NType Int32NativeType;
```

7.2.1.1.18.3.9 NTypes.Int64NativeType Property

Gets the native type of the current 64-bit integer.

C#

```
public static NType Int64NativeType;
```

7.2.1.1.18.3.10 NTypes.Int8NativeType Property

Gets the native type of the current 8-bit integer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static NType Int8NativeType;
```

7.2.1.1.18.3.11 NTypes.IntPtrNativeType Property

Gets the native type of the current integer pointer.

C#

```
public static NType IntPtrNativeType;
```

7.2.1.1.18.3.12 NTypes.NameStringPairNativeType Property

Gets the native type of the current name string pair.

C#

```
public static NType NameStringPairNativeType;
```

7.2.1.1.18.3.13 NTypes.NativeType Property

Gets the native type of the current object.

C#

```
public static NType NativeType;
```

7.2.1.1.18.3.14 NTypes.NAttributesNativeType Property

Gets the native type of the current NAttributes [1002] value.

C#

```
public static NType NAttributesNativeType;
```

7.2.1.1.18.3.15 NTypes.NMemoryTypeNativeType Property

Gets the native type of the current memory type.

C#

```
public static NType NMemoryTypeNativeType;
```

7.2.1.1.18.3.16 NTypes.NOSFamilyNativeType Property

Gets the native type of the current OS family.

C#

```
public static NType NOSFamilyNativeType;
```

7.2.1.1.18.3.17 NTypes.OSFamilyCurrent Property

Gets the current OS family.

C#

```
public static NOSFamily OSFamilyCurrent;
```

7.2.1.1.18.3.18 NTypes.PointerNativeType Property

Gets the native type of the current pointer.

C#

```
public static NType PointerNativeType;
```

7.2.1.1.18.3.19 NTypes.ResultNativeType Property

Gets the native type of the current NResult [1007] value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static NType ResultNativeType;
```

7.2.1.1.18.3.20 NTypes.SingleNativeType Property

Gets the native type of the current floating point value.

C#

```
public static NType SingleNativeType;
```

7.2.1.1.18.3.21 NTypes.StringNativeType Property

Gets the native type of the current string.

C#

```
public static NType StringNativeType;
```

7.2.1.1.18.3.22 NTypes.TimeSpanNativeType Property

Gets the native type of the current NTimeSpan [855] value.

C#

```
public static NType TimeSpanNativeType;
```

7.2.1.1.18.3.23 NTypes.UInt16NativeType Property

Gets the native type of the current unsigned 16-bit integer.

C#

```
public static NType UInt16NativeType;
```

7.2.1.1.18.3.24 NTypes.UInt32NativeType Property

Gets the native type of the current unsigned 32-bit integer.

C#

```
public static NType UInt32NativeType;
```

7.2.1.1.18.3.25 NTypes.UInt64NativeType Property

Gets the native type of the current unsigned 64-bit integer.

C#

```
public static NType UInt64NativeType;
```

7.2.1.1.18.3.26 NTypes.UInt8NativeType Property

Gets the native type of the current unsigned 8-bit integer.

C#

```
public static NType UInt8NativeType;
```

7.2.1.1.18.3.27 NTypes.UIntPtrNativeType Property

Gets the native type of the current unsigned integer pointer.

C#

```
public static NType UIntPtrNativeType;
```

7.2.1.1.18.3.28 NTypes.WCharNativeType Property

Gets the native type of the current Unicode char.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static NType WCharNativeType;
```

7.2.1.1.19 NURational Structure

Represents an unsigned rational number.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NURational : IFormattable {
  public static readonly NURational Empty = new NURational(0, 0);
}
```

NURational Fields

	Name	Description
S R	Empty [1608]	Represents a NURational that is a null reference.

Methods

	Name	Description
	NURational [1608]	Initializes a new instance of the NURational structure.

NURational Methods

	Name	Description
	ToString [1609]	Retrieves a string representation of the object.

NURational Properties

	Name	Description
	Denominator [1609]	Sets or retrieves the NRational [1587] value Denominator.
S	NativeType [1609]	Defines the native type of the type.
	Numerator [1609]	Sets or retrieves the NRational [1587] value Numerator.

7.2.1.1.19.1 NURational.NURational Constructor

Initializes a new instance of the NURational structure.

C#

```
public NURational(uint numerator, uint denominator);
```

Parameters

Parameters	Description
uint numerator	Numerator [1609] of this NURational.
uint denominator	Denominator [1609] of this NURational.

7.2.1.1.19.2 NURational Fields

7.2.1.1.19.2.1 NURational.Empty Field

Represents a NURational [1608] that is a null reference.

C#

```
public static readonly NURational Empty = new NURational(0, 0);
```

7.2.1.1.19.3 NURational Methods

7.2.1.1.19.3.1 ToString Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.19.3.1.1 NURational.ToString Method ()

Retrieves a string representation of the object.

C#

```
public override string ToString();
```

Returns

A string representation of an object.

7.2.1.1.19.3.1.2 NURational.ToString Method (string)

Retrieves a string representation of the object.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	Format string.

Returns

A string representation of an object.

7.2.1.1.19.4 NURational Properties

7.2.1.1.19.4.1 NURational.Denominator Property

Sets or retrieves the NRational [1587] value Denominator.

C#

```
public uint Denominator;
```

7.2.1.1.19.4.2 NURational.NativeType Property

Defines the native type of the type.

C#

```
public static NType NativeType;
```

7.2.1.1.19.4.3 NURational.Numerator Property

Sets or retrieves the NRational [1587] value Numerator.

C#

```
public uint Numerator;
```

7.2.1.1.20 NValue Class

Provides functionality for management and conversion of values of different types.

C#

```
public class NValue : NObject;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

Methods

NValue Class

	Name	Description
	NValue [1613]	Initializes a new instance of the NValue class.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	FromObject [1554]	Creates a new NObject [1549] from the specified object.
	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NValue Class

	Name	Description
S	ChangeType [1619]	Changes the type of the specified value to the new type.
S	FromObject [1619]	Creates a new value from the specified object.
S	FromPointer [1621]	Creates a new value from the specified pointer.
S	FromResult [1621]	Creates a new value from the specified result.
S	FromString [1622]	Creates a new value from the specified string.
S	FromType [1622]	Creates a new value from the specified type.
	ToBoolean [1623]	Converts the specified value into a Boolean value.
	ToByte [1624]	Converts the specified value into a byte.
	ToChar [1625]	Converts the specified value into a char.
	ToDouble [1625]	Converts the specified value into a double number.
	ToInt16 [1626]	Converts the specified value into a short integer.
	ToInt32 [1627]	Converts the specified value into an integer.
	ToInt64 [1627]	Converts the specified value into a long integer.
	ToIntPtr [1628]	Converts the specified value into a pointer.
	ToObject [1629]	Converts the specified value into an object.
	ToPointer [1630]	Converts the specified value into a pointer.
	ToResult [1631]	Converts the specified value into a result.
	ToSByte [1632]	Converts the specified value into a signed byte.
	ToSingle [1633]	Converts the specified value into a floating point value.
	ToString [1633]	Converts the specified value into a string.
	ToType [1634]	Converts the specified value into a value of specified type.
	ToUInt16 [1636]	Converts the specified value into an unsigned short integer.
	ToUInt32 [1636]	Converts the specified value into an unsigned integer.
	ToUInt64 [1637]	Converts the specified value into an unsigned long integer.
	ToUIntPtr [1638]	Converts the specified value into a pointer.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NValue Class

	Name	Description
	Attributes [1639]	Gets the value attributes.
	IsWrapper [1639]	Checks whether the value is a wrapper.
S	KeyValuePairNativeType [1639]	Defines the native type of the key value pair structure.
	LongSize [1639]	Gets the size of the value (64-bit).
S	NameValuePairNativeType [1639]	Defines the native type of the name value pair structure.
	NativeSize [1639]	Gets the native size of the value.
S	NativeType [1639]	Defines the native type of the object.
	Ptr [1639]	Gets a poitner to the value.
	Size [1639]	Gets the size of the value.
	TypeCode [1640]	Gets the type code of the value.
	ValueType [1640]	Gets the type of the value.

7.2.1.1.20.1 NValue Constructor

7.2.1.1.20.1.1 NValue.NValue Constructor (IntPtr)

Initializes a new instance of the NValue class.

C#

```
public NValue(IntPtr value);
```

Parameters

Parameters	Description
IntPtr value	The original pointer.

7.2.1.1.20.1.2 NValue.NValue Constructor (IntPtr, NAttributes)

Initializes a new instance of the NValue class.

C#

```
public NValue(IntPtr value, NAttributes attributes);
```

Parameters

Parameters	Description
IntPtr value	The original pointer.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.3 NValue.NValue Constructor (UIntPtr)

Initializes a new instance of the NValue class.

C#

```
public NValue(UIntPtr value);
```

Parameters

Parameters	Description
UIntPtr value	The original pointer.

7.2.1.1.20.1.4 NValue.NValue Constructor (UIntPtr, NAttributes)

Initializes a new instance of the NValue class.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NValue(UIntPtr value, NAttributes attributes);
```

Parameters

Parameters	Description
UIntPtr value	The original pointer.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.5 NValue.NValue Constructor (bool)

Initializes a new instance of the NValue class.

C#

```
public NValue(bool value);
```

Parameters

Parameters	Description
bool value	The original Boolean value.

7.2.1.1.20.1.6 NValue.NValue Constructor (bool, NAttributes)

Initializes a new instance of the NValue class.

C#

```
public NValue(bool value, NAttributes attributes);
```

Parameters

Parameters	Description
bool value	The original Boolean value.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.7 NValue.NValue Constructor (byte)

Initializes a new instance of the NValue class.

C#

```
public NValue(byte value);
```

Parameters

Parameters	Description
byte value	The original byte.

7.2.1.1.20.1.8 NValue.NValue Constructor (byte, NAttributes)

Initializes a new instance of the NValue class.

C#

```
public NValue(byte value, NAttributes attributes);
```

Parameters

Parameters	Description
byte value	The original byte.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.9 NValue.NValue Constructor (char)

Initializes a new instance of the NValue class.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NValue(char value);
```

Parameters

Parameters	Description
char value	The original char.

7.2.1.1.20.1.10 NValue.NValue Constructor (char, NAttributes)

Initializes a new instance of the NValue class.

C#

```
public NValue(char value, NAttributes attributes);
```

Parameters

Parameters	Description
char value	The original char.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.11 NValue.NValue Constructor (double)

Initializes a new instance of the NValue class.

C#

```
public NValue(double value);
```

Parameters

Parameters	Description
double value	The original double value.

7.2.1.1.20.1.12 NValue.NValue Constructor (double, NAttributes)

Initializes a new instance of the NValue class.

C#

```
public NValue(double value, NAttributes attributes);
```

Parameters

Parameters	Description
double value	The original double value.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.13 NValue.NValue Constructor (float)

Initializes a new instance of the NValue class.

C#

```
public NValue(float value);
```

Parameters

Parameters	Description
float value	The original floating point value.

7.2.1.1.20.1.14 NValue.NValue Constructor (float, NAttributes)

Initializes a new instance of the NValue class.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NValue(float value, NAttributes attributes);
```

Parameters

Parameters	Description
float value	The original floating point value.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.15 NValue.NValue Constructor (int)

Initializes a new instance of the NValue class.

C#

```
public NValue(int value);
```

Parameters

Parameters	Description
int value	The original integer.

7.2.1.1.20.1.16 NValue.NValue Constructor (int, NAttributes)

Initializes a new instance of the NValue class.

C#

```
public NValue(int value, NAttributes attributes);
```

Parameters

Parameters	Description
int value	The original integer.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.17 NValue.NValue Constructor (long)

Initializes a new instance of the NValue class.

C#

```
public NValue(long value);
```

Parameters

Parameters	Description
long value	The original long integer.

7.2.1.1.20.1.18 NValue.NValue Constructor (long, NAttributes)

Initializes a new instance of the NValue class.

C#

```
public NValue(long value, NAttributes attributes);
```

Parameters

Parameters	Description
long value	The original long integer.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.19 NValue.NValue Constructor (sbyte)

Initializes a new instance of the NValue class.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NValue(sbyte value);
```

Parameters

Parameters	Description
sbyte value	The original signed byte.

7.2.1.1.20.1.20 NValue.NValue Constructor (sbyte, NAttributes)

Initializes a new instance of the NValue class.

C#

```
public NValue(sbyte value, NAttributes attributes);
```

Parameters

Parameters	Description
sbyte value	The original signed byte.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.21 NValue.NValue Constructor (short)

Initializes a new instance of the NValue class.

C#

```
public NValue(short value);
```

Parameters

Parameters	Description
short value	The original short integer.

7.2.1.1.20.1.22 NValue.NValue Constructor (short, NAttributes)

Initializes a new instance of the NValue class.

C#

```
public NValue(short value, NAttributes attributes);
```

Parameters

Parameters	Description
short value	The original short integer.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.23 NValue.NValue Constructor (uint)

Initializes a new instance of the NValue class.

C#

```
public NValue(uint value);
```

Parameters

Parameters	Description
uint value	The original unsigned integer.

7.2.1.1.20.1.24 NValue.NValue Constructor (uint, NAttributes)

Initializes a new instance of the NValue class.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NValue(uint value, NAttributes attributes);
```

Parameters

Parameters	Description
uint value	The original integer.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.25 NValue.NValue Constructor (ulong)

Initializes a new instance of the NValue class.

C#

```
public NValue(ulong value);
```

Parameters

Parameters	Description
ulong value	The original unsigned long integer.

7.2.1.1.20.1.26 NValue.NValue Constructor (ulong, NAttributes)

Initializes a new instance of the NValue class.

C#

```
public NValue(ulong value, NAttributes attributes);
```

Parameters

Parameters	Description
ulong value	The original unsigned long integer.
NAttributes attributes	Value attributes.

7.2.1.1.20.1.27 NValue.NValue Constructor (ushort)

Initializes a new instance of the NValue class.

C#

```
public NValue(ushort value);
```

Parameters

Parameters	Description
ushort value	The original unsigned short integer.

7.2.1.1.20.1.28 NValue.NValue Constructor (ushort, NAttributes)

Initializes a new instance of the NValue class.

C#

```
public NValue(ushort value, NAttributes attributes);
```

Parameters

Parameters	Description
ushort value	The original unsigned short integer.
NAttributes attributes	Value attributes.

7.2.1.1.20.2 NValue Methods

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.20.2.1 ChangeType Method

7.2.1.1.20.2.1.1 NValue.ChangeType Method (NValue, NType)

Changes the type of the specified value to the new type.

C#

```
public static NValue ChangeType(NValue srcValue, NType type);
```

Parameters

Parameters	Description
NValue srcValue	The original value.
NType type	The new type of the value.

Returns

The new value.

7.2.1.1.20.2.1.2 NValue.ChangeType Method (NValue, NType, NAttributes)

Changes the type of the specified value to the new type.

C#

```
public static NValue ChangeType(NValue srcValue, NType type, NAttributes attributes);
```

Parameters

Parameters	Description
NValue srcValue	The original value.
NType type	The new type of the value.
NAttributes attributes	Value attributes.

Returns

The new value.

7.2.1.1.20.2.1.3 NValue.ChangeType Method (NValue, NType, NAttributes, string)

Changes the type of the specified value to the new type.

C#

```
public static NValue ChangeType(NValue srcValue, NType type, NAttributes attributes, string
format);
```

Parameters

Parameters	Description
NValue srcValue	The original value.
NType type	The new type of the value.
NAttributes attributes	Value attributes.
string format	Format string.

7.2.1.1.20.2.2 FromObject Method

7.2.1.1.20.2.2.1 NValue.FromObject Method (NType, NObject)

Creates a new value from the specified object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static NValue FromObject(NType type, NObject value);
```

Parameters

Parameters	Description
NType type	Type of the object.
NObject value	The original object.

Returns

The new value.

7.2.1.1.20.2.2.2 NValue.FromObject Method (NType, NObject, NAttributes)

Creates a new value from the specified object.

C#

```
public static NValue FromObject(NType type, NObject value, NAttributes attributes);
```

Parameters

Parameters	Description
NType type	Type of the object.
NObject value	The original object.
NAttributes attributes	Value attributes.

Returns

The new value.

7.2.1.1.20.2.2.3 NValue.FromObject Method (object)

Creates a new value from the specified object.

C#

```
public static new NValue FromObject(object value);
```

Parameters

Parameters	Description
object value	The original object.

Returns

The new value.

7.2.1.1.20.2.2.4 NValue.FromObject Method (object, NAttributes)

Creates a new value from the specified object.

C#

```
public static NValue FromObject(object value, NAttributes attributes);
```

Parameters

Parameters	Description
object value	The original object.
NAttributes attributes	Value attributes.

Returns

The new value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.20.2.3 FromPointer Method

7.2.1.1.20.2.3.1 NValue.FromPointer Method (IntPtr)

Creates a new value from the specified pointer.

C#

```
public static NValue FromPointer(IntPtr value);
```

Parameters

Parameters	Description
IntPtr value	The original pointer.

Returns

The new value.

7.2.1.1.20.2.3.2 NValue.FromPointer Method (IntPtr, NAttributes)

Creates a new value from the specified pointer.

C#

```
public static NValue FromPointer(IntPtr value, NAttributes attributes);
```

Parameters

Parameters	Description
IntPtr value	The original pointer.
NAttributes attributes	Value attributes.

Returns

The new value.

7.2.1.1.20.2.4 FromResult Method

7.2.1.1.20.2.4.1 NValue.FromResult Method (int)

Creates a new value from the specified result.

C#

```
public static NValue FromResult(int value);
```

Parameters

Parameters	Description
int value	The original result.

Returns

The new value.

7.2.1.1.20.2.4.2 NValue.FromResult Method (int, NAttributes)

Creates a new value from the specified result.

C#

```
public static NValue FromResult(int value, NAttributes attributes);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
int value	The original result.
NAttributes attributes	Value attributes.

Returns

The new value.

7.2.1.1.20.2.5 FromString Method

7.2.1.1.20.2.5.1 NValue.FromString Method (string)

Creates a new value from the specified string.

C#

```
public static NValue FromString(string value);
```

Parameters

Parameters	Description
string value	The original string.

Returns

The new value.

7.2.1.1.20.2.5.2 NValue.FromString Method (string, NAttributes)

Creates a new value from the specified string.

C#

```
public static NValue FromString(string value, NAttributes attributes);
```

Parameters

Parameters	Description
string value	The original string.
NAttributes attributes	Value attributes.

Returns

The new value.

7.2.1.1.20.2.6 FromType Method

7.2.1.1.20.2.6.1 NValue.FromType Method (T)

Creates a new value from the specified type.

C#

```
public static NValue FromType<T>(T value);
```

Parameters

Parameters	Description
T value	The original type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Type Parameters

Type Parameters	Description
T	Type of the value.

Returns

The new value.

7.2.1.1.20.2.6.2 NValue.FromType Method (T, NAttributes)

Creates a new value from the specified type.

C#

```
public static NValue FromType<T>(T value, NAttributes attributes);
```

Parameters

Parameters	Description
T value	The original type.
NAttributes attributes	Value attributes.

Type Parameters

Type Parameters	Description
T	Type of the value.

Returns

The new value.

7.2.1.1.20.2.7 ToBoolean Method

7.2.1.1.20.2.7.1 NValue.ToBoolean Method ()

Converts the specified value into a Boolean value.

C#

```
public bool ToBoolean();
```

Returns

New Boolean value.

7.2.1.1.20.2.7.2 NValue.ToBoolean Method (NAttributes)

Converts the specified value into a Boolean value.

C#

```
public bool ToBoolean(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New Boolean value.

7.2.1.1.20.2.7.3 NValue.ToBoolean Method (NAttributes, string)

Converts the specified value into a Boolean value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public bool ToBoolean(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New Boolean value.

7.2.1.1.20.2.8 ToByte Method

7.2.1.1.20.2.8.1 NValue.ToByte Method ()

Converts the specified value into a byte.

C#

```
public byte ToByte();
```

Returns

New byte.

7.2.1.1.20.2.8.2 NValue.ToByte Method (NAttributes)

Converts the specified value into a byte.

C#

```
public byte ToByte(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New byte.

7.2.1.1.20.2.8.3 NValue.ToByte Method (NAttributes, string)

Converts the specified value into a byte.

C#

```
public byte ToByte(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New byte.

7.2.1.1.20.2.9 ToChar Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.20.2.9.1 NValue.ToChar Method ()

Converts the specified value into a char.

C#

```
public char ToChar();
```

Returns

New char.

7.2.1.1.20.2.9.2 NValue.ToChar Method (NAttributes)

Converts the specified value into a char.

C#

```
public char ToChar(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New char.

7.2.1.1.20.2.9.3 NValue.ToChar Method (NAttributes, string)

Converts the specified value into a char.

C#

```
public char ToChar(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New char.

7.2.1.1.20.2.10 ToDouble Method

7.2.1.1.20.2.10.1 NValue.ToDouble Method ()

Converts the specified value into a double number.

C#

```
public double ToDouble();
```

Returns

New double number.

7.2.1.1.20.2.10.2 NValue.ToDouble Method (NAttributes)

Converts the specified value into a double number.

C#

```
public double ToDouble(NAttributes attributes);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New double number.

7.2.1.1.20.2.10.3 NValue.ToDouble Method (NAttributes, string)

Converts the specified value into a double number.

C#

```
public double ToDouble(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New double number.

7.2.1.1.20.2.11 ToInt16 Method

7.2.1.1.20.2.11.1 NValue.ToInt16 Method ()

Converts the specified value into a short integer.

C#

```
public short ToInt16();
```

Returns

New short integer.

7.2.1.1.20.2.11.2 NValue.ToInt16 Method (NAttributes)

Converts the specified value into a short integer.

C#

```
public short ToInt16(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New short integer.

7.2.1.1.20.2.11.3 NValue.ToInt16 Method (NAttributes, string)

Converts the specified value into a short integer.

C#

```
public short ToInt16(NAttributes attributes, string format);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New short integer.

7.2.1.1.20.2.12 ToInt32 Method

7.2.1.1.20.2.12.1 NValue.ToInt32 Method ()

Converts the specified value into an integer.

C#

```
public int ToInt32();
```

Returns

New integer.

7.2.1.1.20.2.12.2 NValue.ToInt32 Method (NAttributes)

Converts the specified value into an integer.

C#

```
public int ToInt32(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New integer.

7.2.1.1.20.2.12.3 NValue.ToInt32 Method (NAttributes, string)

Converts the specified value into an integer.

C#

```
public int ToInt32(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New integer.

7.2.1.1.20.2.13 ToInt64 Method

7.2.1.1.20.2.13.1 NValue.ToInt64 Method ()

Converts the specified value into a long integer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public long ToInt64();
```

Returns

New long integer.

7.2.1.1.20.2.13.2 NValue.ToInt64 Method (NAttributes)

Converts the specified value into a long integer.

C#

```
public long ToInt64(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New long integer.

7.2.1.1.20.2.13.3 NValue.ToInt64 Method (NAttributes, string)

Converts the specified value into a long integer.

C#

```
public long ToInt64(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New long integer.

7.2.1.1.20.2.14 ToIntPtr Method

7.2.1.1.20.2.14.1 NValue.ToIntPtr Method ()

Converts the specified value into a pointer.

C#

```
public IntPtr ToIntPtr();
```

Returns

New integer pointer.

7.2.1.1.20.2.14.2 NValue.ToIntPtr Method (NAttributes)

Converts the specified value into a pointer.

C#

```
public IntPtr ToIntPtr(NAttributes attributes);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New integer pointer.

7.2.1.1.20.2.14.3 NValue.ToIntPtr Method (NAttributes, string)

Converts the specified value into a pointer.

C#

```
public IntPtr ToIntPtr(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New integer pointer.

7.2.1.1.20.2.15 ToObject Method

7.2.1.1.20.2.15.1 NValue.ToObject Method ()

Converts the specified value into an object.

C#

```
public object ToObject();
```

Returns

New object.

7.2.1.1.20.2.15.2 NValue.ToObject Method (NType)

Converts the specified value into an object.

C#

```
public NObject ToObject(NType type);
```

Parameters

Parameters	Description
NType type	Type of the object.

Returns

New object.

7.2.1.1.20.2.15.3 NValue.ToObject Method (NType, NAttributes)

Converts the specified value into an object.

C#

```
public NObject ToObject(NType type, NAttributes attributes);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NType type	Type of the object.
NAttributes attributes	Value attributes.

Returns

New object.

7.2.1.1.20.2.15.4 NValue.ToObject Method (NType, NAttributes, string)

Converts the specified value into an object.

C#

```
public NObject ToObject(NType type, NAttributes attributes, string format);
```

Parameters

Parameters	Description
NType type	Type of the object.
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New object.

7.2.1.1.20.2.15.5 NValue.ToObject Method (NValue)

Converts the specified value into an object.

C#

```
public static object ToObject(NValue value);
```

Parameters

Parameters	Description
NValue value	The original value.

Returns

The new value.

7.2.1.1.20.2.16 ToPointer Method

7.2.1.1.20.2.16.1 NValue.ToPointer Method ()

Converts the specified value into a pointer.

C#

```
public IntPtr ToPointer();
```

Returns

New pointer.

7.2.1.1.20.2.16.2 NValue.ToPointer Method (NAttributes)

Converts the specified value into a pointer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public IntPtr ToPointer(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New pointer.

7.2.1.1.20.2.16.3 NValue.ToPointer Method (NAttributes, string)

Converts the specified value into a pointer.

C#

```
public IntPtr ToPointer(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New pointer.

7.2.1.1.20.2.17 ToResult Method

7.2.1.1.20.2.17.1 NValue.ToResult Method ()

Converts the specified value into a result.

C#

```
public int ToResult();
```

Returns

New result.

7.2.1.1.20.2.17.2 NValue.ToResult Method (NAttributes)

Converts the specified value into a result.

C#

```
public int ToResult(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New result.

7.2.1.1.20.2.17.3 NValue.ToResult Method (NAttributes, string)

Converts the specified value into a result.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public int ToResult(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New result.

7.2.1.1.20.2.18 ToSByte Method

7.2.1.1.20.2.18.1 NValue.ToSByte Method ()

Converts the specified value into a signed byte.

C#

```
public sbyte ToSByte();
```

Returns

New signed byte.

7.2.1.1.20.2.18.2 NValue.ToSByte Method (NAttributes)

Converts the specified value into a signed byte.

C#

```
public sbyte ToSByte(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New signed byte.

7.2.1.1.20.2.18.3 NValue.ToSByte Method (NAttributes, string)

Converts the specified value into a signed byte.

C#

```
public sbyte ToSByte(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New signed byte.

7.2.1.1.20.2.19 ToSingle Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.20.2.19.1 NValue.ToSingle Method ()

Converts the specified value into a floating point value.

C#

```
public float ToSingle();
```

Returns

New floating point value.

7.2.1.1.20.2.19.2 NValue.ToSingle Method (NAttributes)

Converts the specified value into a floating point value.

C#

```
public float ToSingle(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New floating point value.

7.2.1.1.20.2.19.3 NValue.ToSingle Method (NAttributes, string)

Converts the specified value into a floating point value.

C#

```
public float ToSingle(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New floating point value.

7.2.1.1.20.2.20 ToString Method

7.2.1.1.20.2.20.1 NValue.ToString Method (NAttributes)

Converts the specified value into a string.

C#

```
public string ToString(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.20.2.20.2 NValue.ToString Method (NAttributes, string)

Converts the specified value into a string.

C#

```
public string ToString(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New string.

7.2.1.1.20.2.21 ToType Method

7.2.1.1.20.2.21.1 NValue.ToType Method ()

Converts the specified value into a value of specified type.

C#

```
public T ToType<T>();
```

Type Parameters

Type Parameters	Description
T	Type of the value.

Returns

New value.

7.2.1.1.20.2.21.2 NValue.ToType Method (NAttributes)

Converts the specified value into a value of specified type.

C#

```
public T ToType<T>(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Type Parameters

Type Parameters	Description
T	Type of the value.

Returns

New value.

7.2.1.1.20.2.21.3 NValue.ToType Method (NAttributes, string)

Converts the specified value into a value of specified type.

C#

```
public T ToType<T>(NAttributes attributes, string format);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Type Parameters

Type Parameters	Description
T	Type of the value.

Returns

New value.

7.2.1.1.20.2.21.4 NValue.ToType Method (Type)

Converts the specified value into a value of specified type.

C#

```
public object ToType(Type type);
```

Parameters

Parameters	Description
Type type	Type of the value.

Returns

New value.

7.2.1.1.20.2.21.5 NValue.ToType Method (Type, NAttributes)

Converts the specified value into a value of specified type.

C#

```
public object ToType(Type type, NAttributes attributes);
```

Parameters

Parameters	Description
Type type	Type of the value.
NAttributes attributes	Value attributes.

Returns

New value.

7.2.1.1.20.2.21.6 NValue.ToType Method (Type, NAttributes, string)

Converts the specified value into a value of specified type.

C#

```
public object ToType(Type type, NAttributes attributes, string format);
```

Parameters

Parameters	Description
Type type	Type of the value.
NAttributes attributes	Value attributes.
string format	Format string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

New value.

7.2.1.1.20.2.22 ToUInt16 Method

7.2.1.1.20.2.22.1 NValue.ToUInt16 Method ()

Converts the specified value into an unsigned short integer.

C#

```
public ushort ToUInt16();
```

Returns

New unsigned short integer.

7.2.1.1.20.2.22.2 NValue.ToUInt16 Method (NAttributes)

Converts the specified value into an unsigned short integer.

C#

```
public ushort ToUInt16(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New unsigned short integer.

7.2.1.1.20.2.22.3 NValue.ToUInt16 Method (NAttributes, string)

Converts the specified value into an unsigned short integer.

C#

```
public ushort ToUInt16(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New unsigned short integer.

7.2.1.1.20.2.23 ToUInt32 Method

7.2.1.1.20.2.23.1 NValue.ToUInt32 Method ()

Converts the specified value into an unsigned integer.

C#

```
public uint ToUInt32();
```

Returns

New unsigned integer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.20.2.23.2 NValue.ToUInt32 Method (NAttributes)

Converts the specified value into an unsigned integer.

C#

```
public uint ToUInt32(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New unsigned integer.

7.2.1.1.20.2.23.3 NValue.ToUInt32 Method (NAttributes, string)

Converts the specified value into an unsigned integer.

C#

```
public uint ToUInt32(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New unsigned integer.

7.2.1.1.20.2.24 ToUInt64 Method

7.2.1.1.20.2.24.1 NValue.ToUInt64 Method ()

Converts the specified value into an unsigned long integer.

C#

```
public ulong ToUInt64();
```

Returns

New unsigned long integer.

7.2.1.1.20.2.24.2 NValue.ToUInt64 Method (NAttributes)

Converts the specified value into an unsigned long integer.

C#

```
public ulong ToUInt64(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New unsigned long integer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.20.2.24.3 NValue.ToUInt64 Method (NAttributes, string)

Converts the specified value into an unsigned long integer.

C#

```
public ulong ToUInt64(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New unsigned long integer.

7.2.1.1.20.2.25 ToUIntPtr Method

7.2.1.1.20.2.25.1 NValue.ToUIntPtr Method ()

Converts the specified value into a pointer.

C#

```
public UIntPtr ToUIntPtr();
```

Returns

New unsigned integer pointer.

7.2.1.1.20.2.25.2 NValue.ToUIntPtr Method (NAttributes)

Converts the specified value into a pointer.

C#

```
public UIntPtr ToUIntPtr(NAttributes attributes);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.

Returns

New unsigned integer pointer.

7.2.1.1.20.2.25.3 NValue.ToUIntPtr Method (NAttributes, string)

Converts the specified value into a pointer.

C#

```
public UIntPtr ToUIntPtr(NAttributes attributes, string format);
```

Parameters

Parameters	Description
NAttributes attributes	Value attributes.
string format	Format string.

Returns

New unsigned integer pointer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.20.3 NValue Properties

7.2.1.1.20.3.1 NValue.Attributes Property

Gets the value attributes.

C#

```
public NAttributes Attributes;
```

7.2.1.1.20.3.2 NValue.IsWrapper Property

Checks whether the value is a wrapper.

C#

```
public bool IsWrapper;
```

7.2.1.1.20.3.3 NValue.KeyValuePairNativeType Property

Defines the native type of the key value pair structure.

C#

```
public static NType KeyValuePairNativeType;
```

7.2.1.1.20.3.4 NValue.LongSize Property

Gets the size of the value (64-bit).

C#

```
public long LongSize;
```

7.2.1.1.20.3.5 NValue.NameValuePairNativeType Property

Defines the native type of the name value pair structure.

C#

```
public static NType NameValuePairNativeType;
```

7.2.1.1.20.3.6 NValue.NativeSize Property

Gets the native size of the value.

C#

```
public UIntPtr NativeSize;
```

7.2.1.1.20.3.7 NValue.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.1.1.20.3.8 NValue.Ptr Property

Gets a poitner to the value.

C#

```
public IntPtr Ptr;
```

7.2.1.1.20.3.9 NValue.Size Property

Gets the size of the value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public int Size;
```

7.2.1.1.20.3.10 NValue.TypeCode Property

Gets the type code of the value.

C#

```
public NTypeCode TypeCode;
```

7.2.1.1.20.3.11 NValue.ValueType Property

Gets the type of the value.

C#

```
public NType ValueType;
```

7.2.1.1.21 NVersion Structure

Manages version information.

C#

```
public struct NVersion : IFormattable, IEquatable<NVersion>, IComparable {
  public static readonly NVersion Empty = new NVersion(0);
}
```

NVersion Fields

	Name	Description
S R	Empty [1641]	New NVersion object.

Methods

	Name	Description
	NVersion [1641]	Initializes a new instance of NVersion structure.

NVersion Methods

	Name	Description
	CompareTo [1642]	Compares the current version with the specified version and returns an integer that indicates their relative position in the sort order.
	Equals [1642]	Checks whether the current version equals the specified version.
	GetHashCode [1642]	Gets the hash code.
	ToString [1643]	Gets the string representation of the current version.

NVersion Operators

	Name	Description
S	!= [1643]	Checks whether the two NVersion values are not equal.
S	< [1643]	Checks whether the first NVersion value is smaller than the second NVersion value.
S	<= [1644]	Checks whether the first NVersion value is smaller than or equal to the second NVersion value.
S	== [1644]	Checks whether the two NVersion values are equal.
S	> [1644]	Checks whether the first NVersion value is larger than the second NVersion value.
S	>= [1644]	Checks whether the first NVersion value is larger than or equal to the second NVersion value.
S	ushort [1645]	Converts NVersion value to an unsigned short value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NVersion Properties

	Name	Description
	Major [1645]	Gets the major version value.
	Minor [1645]	Gets the minor version value.
S	NativeType [1645]	Defines the native type of the type.

7.2.1.1.21.1 NVersion Operator

7.2.1.1.21.1.1 NVersion.NVersion Constructor (int, int)

Initializes a new instance of NVersion structure.

C#

```
public NVersion(int major, int minor);
```

Parameters

Parameters	Description
int major	Major [1645] version number.
int minor	Minor [1645] version number.

7.2.1.1.21.1.2 NVersion.NVersion Constructor (ushort)

Initializes a new instance of NVersion structure.

C#

```
public NVersion(ushort value);
```

Parameters

Parameters	Description
ushort value	Version value.

7.2.1.1.21.1.3 NVersion.NVersion Operator (ushort)

Converts an unsigned short value to NVersion value.

C#

```
public static explicit operator NVersion(ushort value);
```

Parameters

Parameters	Description
ushort value	An unsigned short value.

Returns

NVersion value.

7.2.1.1.21.2 NVersion Fields

7.2.1.1.21.2.1 NVersion.Empty Field

New NVersion [1640] object.

C#

```
public static readonly NVersion Empty = new NVersion(0);
```

7.2.1.1.21.3 NVersion Methods

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.21.3.1 NVersion.CompareTo Method

Compares the current version with the specified version and returns an integer that indicates their relative position in the sort order.

C#

```
public int CompareTo(NVersion other);
```

Parameters

Parameters	Description
NVersion other	The other version.

Returns

Integer that indicates their relative position in the sort order.

7.2.1.1.21.3.2 Equals Method

7.2.1.1.21.3.2.1 NVersion.Equals Method (NVersion)

Checks whether the current version equals the specified version.

C#

```
public bool Equals(NVersion other);
```

Parameters

Parameters	Description
NVersion other	The other version.

Returns

true if the versions are equal; else false.

7.2.1.1.21.3.2.2 NVersion.Equals Method (object)

Checks whether the current version equals the specified version object.

C#

```
public override bool Equals(object obj);
```

Parameters

Parameters	Description
object obj	Version object.

Returns

NTrue [1064] if the versions are equal; else NFalse [1041].

7.2.1.1.21.3.3 NVersion.GetHashCode Method

Gets the hash code.

C#

```
public override int GetHashCode();
```

Returns

The hash code.

7.2.1.1.21.3.4 ToString Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.21.3.4.1 NVersion.ToString Method ()

Gets the string representation of the current version.

C#

```
public override string ToString();
```

Returns

A string representing the current version.

7.2.1.1.21.3.4.2 NVersion.ToString Method (string)

Retrieves a string representation of the current version.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	Reserved.

Returns

A string representation of the current version.

7.2.1.1.21.4 NVersion Operators

7.2.1.1.21.4.1 NVersion.!= Operator

Checks whether the two NVersion [1640] values are not equal.

C#

```
public static bool operator !=(NVersion v1, NVersion v2);
```

Parameters

Parameters	Description
NVersion v1	First NVersion [1640] value.
NVersion v2	Second NVersion [1640] value.

Returns

NTrue [1064] if the values are not equal; else NFalse [1041].

7.2.1.1.21.4.2 NVersion.< Operator

Checks whether the first NVersion [1640] value is smaller than the second NVersion [1640] value.

C#

```
public static bool operator <(NVersion v1, NVersion v2);
```

Parameters

Parameters	Description
NVersion v1	First NVersion [1640] value.
NVersion v2	Second NVersion [1640] value.

Returns

>NTrue [1064] if the first NVersion [1640] is smaller; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.21.4.3 NVersion.<= Operator

Checks whether the first NVersion [1640] value is smaller than or equal to the second NVersion [1640] value.

C#

```
public static bool operator <=(NVersion v1, NVersion v2);
```

Parameters

Parameters	Description
NVersion v1	First NVersion [1640] value.
NVersion v2	Second NVersion [1640] value.

Returns

>NTrue [1064] if the first NVersion [1640] is smaller or the values are equal; else NFalse [1041].

7.2.1.1.21.4.4 NVersion.== Operator

Checks whether the two NVersion [1640] values are equal.

C#

```
public static bool operator ==(NVersion v1, NVersion v2);
```

Parameters

Parameters	Description
NVersion v1	First NVersion [1640] value.
NVersion v2	Second NVersion [1640] value.

Returns

NTrue [1064] if the values are equal; else NFalse [1041].

7.2.1.1.21.4.5 NVersion.> Operator

Checks whether the first NVersion [1640] value is larger than the second NVersion [1640] value.

C#

```
public static bool operator >(NVersion v1, NVersion v2);
```

Parameters

Parameters	Description
NVersion v1	First NVersion [1640] value.
NVersion v2	Second NVersion [1640] value.

Returns

>NTrue [1064] if the first NVersion [1640] is larger; else NFalse [1041].

7.2.1.1.21.4.6 NVersion.>= Operator

Checks whether the first NVersion [1640] value is larger than or equal to the second NVersion [1640] value.

C#

```
public static bool operator >=(NVersion v1, NVersion v2);
```

Parameters

Parameters	Description
NVersion v1	First NVersion [1640] value.
NVersion v2	Second NVersion [1640] value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

>NTrue [1064] if the first NVersion [1640] is larger or the values are equal; else NFalse [1041].

7.2.1.1.21.4.7 NVersion.ushort Operator

Converts NVersion [1640] value to an unsigned short value.

C#

```
public static explicit operator ushort(NVersion value);
```

Parameters

Parameters	Description
NVersion value	NVersion [1640] value.

Returns

Unsigned short value.

7.2.1.1.21.5 NVersion Properties

7.2.1.1.21.5.1 NVersion.Major Property

Gets the major version value.

C#

```
public int Major;
```

7.2.1.1.21.5.2 NVersion.Minor Property

Gets the minor version value.

C#

```
public int Minor;
```

7.2.1.1.21.5.3 NVersion.NativeType Property

Defines the native type of the type.

C#

```
public static NType NativeType;
```

7.2.1.1.22 NVersionRange Structure

Manages version ranges.

C#

```
public struct NVersionRange : IFormattable, IEquatable<NVersionRange> {
  public static readonly NVersionRange Empty = new NVersionRange(0);
}
```

NVersionRange Fields

	Name	Description
S R	Empty [1647]	New version range object.

Methods

	Name	Description
	NVersionRange [1646]	Initializes a new instance of NVersionRange structure.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NVersionRange Methods

	Name	Description
	Contains [1647]	Checks whether the current version range contain the specified version.
	Equals [1648]	Checks whether the current version range equals the specified version range.
	GetHashCode [1648]	Gets the hash code.
	Intersect [1648]	Gets the intersection of the current version range and specified version range.
	IntersectsWith [1649]	Checks whether the current version range intersects with the specified version range.
	ToString [1649]	Gets a string representation of the version range.

NVersionRange Operators

	Name	Description
S	!= [1650]	Checks whether the two NVersionRange values are not equal.
S	== [1650]	Checks whether the two NVersionRange values are equal.
S	uint [1650]	Converts NVersionRange value to an unsigned short value.

NVersionRange Properties

	Name	Description
	From [1650]	Lower NVersion [1640] value.
S	NativeType [1650]	Defines the native type of the type.
	To [1651]	Upper NVersion [1640] value.

7.2.1.1.22.1 NVersionRange Operator

7.2.1.1.22.1.1 NVersionRange.NVersionRange Constructor (NVersion, NVersion)

Initializes a new instance of NVersionRange structure.

C#

```
public NVersionRange(NVersion from, NVersion to);
```

Parameters

Parameters	Description
NVersion from	Lower version value.
NVersion to	Upper version value.

7.2.1.1.22.1.2 NVersionRange.NVersionRange Constructor (uint)

Initializes a new instance of NVersionRange structure.

C#

```
public NVersionRange(uint value);
```

Parameters

Parameters	Description
uint value	Version range.

7.2.1.1.22.1.3 NVersionRange.NVersionRange Operator (uint)

Converts an unsigned short value to NVersionRange value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static explicit operator NVersionRange(uint value);
```

Parameters

Parameters	Description
uint value	Unsigned short value.

Returns

NVersionRange value

7.2.1.1.22.2 NVersionRange Fields

7.2.1.1.22.2.1 NVersionRange.Empty Field

New version range object.

C#

```
public static readonly NVersionRange Empty = new NVersionRange(0);
```

7.2.1.1.22.3 NVersionRange Methods

7.2.1.1.22.3.1 Contains Method

7.2.1.1.22.3.1.1 NVersionRange.Contains Method (NVersion)

Checks whether the current version range contain the specified version.

C#

```
public bool Contains(NVersion value);
```

Parameters

Parameters	Description
NVersion value	Version value.

Returns

NTrue [1064] if the version range contain the version value; else NFalse [1041].

7.2.1.1.22.3.1.2 NVersionRange.Contains Method (NVersionRange)

Checks whether the current version range contain the specified version range.

C#

```
public bool Contains(NVersionRange value);
```

Parameters

Parameters	Description
NVersionRange value	Version range value.

Returns

NTrue [1064] if the cuurent version range contain the specified version rang; else NFalse [1041].

7.2.1.1.22.3.2 Equals Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.22.3.2.1 NVersionRange.Equals Method (NVersionRange)

Checks whether the current version range equals the specified version range.

C#

```
public bool Equals(NVersionRange other);
```

Parameters

Parameters	Description
NVersionRange other	The other version range.

Returns

true if the version ranges are equal; else false.

7.2.1.1.22.3.2.2 NVersionRange.Equals Method (object)

Checks whether the current version range equals the specified version range object.

C#

```
public override bool Equals(object obj);
```

Parameters

Parameters	Description
object obj	Version range object.

Returns

NTrue [1064] if version ranges are equal; else NFalse [1041].

7.2.1.1.22.3.3 NVersionRange.GetHashCode Method

Gets the hash code.

C#

```
public override int GetHashCode();
```

Returns

The hash code.

7.2.1.1.22.3.4 Intersect Method

7.2.1.1.22.3.4.1 NVersionRange.Intersect Method (NVersionRange)

Gets the intersection of the current version range and specified version range.

C#

```
public NVersionRange Intersect(NVersionRange value);
```

Parameters

Parameters	Description
NVersionRange value	Version range object.

Returns

Intersection of the version ranges.

7.2.1.1.22.3.4.2 NVersionRange.Intersect Method (NVersionRange, NVersionRange)

Gets the interection of the two specified version ranges.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static NVersionRange Intersect(NVersionRange value1, NVersionRange value2);
```

Parameters

Parameters	Description
NVersionRange value1	First NVersionRange [1645].
NVersionRange value2	Second NVersionRange [1645].

Returns

The resulting NVersionRange [1645].

7.2.1.1.22.3.5 NVersionRange.IntersectsWith Method

Checks whether the current version range intersects with the specified version range.

C#

```
public bool IntersectsWith(NVersionRange value);
```

Parameters

Parameters	Description
NVersionRange value	Version range value.

Returns

NTrue [1064] if the version ranges intersect; else NFalse [1041].

7.2.1.1.22.3.6 ToString Method

7.2.1.1.22.3.6.1 NVersionRange.ToString Method ()

Gets a string representation of the version range.

C#

```
public override string ToString();
```

Returns

A string representing the version range.

7.2.1.1.22.3.6.2 NVersionRange.ToString Method (string)

Retrieves a string representation of the version range.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	Reserved.

Returns

A string representation of version range.

7.2.1.1.22.4 NVersionRange Operators

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.1.1.22.4.1 NVersionRange.!= Operator

Checks whether the two NVersionRange [1645] values are not equal.

C#

```
public static bool operator !=(NVersionRange v1, NVersionRange v2);
```

Parameters

Parameters	Description
NVersionRange v1	First NVersionRange [1645] value.
NVersionRange v2	Second NVersionRange [1645] value.

Returns

NTrue [1064] if the values are not equal; else NFalse [1041].

7.2.1.1.22.4.2 NVersionRange.== Operator

Checks whether the two NVersionRange [1645] values are equal.

C#

```
public static bool operator ==(NVersionRange v1, NVersionRange v2);
```

Parameters

Parameters	Description
NVersionRange v1	First NVersionRange [1645] value.
NVersionRange v2	Second NVersionRange [1645] value.

Returns

NTrue [1064] if the values are equal; else NFalse [1041].

7.2.1.1.22.4.3 NVersionRange.uint Operator

Converts NVersionRange [1645] value to an unsigned short value.

C#

```
public static explicit operator uint(NVersionRange value);
```

Parameters

Parameters	Description
NVersionRange value	NVersionRange [1645] value.

Returns

Unsigned short value.

7.2.1.1.22.5 NVersionRange Properties

7.2.1.1.22.5.1 NVersionRange.From Property

Lower NVersion [1640] value.

C#

```
public NVersion From;
```

7.2.1.1.22.5.2 NVersionRange.NativeType Property

Defines the native type of the type.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static NType NativeType;
```

7.2.1.1.22.5.3 NVersionRange.To Property

Upper NVersion [1640] value.

C#

```
public NVersion To;
```

7.2.1.2 Structs, Records, Enums

7.2.1.2.1 Neurotec.NMemoryType Enumeration

Enumerates memory type options.

C#

```
public enum NMemoryType {
  None = 0,
  Default = 1,
  Aligned = 2,
  Standard = 3,
  Win32Local = 4,
  Win32Global = 5,
  Custom = 15
}
```

Members

Members	Description
None = 0	No memory type.
Default = 1	Default memory type.
Aligned = 2	Aligned memory type.
Standard = 3	Standard memory type.
Win32Local = 4	Win32Local memory type.
Win32Global = 5	Win32Global memory type.
Custom = 15	Custom memory type.

7.2.1.2.2 Neurotec.NModuleOptions Enumeration

Enumerates module options.

C#

```
[Flags]
public enum NModuleOptions {
  None = 0,
  Debug = 0x01,
  Protected = 0x02,
  Unicode = 0x04,
  NoAnsiFunc = 0x08,
  NoUnicode = 0x10,
  Lib = 0x20,
  Exe = 0x40
}
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Members

Members	Description
None = 0	No option for the module.
Debug = 0x01	The module is a debug version.
Protected = 0x02	The module is copy-protected.
Unicode = 0x04	The module in compiled with Unicode in it.
NoAnsiFunc = 0x08	The module has no ANSI versions for function.
NoUnicode = 0x10	The module has no Unicode versions for functions.
Lib = 0x20	The module is in a static library.
Exe = 0x40	The module is in an executable file.

7.2.1.2.3 Neurotec.NProcessorVendor Enumeration

Specifies the processor's vendor name.

C#

```
public enum NProcessorVendor {
  Unknown = 0,
  Amd = 1,
  Centaur = 2,
  Cyrix = 3,
  Intel = 4,
  NationalSemiconductor = 5,
  NexGen = 6,
  RiseTechnology = 7,
  SiS = 8,
  Transmeta = 9,
  Umc = 10,
  Via = 11
}
```

Members

Members	Description
Unknown = 0	The processor's vendor is unknown.
Amd = 1	Advanced Micro Devices [1792], Inc. (AMD).
Centaur = 2	Centaur Technology.
Cyrix = 3	Cyrix.
Intel = 4	Intel Corporation.
NationalSemiconductor = 5	National Semiconductor.
NexGen = 6	NexGen.
RiseTechnology = 7	Rise Technology.
SiS = 8	Silicon Integrated Systems (SIS) Corp.
Transmeta = 9	Transmeta Corporation.
Umc = 10	UMC.
Via = 11	VIA.

7.2.1.2.4 Neurotec.NTypeCode Enumeration

Enumerates type codes.

C#

```
public enum NTypeCode {
  None = 0,
  Other = 1,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
    Byte = 2,
    SByte = 3,
    UInt16 = 4,
    Int16 = 5,
    UInt32 = 6,
    Int32 = 7,
    UInt64 = 8,
    Int64 = 9,
    Single = 10,
    Double = 11,
    Boolean = 12,
    SizeType = 13,
    SSizeType = 14,
    Pointer = 15,
    Result = 16,
    AChar = 17,
    WChar = 18,
    String = 19,
    Object = 20,
    Value = 21,
    Array = 22,
    TimeSpan = 23,
    DateTime = 24,
    URational = 25,
    Rational = 26,
    Complex = 27,
    Guid = 28,
    Buffer = 29
  }
```

Members

Members	Description
None = 0	Empty value.
Other = 1	Other value type.
Byte = 2	A byte.
SByte = 3	Signed byte.
UInt16 = 4	Unsigned 16-bit integer.
Int16 = 5	16-bit integer.
UInt32 = 6	Unsigned 32-bit integer.
Int32 = 7	32-bit integer.
UInt64 = 8	Unsigned 64-bit integer.
Int64 = 9	64-bit integer.
Single = 10	Single value.
Double = 11	Double number.
Boolean = 12	Boolean value.
SizeType = 13	NSizeType [1008] value.
SSizeType = 14	Signed NSizeType [1008] value.
Pointer = 15	Pointer.
Result = 16	NResult [1007] value.
AChar = 17	ASCII character.
WChar = 18	Unicode character.
String = 19	String.
Object = 20	An object.
Value = 21	Custom type value.
Array = 22	Array.
TimeSpan = 23	NTimeSpan [855] value.
DateTime = 24	NDateTime [361] value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

URational = 25	Unsigned rational number.
Rational = 26	Rational number.
Complex = 27	Complex number.
Guid = 28	Global unique identifier.
Buffer = 29	Buffer.

7.2.1.3 Types

7.2.1.3.1 Neurotec.NLibraryGetInfo Type

This delegate supports internal Neurotechnology libraries infrastructure and should not be used directly in your code.

C#

```
public delegate int NLibraryGetInfo(IntPtr pValue);
```

7.2.2 Neurotec.Biometrics Namespace

Class under this namespace provides methods for working with biometrical data.

Classes

	Name	Description
	NBiometricAttributes [1655]	Represents the biometric attributes.
S	NBiometrics [1658]	Provides methods for retrieving NBiometrics Library information.
S	NBiometricTypes [1659]	Defines biometric types used by Neurotechnology components..
	NFCore [1680]	The structure contains information of core.
	NFDelta [1682]	The structure contains information of delta.
	NFDoubleCore [1686]	The structure contains information of double core.
	NFMinutia [1688]	The structure contains information of minutia.
	NFMinutiaNeighbor [1692]	The structure contains information of minutia neighbor.
	NLAttributes [1694]	Provides methods to retrieve face template attributes.
	NleDetectionDetails [1697]	Structure containing face detection results.
	NleEyes [1700]	Structure defining information of an eye pair.
	NleFace [1701]	Structure containing information of a face location in an image.
	NleRotation [1702]	Structure containing information of a face rotation in an image as a deviation from frontal face in degrees.
	NLExtractor [1702]	Provides functionality for extracting Neurotechnology Face Records (NLRecord [1724]) from face images.
	NLFeaturePoint [1721]	Structure defining facial feature point.
	NLMatchingDetails [1723]	Provides functionality to retrieve face templates matching details.
	NLRecord [1724]	Provides functionality for packing, unpacking and editing Neurotechnology Face Records (NLRecords).
	NLTemplate [1730]	Provides functionality for packing, unpacking and editing Neurotechnology Faces [45] Templates (NLTemplates).
	NMatcher [1737]	Provides functionality for comparing all Neurotechnology Templates (NTemplates, NFTemplates, NFRecords, NLTemplates, NLRecords, NETemplates, NERecords), encapsulated in Neurotechnology Matcher (NMatcher) object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NMatchingDetails [1757]	Provides functionality for retrieving Neurotechnology templates matching details.
	NTemplate [1762]	Provides functionality for packing, unpacking and editing Neurotechnology Templates (NTemplates).

Module

.NET Reference [1468]

Structs, Records, Enums

	Name	Description
	NBiometricStatus [1769]	Enumerates biometric status values. These values are returned by NBiometricDevice [1142].
	NBiometricSubtype [1770]	Specifies the subtype of biometric data.
	NBiometricType [1771]	Specifies the type of biometric data.
	NEPosition [1772]	Holds information on iris image position.
	NFImpressionType [1773]	Specifies the impression types.
	NFMinutiaFormat [1774]	Specifies minutia format.
	NFMinutiaType [1774]	Specifies types of minutia.
	NFPatternClass [1775]	Specifies pattern class of the fingerprint.
	NFPosition [1776]	Specifies finger position.
	NFRidgeCountsType [1778]	Specifies type of ridge counts contained in NFRecord.
	NGender [1778]	Enumerates gender constants.
	NIeExtractionStatus [1779]	Specifies face features extraction status.
	NIeTemplateSize [1779]	Specifies NLRecord [1724] type by its size.
	NLExpression [1780]	Enumerates facial expression constants.
	NLProperties [1780]	Enumerates basic face properties constants
	NMatchingSpeed [1781]	Specifies biometric templates matching speed.
	NMFusionType [1781]	Specifies the fingerprints, palmprints, faces and irises matching fusion.

7.2.2.1 Classes

7.2.2.1.1 NBiometricAttributes Class

Represents the biometric attributes.

C#

```
public class NBiometricAttributes : NObject;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	FromObject [1554]	Creates a new NObject [1549] from the specified object.
	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.
S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.
S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	Owner [1568]	Gets owner of the object.

NBiometricAttributes Class

	Name	Description
	BiometricType [1658]	Gets the biometric type.
S	NativeType [1658]	Defines the native type of the object.
	Quality [1658]	Gets or sets the quality.
	Status [1658]	Gets or sets the biometric status.

7.2.2.1.1.1 NBiometricAttributes Properties

7.2.2.1.1.1.1 NBiometricAttributes.BiometricType Property

Gets the biometric type.

C#

```
public NBiometricType BiometricType;
```

7.2.2.1.1.1.2 NBiometricAttributes.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.2.1.1.1.3 NBiometricAttributes.Quality Property

Gets or sets the quality.

C#

```
public byte Quality;
```

7.2.2.1.1.1.4 NBiometricAttributes.Status Property

Gets or sets the biometric status.

C#

```
public NBiometricStatus Status;
```

7.2.2.1.2 NBiometrics Class

Provides methods for retrieving NBiometrics Library information.

C#

```
public static class NBiometrics;
```

NBiometrics Fields

	Name	Description
	DllName [1659]	Name of DLL containing unmanaged part of this class.

NBiometrics Properties

	Name	Description
S	NativeModule [1659]	Gets or sets a pointer to the function that retrieves NModule [1543] object for the specified module name.

7.2.2.1.2.1 NBiometrics Fields

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.2.1.1 NBiometrics.DllName Field

Name of DLL containing unmanaged part of this class.

C#

```
public const string DllName = "NBiometrics";
```

7.2.2.1.2.2 NBiometrics Properties

7.2.2.1.2.2.1 NBiometrics.NativeModule Property

Gets or sets a pointer to the function that retrieves NModule [1543] object for the specified module name.

C#

```
public static NModule NativeModule;
```

Property value

A pointer to the function that retrieves NModule [1543] object for the specified module name.

7.2.2.1.3 NBiometricTypes Class

Defines biometric types used by Neurotechnology components..

C#

```
public static class NBiometricTypes;
```

NBiometricTypes Fields

	Name	Description
	PhraseIdUnknown [1661]	Unknown phrase ID.
	QualityFailed [1661]	Failed quality.
	QualityMax [1661]	Maximal quality.
	QualityMin [1662]	Minimal quality.
	QualityUnknown [1662]	Unknown quality.

NBiometricTypes Methods

	Name	Description
	AngleFromRadians [1662]	Converts the specified angle from radians to degrees.
	AngleToRadians [1662]	Converts the specified angle from degrees to radians.
	AngleToString [1662]	Gets a string representation of the specified angle.
	GetPositionAvailableParts [1663]	Determines available iris positions based on specified missing iris positions and base position.
	IsBiometricStatusFinal [1664]	Checks whether the biometric status is final.
	IsBiometricStatusValid [1664]	Checks whether the value passed as a parameter is defined in NBiometricStatus [1769].
	IsBiometricSubtypeValid [1664]	Checks whether the value passed as a parameter is defined in NBiometricSubtype [1770].
	IsBiometricTypeValid [1664]	Checks whether the value passed as a parameter is defined in NBiometricType [1771].
	IsImpressionTypeCompatibleWith [1665]	Gets the Bool value that specify whether two values are compatible.
	IsImpressionTypeContact [1665]	Checks whether the specified NFImpressionType [1773] value is one of the following: nfitLiveScanOpticalContactPlain, nfitLiveScanOpticalContactRolled, nfitLiveScanNonOpticalContactPlain, nfitLiveScanNonOpticalContactRolled

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	IsImpressionTypeContactless [1665]	Checks if the specified NFImpressionType [1773] value is one of the following: nfitLiveScanContactless, nfitLiveScanOpticalContactlessPlain, nfitLiveScanOpticalContactlessRolled, nfitLiveScanNonOpticalContactlessPlain, nfitLiveScanNonOpticalContactlessRolled.
	IsImpressionTypeFinger [1665]	Checks whether the specified value is fingerprint.
	IsImpressionTypeGeneric [1666]	Checks whether the impression type is genertic.
	IsImpressionTypeLatent [1666]	Checks whether the specified value is a latent scan.
	IsImpressionTypeLiveScan [1666]	Checks whether the specified value is a live scan.
	IsImpressionTypeNonliveScan [1666]	Checks whether the specified value is a non-live scan.
	IsImpressionTypeNonOptical [1667]	Checks whether the specified NFImpressionType [1773] value is one of the following: nfitLiveScanNonOpticalContactPlain, nfitLiveScanNonOpticalContactRolled, nfitLiveScanNonOpticalContactlessPlain, nfitLiveScanNonOpticalContactlessRolled.
	IsImpressionTypeOneOf [1667]	Checks whether specified NFImpressionType [1773] value matches one in an array.
	IsImpressionTypeOptical [1667]	Checks if the specified NFImpressionType [1773] value is one of the following: nfitLiveScanOpticalContactPlain, nfitLiveScanOpticalContactRolled, nfitLiveScanOpticalContactlessPlain, nfitLiveScanOpticalContactlessRolled.
	IsImpressionTypePalm [1667]	Checks whether the specified value is a palmprint.
	IsImpressionTypePlain [1668]	Checks whether the specified value is a plain scan or optical contact/contactless.
	IsImpressionTypeRolled [1668]	Checks whether the specified value is one of rolled biometric types.
	IsImpressionTypeSwipe [1668]	Checks whether the specified NFImpressionType [1773] value is one of the following: nfitLiveScanVerticalSwipe.
	IsImpressionTypeValid [1668]	Checks whether the value passed as a parameter is a valid NFImpressionType [1773] value.
	IsImpressionTypeValidFinger [1669]	Checks whether the specified value is a valid fingerprint.
	IsImpressionTypeValidPalm [1669]	Checks whether the specified value is a valid palmprint.
	IsPositionCompatibleWith [1669]	Gets the Bool value that specify whether two values are compatible.
	IsPositionFinger [1670]	Checks whether the specified value is fingerprint.
	IsPositionFourFingers [1670]	Checks if the specified value is a type of 4 fingerprints data.
	IsPositionKnown [1670]	Checks whether the specified position is known.
	IsPositionLeft [1671]	Checks whether the specified position is left.
	IsPositionLeftAndRight [1671]	Checks whether the specified position is left and right fingerprint.
	IsPositionOneOf [1672]	Checks whether specified NEPosition [1772] value matches one in an array.
	IsPositionPalm [1672]	Checks whether the specified value is a palmprint.
	IsPositionRight [1673]	Checks whether the specified position is right.
	IsPositionRightAndLeft [1673]	Checks whether the specified position is both right and left.
	IsPositionSingleEye [1673]	Checks whether the specified position represents a single eye.
	IsPositionSingleFinger [1674]	Checks whether the specified value is a single fingerprint.
	IsPositionTheEye [1674]	Checks whether the specified position represents an eye.
	IsPositionTheFinger [1674]	Checks whether the specified value is the fingerprint.
	IsPositionThePalm [1674]	Checks whether the specified value is a palmprint.
	IsPositionThreeFingers [1675]	Checks whether the specified value is a type of 3 fingerprints data.
	IsPositionTwoEyes [1675]	Checks whether the specified position represents two eyes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	IsPositionTwoFingers [1675]	Checks whether the specified value is a type of 2 fingerprints data.
	IsPositionValid [1675]	Checks whether the value passed as a parameter is a valid NEPosition [1772] value.
	IsPositionValidFinger [1676]	Checks whether the specified value is a valid fingerprint.
	IsPositionValidPalm [1676]	Checks whether the specified value is a valid palmprint.
	IsPositionValidTheEye [1676]	Checks whether the specified value is a valid NEPosition [1772] value.
	IsPositionValidTheFinger [1677]	Checks whether the specified value is a valid fingerprint.
	IsQualityValid [1677]	Checkes whether the quality value is valid.
	QualityToString [1677]	Gets a string representation of the specified quality value.
	ToImpressionTypeFinger [1678]	Converts an impression type to finger impression type.

NBiometricTypes Properties

	Name	Description
	NativeType [1678]	Defines the native type of the object.
	NBiometricStatusNativeType [1678]	Defines the native type of the biometric status.
	NBiometricSubtypeNativeType [1678]	Defines the native type of the biometric subtype.
	NBiometricTypeNativeType [1678]	Defines the native type of the biometric type.
	NEPositionNativeType [1678]	Defines the native type of the iris position.
	NFImpressionTypeNativeType [1679]	Defines the native type of the finger impression type.
	NFMinutiaFormatNativeType [1679]	Defines the native type of the fingerprint minutia format.
	NFMinutiaTypeNativeType [1679]	Defines the native type of the fingerprint minutia type.
	NFPatternClassNativeType [1679]	Defines the native type of the pattern class.
	NFPositionNativeType [1679]	Defines the native type of the finger position.
	NFRidgeCountsTypeNativeType [1679]	Defines the native type of the fingerprint ridge count.
	NGenderNativeType [1679]	Defines the native type of the gender.
	NLExpressionNativeType [1679]	Defines the native type of the face expression.
	NLPropertiesNativeType [1679]	Defines the native type of the face properties.

7.2.2.1.3.1 NBiometricTypes Fields

7.2.2.1.3.1.1 NBiometricTypes.PhraseIdUnknown Field

Unknown phrase ID.

C#

```
public const int PhraseIdUnknown = 0;
```

7.2.2.1.3.1.2 NBiometricTypes.QualityFailed Field

Failed quality.

C#

```
public const byte QualityFailed = 255;
```

7.2.2.1.3.1.3 NBiometricTypes.QualityMax Field

Maximal quality.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public const byte QualityMax = 100;
```

7.2.2.1.3.1.4 NBiometricTypes.QualityMin Field

Minimal quality.

C#

```
public const byte QualityMin = 0;
```

7.2.2.1.3.1.5 NBiometricTypes.QualityUnknown Field

Unknown quality.

C#

```
public const byte QualityUnknown = 254;
```

7.2.2.1.3.2 NBiometricTypes Methods

7.2.2.1.3.2.1 NBiometricTypes.AngleFromRadians Method

Converts the specified angle from radians to degrees.

C#

```
public static int AngleFromRadians(double value);
```

Parameters

Parameters	Description
double value	Radian value of the angle.

Returns

Degree value of the angle.

7.2.2.1.3.2.2 NBiometricTypes.AngleToRadians Method

Converts the specified angle from degrees to radians.

C#

```
public static double AngleToRadians(int value);
```

Parameters

Parameters	Description
int value	Degree value of the angle.

Returns

Radian value of the angle.

7.2.2.1.3.2.3 AngleToString Method

7.2.2.1.3.2.3.1 NBiometricTypes.AngleToString Method (int)

Gets a string representation of the specified angle.

C#

```
public static string AngleToString(int value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
int value	Degree value of the angle.

Returns

A string representing the angle.

7.2.2.1.3.2.3.2 NBiometricTypes.AngleToString Method (int, string)

Gets a string representation of the specified angle.

C#

```
public static string AngleToString(int value, string format);
```

Parameters

Parameters	Description
int value	Degree value of the angle.
string format	Conversion format.

Returns

A string representing the angle.

7.2.2.1.3.2.4 GetPositionAvailableParts Method

7.2.2.1.3.2.4.1 NBiometricTypes.GetPositionAvailableParts Method (NEPosition, NEPosition[])

Determines available iris positions based on specified missing iris positions and base position.

C#

```
public static NEPosition[] GetPositionAvailableParts(NEPosition value, NEPosition[]
missingPositions);
```

Parameters

Parameters	Description
NEPosition value	NEPosition [1772] object, representing the general position.
NEPosition[] missingPositions	An array of NEPosition [1772] objects, representing missing position parts.

Returns

An array of NEPosition [1772] objects, representing available position parts.

7.2.2.1.3.2.4.2 NBiometricTypes.GetPositionAvailableParts Method (NFPosition, NFPosition[])

Determines available finger positions based on specified missing finger positions and base position.

C#

```
public static NFPosition[] GetPositionAvailableParts(NFPosition value, NFPosition[]
missingPositions);
```

Parameters

Parameters	Description
NFPosition value	NFPosition [1776] object, representing the general position.
NFPosition[] missingPositions	An array of NFPosition [1776] objects, representing missing position parts.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

An array of NFPosition [1776] objects, representing available position parts.

7.2.2.1.3.2.5 NBiometricTypes.IsBiometricStatusFinal Method

Checks whether the biometric status is final.

C#

```
public static bool IsBiometricStatusFinal(NBiometricStatus value);
```

Parameters

Parameters	Description
NBiometricStatus value	One of NBiometricStatus [1769] values to check.

Returns

NTrue [1064] if the value is final; else NFalse [1041].

7.2.2.1.3.2.6 NBiometricTypes.IsBiometricStatusValid Method

Checks whether the value passed as a parameter is defined in NBiometricStatus [1769].

C#

```
public static bool IsBiometricStatusValid(NBiometricStatus value);
```

Parameters

Parameters	Description
NBiometricStatus value	One of NBiometricStatus [1769] values to check.

Returns

Bool type value.

7.2.2.1.3.2.7 NBiometricTypes.IsBiometricSubtypeValid Method

Checks whether the value passed as a parameter is defined in NBiometricSubtype [1770].

C#

```
public static bool IsBiometricSubtypeValid(NBiometricSubtype value);
```

Parameters

Parameters	Description
NBiometricSubtype value	One of NBiometricSubtype [1770] values to check.

Returns

Bool type value.

7.2.2.1.3.2.8 NBiometricTypes.IsBiometricTypeValid Method

Checks whether the value passed as a parameter is defined in NBiometricType [1771].

C#

```
public static bool IsBiometricTypeValid(NBiometricType value);
```

Parameters

Parameters	Description
NBiometricType value	One of NBiometricType [1771] values to check.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Bool type value.

7.2.2.1.3.2.9 NBiometricTypes.IsImpressionTypeCompatibleWith Method

Gets the Bool value that specify whether two values are compatible.

C#

```
public static bool IsImpressionTypeCompatibleWith(NFImpressionType value, NFImpressionType
otherValue);
```

Parameters

Parameters	Description
NFImpressionType value	Value to check if it is compatible with other value.
NFImpressionType otherValue	Value which is used to check compatibility with value.

Returns

A Bool value that specifies whether two values are compatible.

7.2.2.1.3.2.10 NBiometricTypes.IsImpressionTypeContact Method

Checks whether the specified NFImpressionType [1773] value is one of the following: nfitLiveScanOpticalContactPlain, nfitLiveScanOpticalContactRolled, nfitLiveScanNonOpticalContactPlain, nfitLiveScanNonOpticalContactRolled

C#

```
public static bool IsImpressionTypeContact(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	NFImpressionType [1773] value to check.

Returns

A Bool value that specifies whether the specified value is a contact type.

7.2.2.1.3.2.11 NBiometricTypes.IsImpressionTypeContactless Method

Checks if the specified NFImpressionType [1773] value is one of the following: nfitLiveScanContactless, nfitLiveScanOpticalContactlessPlain, nfitLiveScanOpticalContactlessRolled, nfitLiveScanNonOpticalContactlessPlain, nfitLiveScanNonOpticalContactlessRolled.

C#

```
public static bool IsImpressionTypeContactless(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	NFImpressionType [1773] value to check.

Returns

A Bool value that specifies whether the specified value is a contactless type.

7.2.2.1.3.2.12 NBiometricTypes.IsImpressionTypeFinger Method

Checks whether the specified value is fingerprint.

C#

```
public static bool IsImpressionTypeFinger(NFImpressionType value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NFImpressionType value	NFImpressionType [1773] value to check its type.

Returns

A Bool value that specifies whether the value is a fingerprint.

7.2.2.1.3.2.13 NBiometricTypes.IsImpressionTypeGeneric Method

Checks whether the impression type is genertic.

C#

```
public static bool IsImpressionTypeGeneric(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	Impression type value.

Returns

NTrue [1064] if the value is generic; else NFalse [1041].

7.2.2.1.3.2.14 NBiometricTypes.IsImpressionTypeLatent Method

Checks whether the specified value is a latent scan.

C#

```
public static bool IsImpressionTypeLatent(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	NFImpressionType [1773] value to check its type.

Returns

Bool type value.

7.2.2.1.3.2.15 NBiometricTypes.IsImpressionTypeLiveScan Method

Checks whether the specified value is a live scan.

C#

```
public static bool IsImpressionTypeLiveScan(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	NFImpressionType [1773] value to check its type.

Returns

Bool type value.

7.2.2.1.3.2.16 NBiometricTypes.IsImpressionTypeNonliveScan Method

Checks whether the specified value is a non-live scan.

C#

```
public static bool IsImpressionTypeNonliveScan(NFImpressionType value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NFImpressionType value	NFImpressionType [1773] value to check its type.

Returns

Bool type value specifying whether the value is a non-live scan.

7.2.2.1.3.2.17 NBiometricTypes.IsImpressionTypeNonOptical Method

Checks whether the specified NFImpressionType [1773] value is one of the following: nfitLiveScanNonOpticalContactPlain, nfitLiveScanNonOpticalContactRolled, nfitLiveScanNonOpticalContactlessPlain, nfitLiveScanNonOpticalContactlessRolled.

C#

```
public static bool IsImpressionTypeNonOptical(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	NFImpressionType [1773] value to check.

Returns

Bool type value.

7.2.2.1.3.2.18 NBiometricTypes.IsImpressionTypeOneOf Method

Checks whether specified NFImpressionType [1773] value matches one in an array.

C#

```
public static bool IsImpressionTypeOneOf(NFImpressionType value, NFImpressionType[]
supportedImpressionTypes);
```

Parameters

Parameters	Description
NFImpressionType value	One of the NFImpressionType [1773] values to check.
NFImpressionType[] supportedImpressionTypes	NFImpressionType [1773] values array.

Returns

Bool type value specifying whether value is on of the values in an array.

7.2.2.1.3.2.19 NBiometricTypes.IsImpressionTypeOptical Method

Checks if the specified NFImpressionType [1773] value is one of the following: nfitLiveScanOpticalContactPlain, nfitLiveScanOpticalContactRolled, nfitLiveScanOpticalContactlessPlain, nfitLiveScanOpticalContactlessRolled.

C#

```
public static bool IsImpressionTypeOptical(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	NFImpressionType [1773] value to check.

Returns

Bool type value.

7.2.2.1.3.2.20 NBiometricTypes.IsImpressionTypePalm Method

Checks whether the specified value is a palmprint.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static bool IsImpressionTypePalm(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	NFImpressionType [1773] value to check its type.

Returns

Bool type value.

7.2.2.1.3.2.21 NBiometricTypes.IsImpressionTypePlain Method

Checks whether the specified value is a plain scan or optical contact/contactless.

C#

```
public static bool IsImpressionTypePlain(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	NFImpressionType [1773] value to check its type.

Returns

Bool type value.

7.2.2.1.3.2.22 NBiometricTypes.IsImpressionTypeRolled Method

Checks whether the specified value is one of rolled biometric types.

C#

```
public static bool IsImpressionTypeRolled(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	NFImpressionType [1773] value to check its type.

Returns

Bool type value.

7.2.2.1.3.2.23 NBiometricTypes.IsImpressionTypeSwipe Method

Checks whether the specified NFImpressionType [1773] value is one of the following: nfitLiveScanVerticalSwipe.

C#

```
public static bool IsImpressionTypeSwipe(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	NFImpressionType [1773] value to check.

Returns

Bool type value.

7.2.2.1.3.2.24 NBiometricTypes.IsImpressionTypeValid Method

Checks whether the value passed as a parameter is a valid NFImpressionType [1773] value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static bool IsImpressionTypeValid(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	One of NFImpressionType [1773] values to check.

Returns

Bool type value.

7.2.2.1.3.2.25 NBiometricTypes.IsImpressionTypeValidFinger Method

Checks whether the specified value is a valid fingerprint.

C#

```
public static bool IsImpressionTypeValidFinger(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	One of NFImpressionType [1773] values to check.

Returns

Bool type value.

7.2.2.1.3.2.26 NBiometricTypes.IsImpressionTypeValidPalm Method

Checks whether the specified value is a valid palmprint.

C#

```
public static bool IsImpressionTypeValidPalm(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	One of NFImpressionType [1773] values to check.

Returns

Bool type value.

7.2.2.1.3.2.27 IsPositionCompatibleWith Method

7.2.2.1.3.2.27.1 NBiometricTypes.IsPositionCompatibleWith Method (NFPosition, NFImpressionType)

Gets the Bool value that specify whether two values are compatible.

C#

```
public static bool IsPositionCompatibleWith(NFPosition value, NFImpressionType imp);
```

Parameters

Parameters	Description
NFPosition value	Value to check if it is compatible with other value.
NFImpressionType imp	Value which is used to check compatibility with value.

Returns

A Bool value that specifies whether two values are compatible.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.3.2.27.2 NBiometricTypes.IsPositionCompatibleWith Method (NFPosition, NFPosition)

Checks whether two position value are compatible.

C#

```
public static bool IsPositionCompatibleWith(NFPosition value, NFPosition otherValue);
```

Parameters

Parameters	Description
NFPosition value	First position value.
NFPosition otherValue	Second position value.

Returns

NTrue [1064] if the values are compatible; else NFalse [1041].

7.2.2.1.3.2.28 NBiometricTypes.IsPositionFinger Method

Checks whether the specified value is fingerprint.

C#

```
public static bool IsPositionFinger(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	NFPosition [1776] value to check its type.

Returns

A Bool value that specifies whether the value is a fingerprint.

7.2.2.1.3.2.29 NBiometricTypes.IsPositionFourFingers Method

Checks if the specified value is a type of 4 fingerprints data.

C#

```
public static bool IsPositionFourFingers(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	Value to check its type.

Returns

A bool type value.

7.2.2.1.3.2.30 IsPositionKnown Method

7.2.2.1.3.2.30.1 NBiometricTypes.IsPositionKnown Method (NEPosition)

Checks whether the specified position is known.

C#

```
public static bool IsPositionKnown(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	The position to check.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NTrue [1064] if the position is known; else NFalse [1041].

7.2.2.1.3.2.30.2 NBiometricTypes.IsPositionKnown Method (NFPosition)

Checks whether the record contains biometric data that type is one of the known fingerprint types (thumb, palmprint, etc.).

C#

```
public static bool IsPositionKnown(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	Value to check its type.

Returns

Bool value - check result.

7.2.2.1.3.2.31 IsPositionLeft Method

7.2.2.1.3.2.31.1 NBiometricTypes.IsPositionLeft Method (NEPosition)

Checks whether the specified position is left.

C#

```
public static bool IsPositionLeft(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	The position to check.

Returns

NTrue [1064] if the position is left; else NFalse [1041].

7.2.2.1.3.2.31.2 NBiometricTypes.IsPositionLeft Method (NFPosition)

Checks whether the specified position is left fingerprint.

C#

```
public static bool IsPositionLeft(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	NFImpressionType [1773] value to check its type.

Returns

Bool type value.

7.2.2.1.3.2.32 NBiometricTypes.IsPositionLeftAndRight Method

Checks whether the specified position is left and right fingerprint.

C#

```
public static bool IsPositionLeftAndRight(NFPosition value);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NFPosition value	Fingerprint position to check its type.

Returns

Bool type value.

7.2.2.1.3.2.33 IsPositionOneOf Method

7.2.2.1.3.2.33.1 NBiometricTypes.IsPositionOneOf Method (NEPosition, NEPosition[])

Checks whether specified NEPosition [1772] value matches one in an array.

C#

```
public static bool IsPositionOneOf(NEPosition value, NEPosition[] supportedPositions);
```

Parameters

Parameters	Description
NEPosition value	One of the NEPosition [1772] values to check.
NEPosition[] supportedPositions	NEPosition [1772] values array.

Returns

Bool type value specifying whether NEPosition [1772] value is on of the values in an array.

7.2.2.1.3.2.33.2 NBiometricTypes.IsPositionOneOf Method (NFPosition, NFPosition[])

Checks whether specified NFPosition [1776] value matches one in an array.

C#

```
public static bool IsPositionOneOf(NFPosition value, NFPosition[] supportedPositions);
```

Parameters

Parameters	Description
NFPosition value	One of the NFPosition [1776] values to check.
NFPosition[] supportedPositions	NFPosition [1776] values array.

Returns

Bool type value specifying whether NFPosition [1776] value is on of the values in an array.

7.2.2.1.3.2.34 NBiometricTypes.IsPositionPalm Method

Checks whether the specified value is a palmprint.

C#

```
public static bool IsPositionPalm(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	NFPosition [1776] value to check its type.

Returns

Bool type value.

7.2.2.1.3.2.35 IsPositionRight Method

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.3.2.35.1 NBiometricTypes.IsPositionRight Method (NEPosition)

Checks whether the specified position is right.

C#

```
public static bool IsPositionRight(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	

Returns

NTrue [1064] if the position is right; else NFalse [1041].

7.2.2.1.3.2.35.2 NBiometricTypes.IsPositionRight Method (NFPosition)

Checks whether the specified position is right fingerprint.

C#

```
public static bool IsPositionRight(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	Value to check its type.

Returns

Bool type value.

7.2.2.1.3.2.36 NBiometricTypes.IsPositionRightAndLeft Method

Checks whether the specified position is both right and left.

C#

```
public static bool IsPositionRightAndLeft(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	The position to check.

Returns

NTrue [1064] if the position is both right and left.; else NFalse [1041].

7.2.2.1.3.2.37 NBiometricTypes.IsPositionSingleEye Method

Checks whether the specified position represents a single eye.

C#

```
public static bool IsPositionSingleEye(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	The position to check.

Returns

NTrue [1064] if the position represents a single eye; else NFalse [1041].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.3.2.38 NBiometricTypes.IsPositionSingleFinger Method

Checks whether the specified value is a single fingerprint.

C#

```
public static bool IsPositionSingleFinger(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	NFPosition [1776] value to check its type.

Returns

Bool type value.

7.2.2.1.3.2.39 NBiometricTypes.IsPositionTheEye Method

Checks whether the specified position represents an eye.

C#

```
public static bool IsPositionTheEye(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	The position to check.

Returns

NTrue [1064] if the position represents an eye; else NFalse [1041].

7.2.2.1.3.2.40 NBiometricTypes.IsPositionTheFinger Method

Checks whether the specified value is the fingerprint.

C#

```
public static bool IsPositionTheFinger(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	NFPosition [1776] value to check its type.

Returns

Bool type value.

7.2.2.1.3.2.41 NBiometricTypes.IsPositionThePalm Method

Checks whether the specified value is a palmprint.

C#

```
public static bool IsPositionThePalm(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	NFPosition [1776] value to check its type.

Returns

Bool type value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.3.2.42 NBiometricTypes.IsPositionThreeFingers Method

Checks whether the specified value is a type of 3 fingerprints data.

C#

```
public static bool IsPositionThreeFingers(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	Value to check its type.

Returns

Bool type value.

7.2.2.1.3.2.43 NBiometricTypes.IsPositionTwoEyes Method

Checks whether the specified position represents two eyes.

C#

```
public static bool IsPositionTwoEyes(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	The position to check.

Returns

NTrue [1064] if the position represents two eyes; else NFalse [1041].

7.2.2.1.3.2.44 NBiometricTypes.IsPositionTwoFingers Method

Checks whether the specified value is a type of 2 fingerprints data.

C#

```
public static bool IsPositionTwoFingers(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	Value to check its type.

Returns

Bool type value.

7.2.2.1.3.2.45 IsPositionValid Method

7.2.2.1.3.2.45.1 NBiometricTypes.IsPositionValid Method (NEPosition)

Checks whether the value passed as a parameter is a valid NEPosition [1772] value.

C#

```
public static bool IsPositionValid(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	One of NEPosition [1772] values to check.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Bool type value.

7.2.2.1.3.2.45.2 NBiometricTypes.IsPositionValid Method (NFPosition)

Checks whether the value passed as a parameter is a valid NFPosition [1776] value.

C#

```
public static bool IsPositionValid(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	One of NFPosition [1776] values to check.

Returns

Bool type value.

7.2.2.1.3.2.46 NBiometricTypes.IsPositionValidFinger Method

Checks whether the specified value is a valid fingerprint.

C#

```
public static bool IsPositionValidFinger(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	One of NFPosition [1776] values to check.

Returns

Bool type value.

7.2.2.1.3.2.47 NBiometricTypes.IsPositionValidPalm Method

Checks whether the specified value is a valid palmprint.

C#

```
public static bool IsPositionValidPalm(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	One of NFPosition [1776] values to check.

Returns

Bool type value.

7.2.2.1.3.2.48 NBiometricTypes.IsPositionValidTheEye Method

Checks whether the specified value is a valid NEPosition [1772] value.

C#

```
public static bool IsPositionValidTheEye(NEPosition value);
```

Parameters

Parameters	Description
NEPosition value	NEPosition [1772] value to check.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Bool type value.

7.2.2.1.3.2.49 NBiometricTypes.IsPositionValidTheFinger Method

Checks whether the specified value is a valid fingerprint.

C#

```
public static bool IsPositionValidTheFinger(NFPosition value);
```

Parameters

Parameters	Description
NFPosition value	One of NFPosition [1776] values to check.

Returns

Bool type value.

7.2.2.1.3.2.50 NBiometricTypes.IsQualityValid Method

Checkes whether the quality value is valid.

C#

```
public static bool IsQualityValid(byte value);
```

Parameters

Parameters	Description
byte value	The quality value to checks.

Returns

NTrue [1064] if the quality value is valid; else NFalse [1041].

7.2.2.1.3.2.51 QualityToString Method

7.2.2.1.3.2.51.1 NBiometricTypes.QualityToString Method (byte)

Gets a string representation of the specified quality value.

C#

```
public static string QualityToString(byte value);
```

Parameters

Parameters	Description
byte value	Quality value.

Returns

A string representation of the quality.

7.2.2.1.3.2.51.2 NBiometricTypes.QualityToString Method (byte, string)

Gets a string representation of the specified quality value.

C#

```
public static string QualityToString(byte value, string format);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
byte value	Quality value.
string format	Conversion format.

Returns

A string representation of the quality.

7.2.2.1.3.2.52 NBiometricTypes.ToImpressionTypeFinger Method

Converts an impression type to finger impression type.

C#

```
public static NFImpressionType ToImpressionTypeFinger(NFImpressionType value);
```

Parameters

Parameters	Description
NFImpressionType value	Impression type to convert.

Returns

Finger impression type.

7.2.2.1.3.3 NBiometricTypes Properties

7.2.2.1.3.3.1 NBiometricTypes.NativeType Property

Defines the native type of the object.

C#

```
public static NType NativeType;
```

7.2.2.1.3.3.2 NBiometricTypes.NBiometricStatusNativeType Property

Defines the native type of the biometric status.

C#

```
public static NType NBiometricStatusNativeType;
```

7.2.2.1.3.3.3 NBiometricTypes.NBiometricSubtypeNativeType Property

Defines the native type of the biometric subtype.

C#

```
public static NType NBiometricSubtypeNativeType;
```

7.2.2.1.3.3.4 NBiometricTypes.NBiometricTypeNativeType Property

Defines the native type of the biometric type.

C#

```
public static NType NBiometricTypeNativeType;
```

7.2.2.1.3.3.5 NBiometricTypes.NEPositionNativeType Property

Defines the native type of the iris position.

C#

```
public static NType NEPositionNativeType;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.3.3.6 NBiometricTypes.NFImpressionTypeNativeType Property

Defines the native type of the finger impression type.

C#

```
public static NType NFImpressionTypeNativeType;
```

7.2.2.1.3.3.7 NBiometricTypes.NFMinutiaFormatNativeType Property

Defines the native type of the fingerprint minutia format.

C#

```
public static NType NFMinutiaFormatNativeType;
```

7.2.2.1.3.3.8 NBiometricTypes.NFMinutiaTypeNativeType Property

Defines the native type of the fingerprint minutia type.

C#

```
public static NType NFMinutiaTypeNativeType;
```

7.2.2.1.3.3.9 NBiometricTypes.NFPatternClassNativeType Property

Defines the native type of the pattern class.

C#

```
public static NType NFPatternClassNativeType;
```

7.2.2.1.3.3.10 NBiometricTypes.NFPositionNativeType Property

Defines the native type of the finger position.

C#

```
public static NType NFPositionNativeType;
```

7.2.2.1.3.3.11 NBiometricTypes.NFRidgeCountsTypeNativeType Property

Defines the native type of the fingerprint ridge count.

C#

```
public static NType NFRidgeCountsTypeNativeType;
```

7.2.2.1.3.3.12 NBiometricTypes.NGenderNativeType Property

Defines the native type of the gender.

C#

```
public static NType NGenderNativeType;
```

7.2.2.1.3.3.13 NBiometricTypes.NLExpressionNativeType Property

Defines the native type of the face expression.

C#

```
public static NType NLExpressionNativeType;
```

7.2.2.1.3.3.14 NBiometricTypes.NLPropertiesNativeType Property

Defines the native type of the face properties.

C#

```
public static NType NLPropertiesNativeType;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.4 NFCore Structure

The structure contains information of core.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NFCore : IFormattable {
}
```

Methods

	Name	Description
	NFCore [1680]	Initializes a new instance of the NFCore structure.

NFCore Methods

	Name	Description
	ToString [1681]	Retrieves string representation of the specified object.

NFCore Properties

	Name	Description
	Angle [1681]	Gets or sets Angle of core.
S	NativeType [1681]	Defines the native type of the object.
	RawAngle [1682]	Gets or sets raw angle of core
	X [1682]	Gets or sets x coordinate of core.
	Y [1682]	Gets or sets y coordinate of core.

7.2.2.1.4.1 NFCore Constructor

7.2.2.1.4.1.1 NFCore.NFCore Constructor (ushort, ushort)

Initializes a new instance of the NFCore structure.

C#

```
public NFCore(ushort x, ushort y);
```

Parameters

Parameters	Description
ushort x	The x - coordinate of the core.
ushort y	The y - coordinate of the core.

7.2.2.1.4.1.2 NFCore.NFCore Constructor (ushort, ushort, double)

Initializes a new instance of the NFCore class.

C#

```
public NFCore(ushort x, ushort y, double angle);
```

Parameters

Parameters	Description
ushort x	The x - coordinate of the core.
ushort y	The y - coordinate of the core.
double angle	The angle of the core.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.4.1.3 NFCore.NFCore Constructor (ushort, ushort, int)

Initializes a new instance of the NFCore class.

C#

```
public NFCore(ushort x, ushort y, int angle);
```

Parameters

Parameters	Description
ushort x	The x - coordinate of the core.
ushort y	The y - coordinate of the core.
int angle	The angle of the core.

7.2.2.1.4.2 NFCore Methods

7.2.2.1.4.2.1 ToString Method

7.2.2.1.4.2.1.1 NFCore.ToString Method ()

Retrieves string representation of the specified object.

C#

```
public override string ToString();
```

Returns

String representation of an object.

7.2.2.1.4.2.1.2 NFCore.ToString Method (string)

Retrieves string representation of the specified object.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	Reserved for future use.

Returns

No tables or formatting elements here!!!

7.2.2.1.4.3 NFCore Properties

7.2.2.1.4.3.1 NFCore.Angle Property

Gets or sets Angle of core.

C#

```
public double Angle;
```

Property value

The angle of the core.

7.2.2.1.4.3.2 NFCore.NativeType Property

Defines the native type of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static NType NativeType;
```

7.2.2.1.4.3.3 NFCore.RawAngle Property

Gets or sets raw angle of core

C#

```
public int RawAngle;
```

Remarks

The angle of the core is specified in 180/128 degrees units in counterclockwise order and cannot be less than zero or greater than 256 minus one. The value of -1 can be specified if the angle of the core is unknown.

Property value

The raw angle of the core.

7.2.2.1.4.3.4 NFCore.X Property

Gets or sets x coordinate of core.

C#

```
public ushort X;
```

Remarks

The x coordinate of the core is specified in pixels at resolution and "X * [NFRecord horizontal resolution] / Resolution" cannot be greater than MaxDimension or NFRecord width minus one.

Property value

The x coordinate of the core.

7.2.2.1.4.3.5 NFCore.Y Property

Gets or sets y coordinate of core.

C#

```
public ushort Y;
```

Remarks

The y coordinate of the core is specified in pixels at resolution and "Y * [NFRecord vertical resolution] / Resolution" cannot be greater than MaxDimension or NFRecord width minus one.

Property value

The y coordinate of the core.

7.2.2.1.5 NFDelta Structure

The structure contains information of delta.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NFDelta : IFormattable {
}
```

Methods

	Name	Description
	NFDelta [1683]	Initializes a new instance of the NFDelta structure.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFDelta Methods

	Name	Description
	ToString [1684]	Returns a string that represents the current NFDelta object.

NFDelta Properties

	Name	Description
	Angle1 [1684]	Gets or sets the first angle of delta.
	Angle2 [1685]	Gets or sets the second angle of delta.
	Angle3 [1685]	Gets or sets the third angle of delta.
S	NativeType [1685]	Defines the native type of the object.
	RawAngle1 [1685]	Gets or sets the first row angle of the delta.
	RawAngle2 [1685]	Gets or sets the second row angle of the delta.
	RawAngle3 [1685]	Gets or sets the third row angle of the delta.
	X [1686]	Gets or sets x coordinate of delta.
	Y [1686]	Gets or sets y coordinate of delta.

7.2.2.1.5.1 NFDelta Constructor

7.2.2.1.5.1.1 NFDelta.NFDelta Constructor (ushort, ushort)

Initializes a new instance of the NFDelta structure.

C#

```
public NFDelta(ushort x, ushort y);
```

Parameters

Parameters	Description
ushort x	The x - coordinate of the delta.
ushort y	The y - coordinate of the delta.

7.2.2.1.5.1.2 NFDelta.NFDelta Constructor (ushort, ushort, double, double, double)

Initializes a new instance of the NFDelta structure.

C#

```
public NFDelta(ushort x, ushort y, double angle1, double angle2, double angle3);
```

Parameters

Parameters	Description
ushort x	The x - coordinate of the delta.
ushort y	The y - coordinate of the delta.
double angle1	The first angle of the delta.
double angle2	The second angle of the delta.
double angle3	The third angle of the delta.

7.2.2.1.5.1.3 NFDelta.NFDelta Constructor (ushort, ushort, int, int, int)

Initializes a new instance of the NFDelta structure.

C#

```
public NFDelta(ushort x, ushort y, int angle1, int angle2, int angle3);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
ushort x	The x - coordinate of the delta.
ushort y	The y - coordinate of the delta.
int angle1	The first angle of the delta.
int angle2	The second angle of the delta.
int angle3	The third angle of the delta.

7.2.2.1.5.2 NFDelta Methods

7.2.2.1.5.2.1 ToString Method

7.2.2.1.5.2.1.1 NFDelta.ToString Method ()

Returns a string that represents the current NFDelta [1682] object.

C#

```
public override string ToString();
```

Returns

String that represents NFDelta [1682] object.

7.2.2.1.5.2.1.2 NFDelta.ToString Method (string)

Returns a string that represents the current NFDelta [1682] object.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	Reserved for future use..

Returns

String that represents NFDelta [1682] object.

7.2.2.1.5.3 NFDelta Properties

7.2.2.1.5.3.1 NFDelta.Angle1 Property

Gets or sets the first angle of delta.

C#

```
public double Angle1;
```

Remarks

The angle of the delta is specified in 180/128 degrees units in counterclockwise order and cannot be less than zero or greater than 256 minus one. The value of -1 can be specified if the first angle of the delta is unknown.

Property value

The first angle of the delta.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.5.3.2 NFDelta.Angle2 Property

Gets or sets the second angle of delta.

C#

```
public double Angle2;
```

Remarks

The angle of the delta is specified in 180/128 degrees units in counterclockwise order and cannot be less than zero or greater than 256 minus one. The value of -1 can be specified if the first angle of the delta is unknown.

Property value

The second angle of the delta.

7.2.2.1.5.3.3 NFDelta.Angle3 Property

Gets or sets the third angle of delta.

C#

```
public double Angle3;
```

Remarks

The angle of the delta is specified in 180/128 degrees units in counterclockwise order and cannot be less than zero or greater than 256 minus one. The value of -1 can be specified if the first angle of the delta is unknown.

Property value

The third angle of the delta.

7.2.2.1.5.3.4 NFDelta.NativeType Property

Defines the native type of the object.

C#

```
public static NType NativeType;
```

7.2.2.1.5.3.5 NFDelta.RawAngle1 Property

Gets or sets the first row angle of the delta.

C#

```
public int RawAngle1;
```

Property value

The first raw angle of the delta.

7.2.2.1.5.3.6 NFDelta.RawAngle2 Property

Gets or sets the second row angle of the delta.

C#

```
public int RawAngle2;
```

Property value

The second raw angle of the delta.

7.2.2.1.5.3.7 NFDelta.RawAngle3 Property

Gets or sets the third row angle of the delta.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public int RawAngle3;
```

Property value

The third raw angle fog the delta.

7.2.2.1.5.3.8 NFDelta.X Property

Gets or sets x coordinate of delta.

C#

```
public ushort X;
```

Remarks

The x coordinate of the delta is specified in pixels at resolution and "X * [NFRecord horizontal resolution] / Resolution" cannot be greater than MaxDimension or NFRecord width minus one.

Property value

The x coordinate of the delta.

7.2.2.1.5.3.9 NFDelta.Y Property

Gets or sets y coordinate of delta.

C#

```
public ushort Y;
```

Remarks

The y coordinate of the delta is specified in pixels at resolution and "Y * [NFRecord vertical resolution] / Resolution" cannot be greater than MaxDimension or NFRecord width minus one.

Property value

The x coordinate of the delta.

7.2.2.1.6 NFDoubleCore Structure

The structure contains information of double core.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NFDoubleCore : IFormattable {
}
```

Methods

	Name	Description
	NFDoubleCore [1687]	Initializes a new instance of the NFDoubleCore class.

NFDoubleCore Methods

	Name	Description
	ToString [1687]	Returns a string that represents the current NFDoubleCore object.

NFDoubleCore Properties

	Name	Description
S	NativeType [1687]	Defines the native type of the object.
	X [1687]	Gets or sets x coordinate of the double core.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	Y [1688]	Gets or sets y coordinate of the double core.

7.2.2.1.6.1 NFDoubleCore.NFDoubleCore Constructor

Initializes a new instance of the NFDoubleCore class.

C#

```
public NFDoubleCore(ushort x, ushort y);
```

Parameters

Parameters	Description
ushort x	The x coordinate of the double core.
ushort y	The y coordinate of the double core.

7.2.2.1.6.2 NFDoubleCore Methods

7.2.2.1.6.2.1 ToString Method

7.2.2.1.6.2.1.1 NFDoubleCore.ToString Method ()

Returns a string that represents the current NFDoubleCore [1686] object.

C#

```
public override string ToString();
```

Returns

String that represents NFDoubleCore [1686] object.

7.2.2.1.6.2.1.2 NFDoubleCore.ToString Method (string)

Returns a string that represents the current NFDoubleCore [1686] object.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	Reserved for future use.

Returns

String that represents NFDoubleCore [1686] object.

7.2.2.1.6.3 NFDoubleCore Properties

7.2.2.1.6.3.1 NFDoubleCore.NativeType Property

Defines the native type of the object.

C#

```
public static NType NativeType;
```

7.2.2.1.6.3.2 NFDoubleCore.X Property

Gets or sets x coordinate of the double core.

C#

```
public ushort X;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

The X coordinate of the double core.

7.2.2.1.6.3.3 NFDoubleCore.Y Property

Gets or sets y coordinate of the double core.

C#

```
public ushort Y;
```

Property value

The Y coordinate of the double core.

7.2.2.1.7 NFMinutia Structure

The structure contains information of minutia.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NFMinutia : IFormattable {
}
```

Methods

	Name	Description
	NFMinutia [1688]	Initializes a new instance of the NFMinutia class.

NFMinutia Methods

	Name	Description
	ToString [1690]	Returns a string that represents the current NFMinutia object.

NFMinutia Properties

	Name	Description
	Angle [1690]	Gets or sets the angle of the minutia.
	Curvature [1690]	Gets or sets the ridge curvature near minutia.
	G [1691]	Gets or sets the G (ridge density) near minutia.
S	NativeType [1691]	Defines the native type of the object.
	Quality [1691]	Gets or sets quality of the minutia.
	RawAngle [1691]	Gets or sets the raw angle of the minutia.
	Type [1691]	Gets or sets the type of the minutia.
	X [1692]	Gets or sets x coordinate of the minutia.
	Y [1692]	Gets or sets y coordinate of the minutia.

7.2.2.1.7.1 NFMinutia Constructor

7.2.2.1.7.1.1 NFMinutia.NFMinutia Constructor (ushort, ushort, NFMinutiaType, byte)

Initializes a new instance of the NFMinutia class.

C#

```
public NFMinutia(ushort x, ushort y, NFMinutiaType type, byte angle);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
ushort x	The x coordinate of the minutia.
ushort y	The y coordinate of the minutia.
NFMinutiaType type	One of the NFMinutiaType [1774] values.
byte angle	The angle of the minutia.

7.2.2.1.7.1.2 NFMinutia.NFMinutia Constructor (ushort, ushort, NFMinutiaType, byte, byte, byte, byte)

Initializes a new instance of the NFMinutia class.

C#

```
public NFMinutia(ushort x, ushort y, NFMinutiaType type, byte angle, byte quality, byte
curvature, byte g);
```

Parameters

Parameters	Description
ushort x	The x - coordinate of the minutia.
ushort y	The y - coordinate of the minutia.
NFMinutiaType type	One of the NFMinutiaType [1774] values.
byte angle	The angle of the minutia.
byte quality	The quality of the minutia.
byte curvature	The ridge curvature near minutia.
byte g	The G [1691] (ridge density) near minutia.

7.2.2.1.7.1.3 NFMinutia.NFMinutia Constructor (ushort, ushort, NFMinutiaType, double)

Initializes a new instance of the NFMinutia class.

C#

```
public NFMinutia(ushort x, ushort y, NFMinutiaType type, double angle);
```

Parameters

Parameters	Description
ushort x	The x - coordinate of the minutia.
ushort y	The y - coordinate of the minutia.
NFMinutiaType type	One of the NFMinutiaType [1774] values.
double angle	The angle of the minutia.

7.2.2.1.7.1.4 NFMinutia.NFMinutia Constructor (ushort, ushort, NFMinutiaType, double, byte, byte, byte)

Initializes a new instance of the NFMinutia class.

C#

```
public NFMinutia(ushort x, ushort y, NFMinutiaType type, double angle, byte quality, byte
curvature, byte g);
```

Parameters

Parameters	Description
ushort x	The x - coordinate of the minutia.
ushort y	The y - coordinate of the minutia.
NFMinutiaType type	One of the NFMinutiaType [1774] values.
double angle	The angle of the minutia.
byte quality	The quality of the minutia.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

byte curvature	The ridge curvature near minutia.
byte g	The G [1691] (ridge density) near minutia.

7.2.2.1.7.2 NFMinutia Methods

7.2.2.1.7.2.1 ToString Method

7.2.2.1.7.2.1.1 NFMinutia.ToString Method ()

Returns a string that represents the current NFMinutia [1688] object.

C#

```
public override string ToString();
```

Returns

String representation of NFMinutia [1688] object.

7.2.2.1.7.2.1.2 NFMinutia.ToString Method (string)

Returns a string that represents the current NFMinutia [1688] object.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	String format.

Returns

String representation of NFMinutia [1688] object.

7.2.2.1.7.3 NFMinutia Properties

7.2.2.1.7.3.1 NFMinutia.Angle Property

Gets or sets the angle of the minutia.

C#

```
public double Angle;
```

Remarks

The angle of the minutia is specified in 180/128 degrees units in counterclockwise order and cannot be greater than 256 minus one.

Property value

The angle of the minutia.

7.2.2.1.7.3.2 NFMinutia.Curvature Property

Gets or sets the ridge curvature near minutia.

C#

```
public byte Curvature;
```

Remarks

If curvature of the minutia is unknown it must be set to 255.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

The ridge curvature near minutia.

7.2.2.1.7.3.3 NFMinutia.G Property

Gets or sets the G (ridge density) near minutia.

C#

```
public byte G;
```

Remarks

If G of the minutia is unknown it must be set to 255.

Property value

The G (ridge density) near minutia.

7.2.2.1.7.3.4 NFMinutia.NativeType Property

Defines the native type of the object.

C#

```
public static NType NativeType;
```

7.2.2.1.7.3.5 NFMinutia.Quality Property

Gets or sets quality of the minutia.

C#

```
public byte Quality;
```

Remarks

The quality of the minutia must be in the range [0, 100].The higher it is, the better the quality of the minutia is. If quality of the minutia is unknown it must be set to zero.

Property value

The quality of the minutia.

7.2.2.1.7.3.6 NFMinutia.RawAngle Property

Gets or sets the raw angle of the minutia.

C#

```
public byte RawAngle;
```

Property value

The raw angle of the minutia.

7.2.2.1.7.3.7 NFMinutia.Type Property

Gets or sets the type of the minutia.

C#

```
public NFMinutiaType Type;
```

Property value

One of the NFMinutiaType [1774] values.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.7.3.8 NFMinutia.X Property

Gets or sets x coordinate of the minutia.

C#

```
public ushort X;
```

Remarks

The x coordinate of the minutia is specified in pixels at resolution and "X * [NFRecord horizontal resolution] / Resolution" cannot be greater than MaxDimension or NFRecord width minus one.

Property value

The X coordinate of the minutia.

7.2.2.1.7.3.9 NFMinutia.Y Property

Gets or sets y coordinate of the minutia.

C#

```
public ushort Y;
```

Remarks

The y coordinate of the minutia is specified in pixels at resolution and "Y * [NFRecord vertical resolution] / Resolution" cannot be greater than MaxDimension or NFRecord width minus one.

Property value

The Y coordinate of the minutia.

7.2.2.1.8 NFMinutiaNeighbor Structure

The structure contains information of minutia neighbor.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NFMinutiaNeighbor : IFormattable {
  public static readonly NFMinutiaNeighbor Empty = new NFMinutiaNeighbor(-1, 255);
}
```

NFMinutiaNeighbor Fields

	Name	Description
S R	Empty [1693]	Represents a NFMinutiaNeighbor that is a null reference.

Methods

	Name	Description
	NFMinutiaNeighbor [1693]	Initializes a new instance of the NFMinutiaNeighbor class.

NFMinutiaNeighbor Methods

	Name	Description
	ToString [1693]	Returns a string that represents the current NFMinutiaNeighbor object.

NFMinutiaNeighbor Properties

	Name	Description
	Index [1693]	Gets or sets the index of minutia neighbor.
S	NativeType [1694]	Defines the native type of the object.
	RidgeCount [1694]	Gets or sets the ridge count between the minutia and minutia neighbor.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.8.1 NFMinutiaNeighbor.NFMinutiaNeighbor Constructor

Initializes a new instance of the NFMinutiaNeighbor class.

C#

```
public NFMinutiaNeighbor(int index, byte ridgeCount);
```

Parameters

Parameters	Description
int index	The index of neighbor.
byte ridgeCount	The ridge count of neighbor.

7.2.2.1.8.2 NFMinutiaNeighbor Fields

7.2.2.1.8.2.1 NFMinutiaNeighbor.Empty Field

Represents a NFMinutiaNeighbor [1692] that is a null reference.

C#

```
public static readonly NFMinutiaNeighbor Empty = new NFMinutiaNeighbor(-1, 255);
```

7.2.2.1.8.3 NFMinutiaNeighbor Methods

7.2.2.1.8.3.1 ToString Method

7.2.2.1.8.3.1.1 NFMinutiaNeighbor.ToString Method ()

Returns a string that represents the current NFMinutiaNeighbor [1692] object.

C#

```
public override string ToString();
```

Returns

String representation of NFMinutiaNeighbor [1692] object.

7.2.2.1.8.3.1.2 NFMinutiaNeighbor.ToString Method (string)

Returns a string that represents the current NFMinutiaNeighbor [1692] object.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	String format.

Returns

String representation of NFMinutiaNeighbor [1692] object.

7.2.2.1.8.4 NFMinutiaNeighbor Properties

7.2.2.1.8.4.1 NFMinutiaNeighbor.Index Property

Gets or sets the index of minutia neighbor.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public int Index;
```

Property value

The index of minutia neighbor.

7.2.2.1.8.4.2 NFMinutiaNeighbor.NativeType Property

Defines the native type of the object.

C#

```
public static NType NativeType;
```

7.2.2.1.8.4.3 NFMinutiaNeighbor.RidgeCount Property

Gets or sets the ridge count between the minutia and minutia neighbor.

C#

```
public byte RidgeCount;
```

Property value

The ridge count between the minutia and minutia neighbor.

7.2.2.1.9 NLAttributes Class

Provides methods to retrieve face template attributes.

C#

```
public sealed class NLAttributes : NBiometricAttributes;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

Methods

NLAttributes Class

	Name	Description
	NLAttributes [1697]	Initializes a new instance of NLAttributes class.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	FromObject [1554]	Creates a new NObject [1549] from the specified object.
	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
	GetSizeMany [1558]	Returns the size of objects in the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.
S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NBiometricAttributes Class

	Name	Description
	BiometricType [1658]	Gets the biometric type.
S	NativeType [1658]	Defines the native type of the object.
	Quality [1658]	Gets or sets the quality.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	Status [1658]	Gets or sets the biometric status.

NLAttributes Class

	Name	Description
S	NativeType [1697]	Defines the native type of the object.

7.2.2.1.9.1 NLAttributes.NLAttributes Constructor

Initializes a new instance of NLAttributes class.

C#

```
public NLAttributes();
```

7.2.2.1.9.2 NLAttributes Properties

7.2.2.1.9.2.1 NLAttributes.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.2.1.10 NleDetectionDetails Class

Structure containing face detection results.

C#

```
[StructLayout(LayoutKind.Sequential)]
public class NleDetectionDetails;
```

Methods

	Name	Description
	NleDetectionDetails [1698]	Initializes a new instance of NleDetectionDetails class.

NleDetectionDetails Methods

	Name	Description
	Clear [1698]	Clears NleDetectionDetails object.

NleDetectionDetails Properties

	Name	Description
	BlinkConfidence [1698]	Gets or sets the blink confidence. Value can be [0...100].
	DarkGlassesConfidence [1698]	Gets or sets the dark glasses confidence. Value can be [0...100].
	Expression [1698]	Gets or sets NLExpression [1780] value.
	ExpressionConfidence [1698]	Gets or sets the expression confidence. Value can be [0..100].
	Face [1698]	Gets or sets information of the found face.
	FaceAvailable [1698]	Gets or sets a boolean value determining if face was found.
	Gender [1699]	Gets or sets the gender information.
	GenderConfidence [1699]	Gets or sets the gender confidence. Value can be [0..100].
	GlassesConfidence [1699]	Gets or sets the glasses confidence. Value can be [0..100].
	LeftEyeCenter [1699]	Gets the center of the left eye.
	MouthCenter [1699]	Gets the center of the mouth.
	MouthOpenConfidence [1699]	Gets or sets the mouth open confidence. Value can be [0...100].
	NoseTip [1699]	Gets the tip of the nose.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	Points [1700]	Gets or sets the feature points.
	Properties [1700]	Gets or sets NLProperties [1780] value.
	RightEyeCenter [1700]	Gets the center of the right eye.

7.2.2.1.10.1 NleDetectionDetails.NleDetectionDetails Constructor

Initializes a new instance of NleDetectionDetails class.

C#

```
public NleDetectionDetails();
```

7.2.2.1.10.2 NleDetectionDetails Methods

7.2.2.1.10.2.1 NleDetectionDetails.Clear Method

Clears NleDetectionDetails [1697] object.

C#

```
public void Clear();
```

7.2.2.1.10.3 NleDetectionDetails Properties

7.2.2.1.10.3.1 NleDetectionDetails.BlinkConfidence Property

Gets or sets the blink confidence. Value can be [0...100].

C#

```
public byte BlinkConfidence;
```

7.2.2.1.10.3.2 NleDetectionDetails.DarkGlassesConfidence Property

Gets or sets the dark glasses confidence. Value can be [0...100].

C#

```
public byte DarkGlassesConfidence;
```

7.2.2.1.10.3.3 NleDetectionDetails.Expression Property

Gets or sets NLExpression [1780] value.

C#

```
public NLExpression Expression;
```

7.2.2.1.10.3.4 NleDetectionDetails.ExpressionConfidence Property

Gets or sets the expression confidence. Value can be [0..100].

C#

```
public byte ExpressionConfidence;
```

7.2.2.1.10.3.5 NleDetectionDetails.Face Property

Gets or sets information of the found face.

C#

```
public NleFace Face;
```

7.2.2.1.10.3.6 NleDetectionDetails.FaceAvailable Property

Gets or sets a boolean value determining if face was found.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public bool FaceAvailable;
```

7.2.2.1.10.3.7 NleDetectionDetails.Gender Property

Gets or sets the gender information.

C#

```
public NGender Gender;
```

7.2.2.1.10.3.8 NleDetectionDetails.GenderConfidence Property

Gets or sets the gender confidence. Value can be [0..100].

C#

```
public byte GenderConfidence;
```

7.2.2.1.10.3.9 NleDetectionDetails.GlassesConfidence Property

Gets or sets the glasses confidence. Value can be [0..100].

C#

```
public byte GlassesConfidence;
```

7.2.2.1.10.3.10 NleDetectionDetails.LeftEyeCenter Property

Gets the center of the left eye.

C#

```
public NLFeaturePoint LeftEyeCenter;
```

Property value

The center of the left eye.

7.2.2.1.10.3.11 NleDetectionDetails.MouthCenter Property

Gets the center of the mouth.

C#

```
public NLFeaturePoint MouthCenter;
```

Property value

The center of the mouth.

7.2.2.1.10.3.12 NleDetectionDetails.MouthOpenConfidence Property

Gets or sets the mouth open confidence. Value can be [0...100].

C#

```
public byte MouthOpenConfidence;
```

7.2.2.1.10.3.13 NleDetectionDetails.NoseTip Property

Gets the tip of the nose.

C#

```
public NLFeaturePoint NoseTip;
```

Property value

The tip of the nose.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.10.3.14 NleDetectionDetails.Points Property

Gets or sets the feature points.

C#

```
public NLFeaturePoint Points;
```

7.2.2.1.10.3.15 NleDetectionDetails.Properties Property

Gets or sets NLProperties [1780] value.

C#

```
public NLProperties Properties;
```

7.2.2.1.10.3.16 NleDetectionDetails.RightEyeCenter Property

Gets the center of the right eye.

C#

```
public NLFeaturePoint RightEyeCenter;
```

Property value

The center of the right eye.

7.2.2.1.11 NleEyes Structure

Structure defining information of an eye pair.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NleEyes {
}
```

NleEyes Properties

	Name	Description
	First [1700]	Gets or sets first eye coordinates.
	FirstConfidence [1700]	Gets or sets first eye coordinate confidence
	Second [1701]	Gets or sets second eye coordinates.
	SecondConfidence [1701]	Gets or sets a double value which represents second eye coordinates confidence.

7.2.2.1.11.1 NleEyes Properties

7.2.2.1.11.1.1 NleEyes.First Property

Gets or sets first eye coordinates.

C#

```
public Point First;
```

Property value

First eye coordinates.

7.2.2.1.11.1.2 NleEyes.FirstConfidence Property

Gets or sets first eye coordinate confidence

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public double FirstConfidence;
```

Property value

Double value which represents first eye coordinates confidence.

7.2.2.1.11.1.3 NleEyes.Second Property

Gets or sets second eye coordinates.

C#

```
public Point Second;
```

Property value

Second eye coordinates.

7.2.2.1.11.1.4 NleEyes.SecondConfidence Property

Gets or sets a double value which represents second eye coordinates confidence.

C#

```
public double SecondConfidence;
```

Property value

Double value which represents second eye coordinates confidence.

7.2.2.1.12 NleFace Structure

Structure containing information of a face location in an image.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NleFace {
}
```

NleFace Properties

	Name	Description
	Confidence [1701]	Determines how confident the face region was found.
	Rectangle [1701]	Gets or sets information of the rectangle covering the face region.
	Rotation [1702]	Gets or sets information on face rotation in the image.

7.2.2.1.12.1 NleFace Properties

7.2.2.1.12.1.1 NleFace.Confidence Property

Determines how confident the face region was found.

C#

```
public double Confidence;
```

7.2.2.1.12.1.2 NleFace.Rectangle Property

Gets or sets information of the rectangle covering the face region.

C#

```
public Rectangle Rectangle;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

A set of four integers that represent the location and size of a rectangle.

7.2.2.1.12.1.3 NleFace.Rotation Property

Gets or sets information on face rotation in the image.

C#

```
public NleRotation Rotation;
```

Property value

A set of three shorts that represent rotation of the face in the image in degrees.

7.2.2.1.13 NleRotation Structure

Structure containing information of a face rotation in an image as a deviation from frontal face in degrees.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NleRotation {
}
```

NleRotation Properties

	Name	Description
	Pitch [1702]	Face pitch angle in degrees.
	Roll [1702]	Face roll angle in degrees.
	Yaw [1702]	Face yaw angle in degrees.

7.2.2.1.13.1 NleRotation Properties

7.2.2.1.13.1.1 NleRotation.Pitch Property

Face pitch angle in degrees.

C#

```
public short Pitch;
```

7.2.2.1.13.1.2 NleRotation.Roll Property

Face roll angle in degrees.

C#

```
public short Roll;
```

7.2.2.1.13.1.3 NleRotation.Yaw Property

Face yaw angle in degrees.

C#

```
public short Yaw;
```

7.2.2.1.14 NLExtractor Class

Provides functionality for extracting Neurotechnology Face Records (NLRecord [1724]) from face images.

C#

```
public sealed class NLExtractor : NObject;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NLExtractor Class

	Name	Description
	ParameterBlinkConfidenceThreshold [1708]	Identifier specifying the blink detection confidence threshold. If the confidence does not pass this threshold it will we reset to value 255 indicating that decision taking failed.
	ParameterDarkGlassesConfidenceThreshold [1708]	Identifier specifying the dark glasses detection confidence threshold. If the confidence does not pass this threshold it will we reset to value 255 indicating that decision taking failed.
	ParameterDetectAllFeaturePoints [1709]	Identifier specifying whether all facial feature points should be detected during facial feature detection. The default value is false.
	ParameterDetectBaseFeaturePoints [1709]	Identifier specifying whether base facial feature points should be detected during facial feature detection. Currently, base facial feature points are: center of right eye, center of left eye, nose tip and center of mouth. The default value is false.
	ParameterDetectBlink [1709]	Identifier specifying whether closed eyes should be detected.
	ParameterDetectDarkGlasses [1709]	Identifier specifying whether dark glasses should be detected.
	ParameterDetectExpression [1709]	Identifier specifying whether expression should be estimated.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	ParameterDetectGender [1709]	Identifier specifying whether gender should be estimated. Identifier indicating whether to use gender detection procedure.
	ParameterDetectGlasses [1709]	Identifier specifying whether glasses should be detected.
	ParameterDetectMoreFaces [1710]	Identifier specifying whether to use different face detection procedure, which has a higher recall rate.
	ParameterDetectMouthOpen [1710]	Identifier specifying whether open mouth should be detected.
	ParameterExpressionConfidenceThreshold [1710]	Identifier specifying the expression detection confidence threshold. If the confidence does not pass this threshold it will we reset to value 255 indicating that decision taking failed.
	ParameterFaceConfidenceThreshold [1710]	Specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection method.
	ParameterFaceDetectionScaleCount [1710]	Identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate. Values are in range [1..16].
	ParameterFaceQualityThreshold [1710]	Identifier specifying the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.
	ParameterFaceRollAngleBase [1710]	Parameter specifying a base angle for faces roll angle in the image to search. Available values are these: -180; -90; 0; 90; 180. The default value is 0. Parameter is used to set extractor base angle to support mobile devices with orientation sensor.
	ParameterFavorLargestFace [1710]	If set to true the largest face found in an image will be used for facial feature extraction instead of the one with the highest face confidence score. The default value is true.
	ParameterGenderConfidenceThreshold [1711]	Identifier specifying the gender detection confidence threshold. If the confidence does not pass this threshold it will we reset to value 255 indicating that decision taking failed. This threshold checks the gender of the captured face. The higher the value of this parameter, the more strictly the gender of the face is checked. The value of this parameter can be in range [0..100].
	ParameterGlassesConfidenceThreshold [1711]	Identifier specifying the glasses detection confidence threshold. If the confidence does not pass this threshold it will we reset to value 255 indicating that decision taking failed.
	ParameterLivenessThreshold [1711]	This threshold checks if faces are extracted from a live image stream. The higher the value of this parameter, the more strictly the stream is checked if the face in it is real (for example if it's not forged by showing a photo of a person in front a camera). The value of this parameter can be in range [0..100]. The default value is 50.
	ParameterMaxIod [1711]	Identifier specifying maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection methods.
	ParameterMaxRecordsPerTemplate [1711]	Sets the maximum number of records an extraction method can return in one NLTemplate [1730]. The value of this parameter can be in range [1..16]. The default value is 4.
	ParameterMaxRollAngleDeviation [1711]	Identifier specifying the maximum roll angle deviation from frontal face in degrees. The value of this parameter can be in range [0..180]. The default value is 15.
	ParameterMaxStreamDurationInFrames [1711]	Identifier specifying maximum number of frames to use for extraction from stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	ParameterMaxYawAngleDeviation [1712]	Identifier specifying the maximum yaw angle deviation from frontal face in degrees. The value of this parameter can be in range [0..180]. The default value is 15.
	ParameterMinIod [1712]	Identifier specifying minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection methods.
	ParameterMouthOpenConfidenceThreshold [1712]	Identifier specifying the open mouth detection confidence threshold. If the confidence does not pass this threshold it will we reset to value 255 indicating that decision taking failed.
	ParameterTemplateSize [1712]	Identifier specifying template size parameter. Parameter value can be one of the TemplateSize [1721] enumeration members.
	ParameterUseLivenessCheck [1712]	If set to true liveness check is performed while doing facial feature extraction from an image stream.

Methods

NLExtractor Class

	Name	Description
	NLExtractor [1708]	Initializes a new instance of NLExtractor class.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NLExtractor Class

	Name	Description
	Compress [1712]	Compresses the extracted template to a format suitable for storing the template.
	DetectFace [1712]	Method to detect a face in an image, that best fits the requirements for the face enrollment requirements.
	DetectFaces [1713]	Method to detect all faces in a given image.
	DetectFacialFeatures [1713]	Finds locations of facial feature points.
	Extract [1714]	Describes the extract method of the NLExtractor, which is used to extract face feature templates form the images.
	ExtractEnd [1714]	Ends extracting face features.
	ExtractNext [1714]	Describes the extract method of the NLExtractor, which is used to extract face feature templates from the images.
	ExtractStart [1714]	Starts extraction from an image.
	ExtractUsingDetails [1715]	Extracts feature template from the face with given detection details from an image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	Generalize [1715]	Generalizes extracted templates.

NDisposable Properties

	Name	Description
	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NLExtractor Class

	Name	Description
	BlinkConfidenceThreshold [1715]	Gets or sets a threshold for blink detection confidence.
	DarkGlassesConfidenceThreshold [1715]	Gets or sets a threshold for dark glasses detection confidence.
	DetectAllFeaturePoints [1716]	Gets or sets a value indicating whether all facial feature points should be detected.
	DetectBaseFeaturePoints [1716]	Gets or sets a value indicating whether base facial feature points should be detected.
	DetectBlink [1716]	Gets or sets a value indicating whether to use blink detection procedure. Detected blink indicates closed eyes.
	DetectDarkGlasses [1716]	Gets or sets a value indicating whether to use dark glasses detection procedure.
	DetectExpression [1716]	Gets or sets a value indicating whether to use expression detection procedure. Currently, only smiling is detected.
	DetectGender [1716]	Gets or sets a value indicating whether to use gender detection procedure.
	DetectGlasses [1717]	Gets or sets a value indicating whether to use glasses detection procedure.
	DetectMoreFaces [1717]	Gets or sets a value indicating whether face detection with higher recall should be used.
	DetectMouthOpen [1717]	Gets or sets a value indicating whether to use open mouth detection procedure.
	ExpressionConfidenceThreshold [1717]	Gets or sets a threshold for expression estimation confidence.
	FaceConfidenceThreshold [1717]	Gets or sets the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection method. Must be in range [0..100].
	FaceDetectionScaleCount [1718]	Gets or sets face detection scale count (identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate).
	FaceQualityThreshold [1718]	Gets or sets the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features.
	FaceRollAngleBase [1718]	Gets or sets faces base roll angle in the image to search. That's a base angle which together with MaxRollAngleDeviation [1719] defines the interval of angles considered when looking for faces in an image. Must be in range [-180..180].
	FavorLargestFace [1718]	Gets or sets a value indicating if the largest face will be used for facial features extraction.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	GenderConfidenceThreshold [1718]	Gets or sets a threshold for gender detection confidence.
	GlassesConfidenceThreshold [1718]	Gets or sets a threshold for glasses detection confidence.
	LivenessThreshold [1719]	Gets or sets liveness threshold which checks if faces are extracted from a live image stream. The higher the value of this parameter, the more strictly the stream is checked if the face in it is real (for example if it's not forged by showing a photo of a person in front a camera).
	MaxIod [1719]	Gets or sets maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..16384].
	MaxRecordsPerTemplate [1719]	Gets or sets the maximum number of records an extraction method can return in one NLTemlate.
	MaxRollAngleDeviation [1719]	Gets or sets maximum roll angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180].
	MaxStreamDurationInFrames [1720]	Gets or sets identifier specifying maximum number of frames to use for extraction from stream.
	MaxYawAngleDeviation [1720]	Gets or sets maximum yaw angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180].
	MinIod [1720]	Sets or gets minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection methods. Must be in range [10..16384].
	MouthOpenConfidenceThreshold [1720]	Gets or sets a threshold for open mouth detection confidence.
S	NativeType [1720]	Defines the native type of the object.
S	NleExtractionStatusNativeType [1720]	Defines the native type of the face extractor extraction status.
S	NleTemplateSizeNativeType [1721]	Defines the native type of the face extractor template size.
	TemplateSize [1721]	Gets or sets template size.
	UseLivenessCheck [1721]	Sets or gets a boolean value indicating whether to use liveness check.

7.2.2.1.14.1 NLExtractor.NLExtractor Constructor

Initializes a new instance of NLExtractor class.

C#

```
public NLExtractor();
```

7.2.2.1.14.2 NLExtractor Fields

7.2.2.1.14.2.1 NLExtractor.ParameterBlinkConfidenceThreshold Field

Identifier specifying the blink detection confidence threshold. If the confidence does not pass this threshold it will we reset to value 255 indicating that decision taking failed.

C#

```
public const ushort ParameterBlinkConfidenceThreshold = 10460;
```

7.2.2.1.14.2.2 NLExtractor.ParameterDarkGlassesConfidenceThreshold Field

Identifier specifying the dark glasses detection confidence threshold. If the confidence does not pass this threshold it will we reset to value 255 indicating that decision taking failed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public const ushort ParameterDarkGlassesConfidenceThreshold = 10456;
```

7.2.2.1.14.2.3 NLExtractor.ParameterDetectAllFeaturePoints Field

Identifier specifying whether all facial feature points should be detected during facial feature detection. The default value is false.

C#

```
public const ushort ParameterDetectAllFeaturePoints = 10250;
```

7.2.2.1.14.2.4 NLExtractor.ParameterDetectBaseFeaturePoints Field

Identifier specifying whether base facial feature points should be detected during facial feature detection. Currently, base facial feature points are: center of right eye, center of left eye, nose tip and center of mouth. The default value is false.

C#

```
public const ushort ParameterDetectBaseFeaturePoints = 10255;
```

7.2.2.1.14.2.5 NLExtractor.ParameterDetectBlink Field

Identifier specifying whether closed eyes should be detected.

C#

```
public const ushort ParameterDetectBlink = 10459;
```

7.2.2.1.14.2.6 NLExtractor.ParameterDetectDarkGlasses Field

Identifier specifying whether dark glasses should be detected.

C#

```
public const ushort ParameterDetectDarkGlasses = 10455;
```

7.2.2.1.14.2.7 NLExtractor.ParameterDetectExpression Field

Identifier specifying whether expression should be estimated.

C#

```
public const ushort ParameterDetectExpression = 10457;
```

7.2.2.1.14.2.8 NLExtractor.ParameterDetectGender Field

Identifier specifying whether gender should be estimated.

Identifier indicating whether to use gender detection procedure.

C#

```
public const ushort ParameterDetectGender = 10451;
```

Remarks

By enabling this parameter (setting to true) you can ask to estimate person's gender from the face during facial features template extraction. Gender estimator works more accurate on near-frontal face images.

7.2.2.1.14.2.9 NLExtractor.ParameterDetectGlasses Field

Identifier specifying whether glasses should be detected.

C#

```
public const ushort ParameterDetectGlasses = 10453;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.14.2.10 NLExtractor.ParameterDetectMoreFaces Field

Identifier specifying whether to use different face detection procedure, which has a higher recall rate.

C#

```
public const ushort ParameterDetectMoreFaces = 10110;
```

7.2.2.1.14.2.11 NLExtractor.ParameterDetectMouthOpen Field

Identifier specifying whether open mouth should be detected.

C#

```
public const ushort ParameterDetectMouthOpen = 10461;
```

7.2.2.1.14.2.12 NLExtractor.ParameterExpressionConfidenceThreshold Field

Identifier specifying the expression detection confidence threshold. If the confidence does not pass this threshold it will we reset to value 255 indicating that decision taking failed.

C#

```
public const ushort ParameterExpressionConfidenceThreshold = 10458;
```

7.2.2.1.14.2.13 NLExtractor.ParameterFaceConfidenceThreshold Field

Specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection method.

C#

```
public const ushort ParameterFaceConfidenceThreshold = 10103;
```

7.2.2.1.14.2.14 NLExtractor.ParameterFaceDetectionScaleCount Field

Identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate. Values are in range [1..16].

C#

```
public const ushort ParameterFaceDetectionScaleCount = 10111;
```

7.2.2.1.14.2.15 NLExtractor.ParameterFaceQualityThreshold Field

Identifier specifying the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.

C#

```
public const ushort ParameterFaceQualityThreshold = 10350;
```

7.2.2.1.14.2.16 NLExtractor.ParameterFaceRollAngleBase Field

Parameter specifying a base angle for faces roll angle in the image to search. Available values are these: -180; -90; 0; 90; 180. The default value is 0. Parameter is used to set extractor base angle to support mobile devices with orientation sensor.

C#

```
public const ushort ParameterFaceRollAngleBase = 10117;
```

7.2.2.1.14.2.17 NLExtractor.ParameterFavorLargestFace Field

If set to true the largest face found in an image will be used for facial feature extraction instead of the one with the highest face confidence score. The default value is true.

C#

```
public const ushort ParameterFavorLargestFace = 10104;
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.14.2.18 NLExtractor.ParameterGenderConfidenceThreshold Field

Identifier specifying the gender detection confidence threshold. If the confidence does not pass this threshold it will we reset to value 255 indicating that decision taking failed.

This threshold checks the gender of the captured face. The higher the value of this parameter, the more strictly the gender of the face is checked. The value of this parameter can be in range [0..100].

C#

```
public const ushort ParameterGenderConfidenceThreshold = 10452;
```

7.2.2.1.14.2.19 NLExtractor.ParameterGlassesConfidenceThreshold Field

Identifier specifying the glasses detection confidence threshold. If the confidence does not pass this threshold it will we reset to value 255 indicating that decision taking failed.

C#

```
public const ushort ParameterGlassesConfidenceThreshold = 10454;
```

7.2.2.1.14.2.20 NLExtractor.ParameterLivenessThreshold Field

This threshold checks if faces are extracted from a live image stream. The higher the value of this parameter, the more strictly the stream is checked if the face in it is real (for example if it's not forged by showing a photo of a person in front a camera). The value of this parameter can be in range [0..100]. The default value is 50.

C#

```
public const ushort ParameterLivenessThreshold = 10403;
```

7.2.2.1.14.2.21 NLExtractor.ParameterMaxIod Field

Identifier specifying maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection methods.

C#

```
public const ushort ParameterMaxIod = 10102;
```

7.2.2.1.14.2.22 NLExtractor.ParameterMaxRecordsPerTemplate Field

Sets the maximum number of records an extraction method can return in one NLTemplate [1730]. The value of this parameter can be in range [1..16]. The default value is 4.

C#

```
public const ushort ParameterMaxRecordsPerTemplate = 10408;
```

7.2.2.1.14.2.23 NLExtractor.ParameterMaxRollAngleDeviation Field

Identifier specifying the maximum roll angle deviation from frontal face in degrees. The value of this parameter can be in range [0..180]. The default value is 15.

C#

```
public const ushort ParameterMaxRollAngleDeviation = 10105;
```

7.2.2.1.14.2.24 NLExtractor.ParameterMaxStreamDurationInFrames Field

Identifier specifying maximum number of frames to use for extraction from stream.

C#

```
public const ushort ParameterMaxStreamDurationInFrames = 10409;
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.14.2.25 NLExtractor.ParameterMaxYawAngleDeviation Field

Identifier specifying the maximum yaw angle deviation from frontal face in degrees. The value of this parameter can be in range [0..180]. The default value is 15.

C#

```
public const ushort ParameterMaxYawAngleDeviation = 10109;
```

7.2.2.1.14.2.26 NLExtractor.ParameterMinIod Field

Identifier specifying minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection methods.

C#

```
public const ushort ParameterMinIod = 10101;
```

7.2.2.1.14.2.27 NLExtractor.ParameterMouthOpenConfidenceThreshold Field

Identifier specifying the open mouth detection confidence threshold. If the confidence does not pass this threshold it will we reset to value 255 indicating that decision taking failed.

C#

```
public const ushort ParameterMouthOpenConfidenceThreshold = 10462;
```

7.2.2.1.14.2.28 NLExtractor.ParameterTemplateSize Field

Identifier specifying template size parameter. Parameter value can be one of the TemplateSize [1721] enumeration members.

C#

```
public const ushort ParameterTemplateSize = 10360;
```

7.2.2.1.14.2.29 NLExtractor.ParameterUseLivenessCheck Field

If set to true liveness check is performed while doing facial feature extraction from an image stream.

C#

```
public const ushort ParameterUseLivenessCheck = 10402;
```

7.2.2.1.14.3 NLExtractor Methods

7.2.2.1.14.3.1 NLExtractor.Compress Method

Compresses the extracted template to a format suitable for storing the template.

C#

```
public static NLTemplate Compress(NLTemplate template, NleTemplateSize
compressedTemplateSize);
```

Parameters

Parameters	Description
NLTemplate template	Face feature template to be compressed.
NleTemplateSize compressedTemplateSize	Compressed template size.

Returns

Compressed face feature template.

7.2.2.1.14.3.2 NLExtractor.DetectFace Method

Method to detect a face in an image, that best fits the requirements for the face enrollment requirements.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public bool DetectFace(NGrayscaleImage image, out NleFace face);
```

Parameters

Parameters	Description
NGrayscaleImage image	Source image, in which to search for a face.
out NleFace face	NleFace [1701] structure with information about the found face.

Returns

A boolean value indicating whether a face was found in the image.

Return Values

Return Values	Description
True	Returns true if face was found in the image.
False	Returns false if face was found in the image.

7.2.2.1.14.3.3 NLExtractor.DetectFaces Method

Method to detect all faces in a given image.

C#

```
public NleFace[] DetectFaces(NGrayscaleImage image);
```

Parameters

Parameters	Description
NGrayscaleImage image	Source image, in which to search for the faces.

Returns

Returns an array of NleFace [1701] structures with information of faces found in the image. If no faces were found returns NULL [1075].

Return Values

Return Values	Description
Array	Returns an array of NleFace [1701] structures with information of faces found in the image.
NULL [1075]	If no faces were found returns NULL [1075].

7.2.2.1.14.3.4 NLExtractor.DetectFacialFeatures Method

Finds locations of facial feature points.

C#

```
public NleDetectionDetails DetectFacialFeatures(NGrayscaleImage image, NleFace face);
```

Parameters

Parameters	Description
NGrayscaleImage image	Source image, in which to search for the faces.
NleFace face	NleFace [1701] structure with information about the face in the image.

Returns

Face detection routine results.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.14.3.5 NLExtractor.Extract Method

Describes the extract method of the NLExtractor [1702], which is used to extract face feature templates form the images.

C#

```
public NLTemplate Extract(NGrayscaleImage image, out NleDetectionDetails detectionDetails,
out NleExtractionStatus extractionStatus);
```

Parameters

Parameters	Description
NGrayscaleImage image	Source image, from which the face features should be extracted.
out NleDetectionDetails detectionDetails	Returns face detection routine results.
out NleExtractionStatus extractionStatus	One of the value.

Returns

NLTemplate [1730] object.

7.2.2.1.14.3.6 NLExtractor.ExtractEnd Method

Ends extracting face features.

C#

```
public NLTemplate ExtractEnd(out int baseFrameIndex, out NleDetectionDetails
detectionDetails, out NleExtractionStatus extractionStatus);
```

Parameters

Parameters	Description
out int baseFrameIndex	Index of a frame which was used to create the template.
out NleDetectionDetails detectionDetails	Face detection result.
out NleExtractionStatus extractionStatus	Extraction status value.

Returns

NLTemplate [1730] object.

7.2.2.1.14.3.7 NLExtractor.ExtractNext Method

Describes the extract method of the NLExtractor [1702], which is used to extract face feature templates from the images.

C#

```
public NleExtractionStatus ExtractNext(NGrayscaleImage image, out NleDetectionDetails
detectionDetails);
```

Parameters

Parameters	Description
NGrayscaleImage image	Source image, from which the face features should be extracted.
out NleDetectionDetails detectionDetails	Face detection result.

Returns

NleExtractionStatus [1779] object.

7.2.2.1.14.3.8 NLExtractor.ExtractStart Method

Starts extraction from an image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public void ExtractStart();
```

7.2.2.1.14.3.9 NLExtractor.ExtractUsingDetails Method

Extracts feature template from the face with given detection details from an image.

C#

```
public NLTemplate ExtractUsingDetails(NGrayscaleImage image, NleDetectionDetails
detectionDetails, out NleExtractionStatus extractionStatus);
```

Parameters

Parameters	Description
NGrayscaleImage image	Source image.
NleDetectionDetails detectionDetails	Face detection results.
out NleExtractionStatus extractionStatus	One of the NleExtractionStatus [1779] value.

Returns

NLTemplate [1730] object.

7.2.2.1.14.3.10 NLExtractor.Generalize Method

Generalizes extracted templates.

C#

```
public NLTemplate Generalize(NLTemplate[] templates, out NleExtractionStatus
extractionStatus, out int baseTemplateIndex);
```

Parameters

Parameters	Description
NLTemplate[] templates	Array of face templates to be generalized.
out NleExtractionStatus extractionStatus	Generalization status.
out int baseTemplateIndex	Index to one of templates array member which is the main generalization template. If -1 is returned templates were not generalized.

Returns

Byte array that contains generalized NLTemplate [1730].

7.2.2.1.14.4 NLExtractor Properties

7.2.2.1.14.4.1 NLExtractor.BlinkConfidenceThreshold Property

Gets or sets a threshold for blink detection confidence.

C#

```
public byte BlinkConfidenceThreshold;
```

Property value

Blink detection confidence threshold.

7.2.2.1.14.4.2 NLExtractor.DarkGlassesConfidenceThreshold Property

Gets or sets a threshold for dark glasses detection confidence.

C#

```
public byte DarkGlassesConfidenceThreshold;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

Dark glasses detection confidence threshold.

7.2.2.1.14.4.3 NLExtractor.DetectAllFeaturePoints Property

Gets or sets a value indicating whether all facial feature points should be detected.

C#

```
public bool DetectAllFeaturePoints;
```

Property value

Boolean value indicating whether all facial feature points should be detected. If the value is false, only centers of eyes or base facial features are detected, otherwise all supported facial feature points are detected.

7.2.2.1.14.4.4 NLExtractor.DetectBaseFeaturePoints Property

Gets or sets a value indicating whether base facial feature points should be detected.

C#

```
public bool DetectBaseFeaturePoints;
```

Property value

Boolean value indicating whether all facial feature points should be detected. If the value is true, center of right eye, center of left eye, nose tip and center of mouth are detected.

7.2.2.1.14.4.5 NLExtractor.DetectBlink Property

Gets or sets a value indicating whether to use blink detection procedure. Detected blink indicates closed eyes.

C#

```
public bool DetectBlink;
```

Property value

Bool value indicating whether to use blink detection procedure.

7.2.2.1.14.4.6 NLExtractor.DetectDarkGlasses Property

Gets or sets a value indicating whether to use dark glasses detection procedure.

C#

```
public bool DetectDarkGlasses;
```

Property value

Bool value indicating whether to use dark glasses detection procedure.

7.2.2.1.14.4.7 NLExtractor.DetectExpression Property

Gets or sets a value indicating whether to use expression detection procedure. Currently, only smiling is detected.

C#

```
public bool DetectExpression;
```

Property value

Bool value indicating whether to use expression detection procedure.

7.2.2.1.14.4.8 NLExtractor.DetectGender Property

Gets or sets a value indicating whether to use gender detection procedure.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public bool DetectGender;
```

Property value

Bool value indicating whether to use gender detection procedure.

7.2.2.1.14.4.9 NLExtractor.DetectGlasses Property

Gets or sets a value indicating whether to use glasses detection procedure.

C#

```
public bool DetectGlasses;
```

Property value

Bool value indicating whether to use glasses detection procedure.

7.2.2.1.14.4.10 NLExtractor.DetectMoreFaces Property

Gets or sets a value indicating whether face detection with higher recall should be used.

C#

```
public bool DetectMoreFaces;
```

Property value

Boolean value indicating whether to use face detection procedure, which has a higher recall rate (but also a lower precision rate - more false detections).

7.2.2.1.14.4.11 NLExtractor.DetectMouthOpen Property

Gets or sets a value indicating whether to use open mouth detection procedure.

C#

```
public bool DetectMouthOpen;
```

Property value

Bool value indicating whether to use open mouth detection procedure.

7.2.2.1.14.4.12 NLExtractor.ExpressionConfidenceThreshold Property

Gets or sets a threshold for expression estimation confidence.

C#

```
public byte ExpressionConfidenceThreshold;
```

Property value

Expression estimation confidence threshold.

7.2.2.1.14.4.13 NLExtractor.FaceConfidenceThreshold Property

Gets or sets the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection method. Must be in range [0..100].

C#

```
public double FaceConfidenceThreshold;
```

Property value

Double value which represents face confidence threshold value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.14.4.14 NLExtractor.FaceDetectionScaleCount Property

Gets or sets face detection scale count (identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate).

C#

```
public int FaceDetectionScaleCount;
```

Property value

Int value which represents face detection scale count (values are in range [1..16].

7.2.2.1.14.4.15 NLExtractor.FaceQualityThreshold Property

Gets or sets the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features.

C#

```
public byte FaceQualityThreshold;
```

Property value

Byte value which represents face quality threshold.

7.2.2.1.14.4.16 NLExtractor.FaceRollAngleBase Property

Gets or sets faces base roll angle in the image to search. That's a base angle which together with MaxRollAngleDeviation [1719] defines the interval of angles considered when looking for faces in an image. Must be in range [-180..180].

C#

```
public short FaceRollAngleBase;
```

Property value

Short value which represents face roll angle base.

7.2.2.1.14.4.17 NLExtractor.FavorLargestFace Property

Gets or sets a value indicating if the largest face will be used for facial features extraction.

C#

```
public bool FavorLargestFace;
```

Property value

Boolean value indicating if the largest face should be used for facial features extraction. If set to true the largest face found in an image will be used for facial feature extraction instead of the one with the highest ParameterFaceConfidenceThreshold [1710] score.

7.2.2.1.14.4.18 NLExtractor.GenderConfidenceThreshold Property

Gets or sets a threshold for gender detection confidence.

C#

```
public byte GenderConfidenceThreshold;
```

Property value

Gender detection confidence threshold.

7.2.2.1.14.4.19 NLExtractor.GlassesConfidenceThreshold Property

Gets or sets a threshold for glasses detection confidence.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public byte GlassesConfidenceThreshold;
```

Property value

Glasses detection confidence threshold.

7.2.2.1.14.4.20 NLExtractor.LivenessThreshold Property

Gets or sets liveness threshold which checks if faces are extracted from a live image stream. The higher the value of this parameter, the more strictly the stream is checked if the face in it is real (for example if it's not forged by showing a photo of a person in front a camera).

C#

```
public double LivenessThreshold;
```

Notes

The value can be in range [0..100]. The default value is 50.

Property value

Liveness threshold value.

7.2.2.1.14.4.21 NLExtractor.MaxIod Property

Gets or sets maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..16384].

C#

```
public int MaxIod;
```

Notes

If MinIod [1720] and/or MaxIod will be set very similar to the real distance between the eyes, it could cause to not so accurate results (of face detection).

When the face is set MinIod [1720] and MaxIod the face could be found up to ~1.5 times bigger/smaller value it is set.

Property value

Integer value which represents maximum Iod.

7.2.2.1.14.4.22 NLExtractor.MaxRecordsPerTemplate Property

Gets or sets the maximum number of records an extraction method can return in one NLTemlate.

C#

```
public int MaxRecordsPerTemplate;
```

Property value

Integer value which represents maximum number of record per template. The value can be in range [2..16]. The default value is 5.

7.2.2.1.14.4.23 NLExtractor.MaxRollAngleDeviation Property

Gets or sets maximum roll angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180].

C#

```
public short MaxRollAngleDeviation;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

Short value which represents maximum face roll angle deviation value.

7.2.2.1.14.4.24 NLExtractor.MaxStreamDurationInFrames Property

Gets or sets identifier specifying maximum number of frames to use for extraction from stream.

C#

```
public int MaxStreamDurationInFrames;
```

Property value

Int value representing maximum stream duration in frames.

7.2.2.1.14.4.25 NLExtractor.MaxYawAngleDeviation Property

Gets or sets maximum yaw angle deviation from frontal face in degrees which is considered when looking for faces in an image. Must be in range [0..180].

C#

```
public short MaxYawAngleDeviation;
```

Property value

Short value which represents maximum face yaw angle deviation value.

7.2.2.1.14.4.26 NLExtractor.MinIod Property

Sets or gets minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection methods. Must be in range [10..16384].

C#

```
public int MinIod;
```

Notes

If MinIod and/or MaxIod [1719] will be set very similar to the real distance between the eyes, it could cause to not so accurate results (of face detection).

When the face is set MinIod and MaxIod [1719] the face could be found up to ~1.5 times bigger/smaller value it is set.

7.2.2.1.14.4.27 NLExtractor.MouthOpenConfidenceThreshold Property

Gets or sets a threshold for open mouth detection confidence.

C#

```
public byte MouthOpenConfidenceThreshold;
```

Property value

Open mouth detection confidence threshold.

7.2.2.1.14.4.28 NLExtractor.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.2.1.14.4.29 NLExtractor.NleExtractionStatusNativeType Property

Defines the native type of the face extractor extraction status.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static NType NleExtractionStatusNativeType;
```

7.2.2.1.14.4.30 NLExtractor.NleTemplateSizeNativeType Property

Defines the native type of the face extractor template size.

C#

```
public static NType NleTemplateSizeNativeType;
```

7.2.2.1.14.4.31 NLExtractor.TemplateSize Property

Gets or sets template size.

C#

```
public NleTemplateSize TemplateSize;
```

Property value

One of the NleTemplateSize [1779] value.

7.2.2.1.14.4.32 NLExtractor.UseLivenessCheck Property

Sets or gets a boolean value indicating whether to use liveness check.

C#

```
public bool UseLivenessCheck;
```

Property value

Boolean value indicating whether to use liveness check. The property value is true if liveness check is performed while doing facial feature extraction from an image stream. Otherwise, the property value is false.

7.2.2.1.15 NLFeaturePoint Structure

Structure defining facial feature point.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NLFeaturePoint : IFormattable {
}
```

Methods

	Name	Description
	NLFeaturePoint [1722]	Initializes a new instance of NLFeaturePoint structure.

NLFeaturePoint Methods

	Name	Description
	ToString [1722]	Retrieves a string representation of the object.

NLFeaturePoint Properties

	Name	Description
	Code [1723]	Gets or sets the feature code.
	Confidence [1723]	Gets or sets the feature confidence.
S	NativeType [1723]	Defines the native type of the object.
	X [1723]	Gets or sets the X-coordinate of the feature point.
	Y [1723]	Gets or sets the Y-coordinate of the feature point.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.15.1 NLFeaturePoint Constructor

7.2.2.1.15.1.1 NLFeaturePoint.NLFeaturePoint Constructor (ushort, ushort, ushort)

Initializes a new instance of NLFeaturePoint structure.

C#

```
public NLFeaturePoint(ushort code, ushort x, ushort y);
```

Parameters

Parameters	Description
ushort code	Feature code.
ushort x	X [1723]-coordinate of the the feature point.
ushort y	Y [1723]-coordinate of the the feature point.

7.2.2.1.15.1.2 NLFeaturePoint.NLFeaturePoint Constructor (ushort, ushort, ushort, byte)

Initializes a new instance of NLFeaturePoint structure.

C#

```
public NLFeaturePoint(ushort code, ushort x, ushort y, byte confidence);
```

Parameters

Parameters	Description
ushort code	Feature code.
ushort x	X [1723]-coordinate of the the feature point.
ushort y	Y [1723]-coordinate of the the feature point.
byte confidence	Feature confidence.

7.2.2.1.15.2 NLFeaturePoint Methods

7.2.2.1.15.2.1 ToString Method

7.2.2.1.15.2.1.1 NLFeaturePoint.ToString Method ()

Retrieves a string representation of the object.

C#

```
public override string ToString();
```

Returns

A string representation of an object.

7.2.2.1.15.2.1.2 NLFeaturePoint.ToString Method (string)

Retrieves a string representation of the object.

C#

```
public string ToString(string format);
```

Parameters

Parameters	Description
string format	String format

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

A string representation of an object.

7.2.2.1.15.3 NLFeaturePoint Properties

7.2.2.1.15.3.1 NLFeaturePoint.Code Property

Gets or sets the feature code.

C#

```
public ushort Code;
```

7.2.2.1.15.3.2 NLFeaturePoint.Confidence Property

Gets or sets the feature confidence.

C#

```
public byte Confidence;
```

7.2.2.1.15.3.3 NLFeaturePoint.NativeType Property

Defines the native type of the object.

C#

```
public static NType NativeType;
```

7.2.2.1.15.3.4 NLFeaturePoint.X Property

Gets or sets the X-coordinate of the feature point.

C#

```
public ushort X;
```

7.2.2.1.15.3.5 NLFeaturePoint.Y Property

Gets or sets the Y-coordinate of the feature point.

C#

```
public ushort Y;
```

7.2.2.1.16 NLMatchingDetails Class

Provides functionality to retrieve face templates matching details.

C#

```
public sealed class NLMatchingDetails : NXMatchingDetails;
```

NLMatchingDetails Properties

	Name	Description
S	NativeType [1723]	Defines the native type of the object.

7.2.2.1.16.1 NLMatchingDetails Properties

7.2.2.1.16.1.1 NLMatchingDetails.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.17 NLRecord Class

Provides functionality for packing, unpacking and editing Neurotechnology Face Records (NLRecords).

C#

```
public sealed class NLRecord : NObject;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

Methods

NLRecord Class

	Name	Description
	NLRecord [1726]	Initializes a new instance of the NLRecord class.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.
S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.
S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NLRecord Class

	Name	Description
S	Check [1728]	Checks if format of packed NLRecord is correct. Checks if format of packed NLRecord is correct.
S	GetQuality [1728]	Retrieves face quality from packed NLRecord.
S	GetSize [1729]	Gets the size of packed NLRecord.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NLRecord Class

	Name	Description
	CbeffProductType [1729]	Gets or sets the Cbeff product type of the NLRecord.
S	NativeType [1729]	Defines the native type of the object.
	Quality [1730]	Gets or sets face quality.

7.2.2.1.17.1 NLRecord Constructor

7.2.2.1.17.1.1 NLRecord.NLRecord Constructor ()

Initializes a new instance of the NLRecord class.

C#

```
public NLRecord();
```

7.2.2.1.17.1.2 NLRecord.NLRecord Constructor (NBuffer)

Initializes a new instance of the NLRecord class.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NLRecord(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	The packed NLRecord object.

7.2.2.1.17.1.3 NLRecord.NLRecord Constructor (NBuffer, uint)

Initializes a new instance of the NLRecord class.

C#

```
public NLRecord(NBuffer buffer, uint flags);
```

Parameters

Parameters	Description
NBuffer buffer	The packed NLRecord object.
uint flags	Must be zero.

7.2.2.1.17.1.4 NLRecord.NLRecord Constructor (NBuffer, uint, out int)

Initializes a new instance of the NLRecord class.

C#

```
public NLRecord(NBuffer buffer, uint flags, out int size);
```

Parameters

Parameters	Description
NBuffer buffer	The packed NLRecord object.
uint flags	Must be zero.
out int size	[out] NLRecord size.

7.2.2.1.17.1.5 NLRecord.NLRecord Constructor (byte[])

Initializes a new instance of the NLRecord class.

C#

```
public NLRecord(byte[] buffer);
```

Parameters

Parameters	Description
byte[] buffer	The packed NLRecord object.

7.2.2.1.17.1.6 NLRecord.NLRecord Constructor (uint)

Initializes a new instance of the NLRecord class.

C#

```
public NLRecord(uint flags);
```

Parameters

Parameters	Description
uint flags	Bitwise combination of zero or more flags that controls behavior of the constructor. Exclusive-OR is performed when few flags needed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Remarks

The following flags are supported:

- FlagSkipBlockedOrients
- FlagSkipGs
- FlagSkipQualities

7.2.2.1.17.2 NLRecord Methods

7.2.2.1.17.2.1 Check Method

7.2.2.1.17.2.1.1 NLRecord.Check Method (NBuffer)

Checks if format of packed NLRecord [1724] is correct.

Checks if format of packed NLRecord [1724] is correct.

C#

```
public static void Check(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	The packed NLRecord [1724] object.

7.2.2.1.17.2.1.2 NLRecord.Check Method (byte[])

Checks if format of packed NLRecord [1724] is correct.

Checks if format of packed NLRecord [1724] is correct.

C#

```
public static void Check(byte[] buffer);
```

Parameters

Parameters	Description
byte[] buffer	The packed NLRecord [1724] object.

7.2.2.1.17.2.2 GetQuality Method

7.2.2.1.17.2.2.1 NLRecord.GetQuality Method (NBuffer)

Retrieves face quality from packed NLRecord [1724].

C#

```
public static byte GetQuality(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] of packed NLRecord [1724].

Returns

The value of face quality.

7.2.2.1.17.2.2.2 NLRecord.GetQuality Method (byte[])

Retrieves face quality from packed NLRecord [1724].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static byte GetQuality(byte[] buffer);
```

Parameters

Parameters	Description
byte[] buffer	The byte array of packed NLRecord [1724].

Returns

The value of face quality.

7.2.2.1.17.2.3 GetSize Method

7.2.2.1.17.2.3.1 NLRecord.GetSize Method (NBuffer)

Gets the size of packed NLRecord [1724].

C#

```
public static int GetSize(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	The packed NLRecord [1724].

Returns

Size of NLRecord [1724].

7.2.2.1.17.2.3.2 NLRecord.GetSize Method (byte[])

Gets the size of packed NLRecord [1724].

C#

```
public static int GetSize(byte[] buffer);
```

Parameters

Parameters	Description
byte[] buffer	The packed NLRecord [1724].

Returns

Size of NLRecord [1724].

7.2.2.1.17.3 NLRecord Properties

7.2.2.1.17.3.1 NLRecord.CbeffProductType Property

Gets or sets the Cbeff product type of the NLRecord [1724].

C#

```
public ushort CbeffProductType;
```

Property value

The Cbeff product type.

7.2.2.1.17.3.2 NLRecord.NativeType Property

Defines the native type of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static new NType NativeType;
```

7.2.2.1.17.3.3 NLRecord.Quality Property

Gets or sets face quality.

C#

```
public byte Quality;
```

Property value

Face quality.

7.2.2.1.18 NLTemplate Class

Provides functionality for packing, unpacking and editing Neurotechnology Faces [45] Templates (NLTemplates).

C#

```
public sealed class NLTemplate : NObject;
```

NLTemplate Classes

NLTemplate Class

	Name	Description
	RecordCollection [1734]	Collection of NLRecords.

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NLTemplate Class

	Name	Description
	FlagProcessFirstRecordOnly [1734]	The flag indicating whether only the first face record should be unpacked or packed while unpacking or packing NLTemplate.
	MaxRecordCount [1734]	The maximum number of face records NLTemplate can contain.

Methods

NLTemplate Class

	Name	Description
	NLTemplate [1733]	Initializes a new instance of the NLTemplate class.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NLTemplate Class

	Name	Description
	CalculateSize [1734]	Calculates NLTemplate size.
	Check [1735]	Checks if format of packed NLTemplate format is correct.
	GetRecordCount [1735]	Retrieves records count.
	GetSize [1736]	Gets the size of packed NLTemplate.

NDisposable Properties

	Name	Description
	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NLTemplate Class

	Name	Description
S	NativeType [1736]	Defines the native type of the object.
	Records [1736]	Gets Records collection.

7.2.2.1.18.1 NLTemplate Constructor

7.2.2.1.18.1.1 NLTemplate.NLTemplate Constructor ()

Initializes a new instance of the NLTemplate class.

C#

```
public NLTemplate();
```

7.2.2.1.18.1.2 NLTemplate.NLTemplate Constructor (NBuffer)

Initializes a new instance of the NLTemplate class.

C#

```
public NLTemplate(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] with packed NLTemplate objects.

7.2.2.1.18.1.3 NLTemplate.NLTemplate Constructor (NBuffer, uint)

Initializes a new instance of the NLTemplate class.

C#

```
public NLTemplate(NBuffer buffer, uint flags);
```

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] with packed NLTemplate objects.
uint flags	Reserved for future use. Must be zero.

7.2.2.1.18.1.4 NLTemplate.NLTemplate Constructor (NBuffer, uint, out int)

Initializes a new instance of the NLTemplate class.

C#

```
public NLTemplate(NBuffer buffer, uint flags, out int size);
```

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] with packed NLTemplate objects.
uint flags	Reserved for future use. Must be zero.
out int size	[out] NLTemplate size.

7.2.2.1.18.1.5 NLTemplate.NLTemplate Constructor (byte[])

Initializes a new instance of the NLTemplate class.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NLTemplate(byte[] buffer);
```

Parameters

Parameters	Description
byte[] buffer	A byte array with packed NLTemplate objects.

7.2.2.1.18.1.6 NLTemplate.NLTemplate Constructor (uint)

Initializes a new instance of the NLTemplate class.

C#

```
public NLTemplate(uint flags);
```

Parameters

Parameters	Description
uint flags	Bitwise combination of zero or more flags that controls behavior of the constructor.

7.2.2.1.18.2 NLTemplate Classes

7.2.2.1.18.2.1 NLTemplate.RecordCollection Class

Collection of NLRecords.

C#

```
public sealed class RecordCollection : NObjectCollection<NLRecord>;
```

7.2.2.1.18.3 NLTemplate Fields

7.2.2.1.18.3.1 NLTemplate.FlagProcessFirstRecordOnly Field

The flag indicating whether only the first face record should be unpacked or packed while unpacking or packing NLTemplate [1730].

C#

```
public const uint FlagProcessFirstRecordOnly = 0x00000100;
```

7.2.2.1.18.3.2 NLTemplate.MaxRecordCount Field

The maximum number of face records NLTemplate [1730] can contain.

C#

```
public const int MaxRecordCount = 255;
```

7.2.2.1.18.4 NLTemplate Methods

7.2.2.1.18.4.1 NLTemplate.CalculateSize Method

Calculates NLTemplate [1730] size.

C#

```
public static int CalculateSize(int[] recordSizes);
```

Parameters

Parameters	Description
int[] recordSizes	An array that contains NLRecords sizes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

A size of records.

7.2.2.1.18.4.2 Check Method

7.2.2.1.18.4.2.1 NLTemplate.Check Method (NBuffer)

Checks if format of packed NLTemplate [1730] format is correct.

C#

```
public static void Check(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] with packed NLTemplate [1730].

7.2.2.1.18.4.2.2 NLTemplate.Check Method (byte[])

Checks if format of packed NLTemplate [1730] format is correct.

C#

```
public static void Check(byte[] buffer);
```

Parameters

Parameters	Description
byte[] buffer	A byte array with packed NLTemplate [1730].

Returns

true if NLTemplate [1730] format is correct, false otherwise.

7.2.2.1.18.4.3 GetRecordCount Method

7.2.2.1.18.4.3.1 NLTemplate.GetRecordCount Method (NBuffer)

Retrieves records count.

C#

```
public static int GetRecordCount(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] with NLTemplate [1730].

Returns

Records [1736] count.

7.2.2.1.18.4.3.2 NLTemplate.GetRecordCount Method (byte[])

Retrieves records count.

C#

```
public static int GetRecordCount(byte[] buffer);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
byte[] buffer	A byte array with NLTemplate [1730].

Returns

Records [1736] count.

7.2.2.1.18.4.4 GetSize Method

7.2.2.1.18.4.4.1 NLTemplate.GetSize Method (NBuffer)

Gets the size of packed NLTemplate [1730].

C#

```
public static int GetSize(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	The packed NLTemplate [1730].

Returns

NLTemplate [1730] size.

7.2.2.1.18.4.4.2 NLTemplate.GetSize Method (byte[])

Gets the size of packed NLTemplate [1730].

C#

```
public static int GetSize(byte[] buffer);
```

Parameters

Parameters	Description
byte[] buffer	The packed NLTemplate [1730].

Returns

NLTemplate [1730] size.

7.2.2.1.18.5 NLTemplate Properties

7.2.2.1.18.5.1 NLTemplate.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.2.1.18.5.2 NLTemplate.Records Property

Gets Records collection.

C#

```
public RecordCollection Records;
```

Property value

A RecordCollection [1734] collection.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.19 NMatcher Class

Provides functionality for comparing all Neurotechnology Templates (NTemplates, NFTemplates, NFRecords, NLTemplates, NLRecords, NETemplates, NERecords), encapsulated in Neurotechnology Matcher (NMatcher) object.

C#

```
public sealed class NMatcher : NObject;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NMatcher Class

	Name	Description
	FingersModeAtmelFingerChip [1743]	Represents Atmel Finger Chip scanner.
	FingersModeAuthentecAes2501B [1743]	Represents Authentec Aes 2501B scanner.
	FingersModeAuthentecAes4000 [1743]	Represents Authentec Aes 4000 scanner.
	FingersModeAuthentecAFS2 [1743]	Represents Authentec AFS2 scanner.
	FingersModeBiometrikaFX2000 [1743]	Represents Biometrika FX2000 scanner.
	FingersModeBiometrikaFX3000 [1743]	Represents Biometrika FX3000 scanner.
	FingersModeBmfBlp100 [1743]	Represents Bmf Blp 100 scanner.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	FingersModeCrossMatchVerifier300 [1743]	Represents Cross Match Verifier 300 scanner.
	FingersModeDigentIzzix [1743]	Represents Digent Izzix scanner.
	FingersModeDigitalPersonaUareU [1744]	Represents Digital Persona U are U scanner.
	FingersModeEthentica [1744]	Represents Ethentica scanner.
	FingersModeFujitsuMbf200 [1744]	Represents Fujitsu Mbf200 scanner.
	FingersModeFutronicFS80 [1744]	Represents Futronic FS80 scanner.
	FingersModeGeneral [1744]	Represents general scanner.
	FingersModeIdenticatorTechnologyDF90 [1744]	Represents Identicator Technology DF90 scanner.
	FingersModeIdentixDfr2090 [1744]	Represents Identix Dfr2090 scanner.
	FingersModeIdentixTouchView [1744]	Represents Identix TouchView scanner.
	FingersModeKeytronicSecureDesktop [1744]	Represents Keytronic Secure Desktop scanner.
	FingersModeLighTuningLttC500 [1745]	Represents Light Tuning LttC500 scanner.
	FingersModeNitgenFingkeyHamster [1745]	Represents Nitgen Fingkey Hamster scanner.
	FingersModePreciseBiometrics100CS [1745]	Represents Precise Biometrics [1654] 100CS scanner.
	FingersModeSecugenHamster [1745]	Represents Secugen Hamster scanner.
	FingersModeStartekFM200 [1745]	Represents Startek FM200 scanner.
	FingersModeTacomaCmos [1745]	Represents Tacoma Cmos scanner.
	FingersModeTestechBioI [1745]	Represents Testech Bio I scanner.
	FingersModeUpekTouchChip [1745]	Represents Upek Touch Chip scanner.
	ParameterFacesMatcherBase [1745]	Helper parameter for internal usage.
	ParameterFacesMatchingSpeed [1746]	Specifies face templates matching speed. Can be Low or High matching speed.
	ParameterFacesMatchingThresholdNew [1746]	Matters only if ParameterFusionType [1747] is SelectByFaceThenFuse(NMFusionType [1781]) set.
	ParameterFingersFillFullMatchingDetails [1746]	Specifies whether full matching details are filled during the fingers matching.
	ParameterFingersMatcherBase [1746]	Helper parameter for internal usage.
	ParameterFingersMatchingSpeed [1746]	Specifies finger templates matching speed. Can be Low, Medium or High matching speed.
	ParameterFingersMaximalRotation [1746]	Identifier specifying modulus of maximal rotation allowed between two matched NFRecords parameter of type N_TYPE_BYTE [576]. Parameter value is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).
	ParameterFingersMinMatchedCount [1746]	Identifier specifying minimal number of fingerprints from probe template to match parameter of type int. Parameter value is from zero to NFTemplate.MaxRecordCount.
	ParameterFingersMinMatchedCountThreshold [1746]	Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to Neurotec.Biometrics.NMatcher.ParameterFingersMinMatchedCount [1746] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
	ParameterFingersMode [1747]	Parameter specifying fingers mode (parameter value set). Parameter value can be one of the FingersModeXXX.
	ParameterFusionType [1747]	Identifier specifying fingerprints and faces fusion of type int. Parameter value can be one of the NMFusionType [1781] enumeration members.
	ParameterIrisesMatcherBase [1747]	Helper parameter for internal usage.
	ParameterIrisesMatchingSpeed [1747]	Specifies iris templates matching speed. Can be Low, Medium or High matching speed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	ParameterIrisesMatchingThreshold [1747]	Matters only if ParameterFusionType [1747] is SelectByIrisThenFuse(NMFusionType [1781]) set.
	ParameterIrisesMaximalRotation [1747]	Defines the maximal rotation allowed between two matched irises. It is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).
	ParameterIrisesMinMatchedCount [1747]	Identifier specifying minimal number of irises from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NET_MAX_RECORD_COUNT.
	ParameterIrisesMinMatchedCountThreshold [1747]	Parameter specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to Neurotec.Biometrics.NMatcher.ParameterIrisesMinMatchedCount [1747] parameter value). Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
	ParameterMatchingThreshold [1748]	Identifier specifying matching threshold (biggest allowed FAR) parameter of type int. Parameter value is equal to - 12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
	ParameterPalmsFillFullMatchingDetails [1748]	Specifies whether full matching details are filled during the palms matching.
	ParameterPalmsMatcherBase [1748]	Helper parameter for internal usage.
	ParameterPalmsMatchingSpeed [1748]	Specifies palm templates matching speed. Can be Low, Medium or High matching speed.
	ParameterPalmsMaximalRotation [1748]	Identifier specifying modulus of maximal rotation allowed between two matched NFRecords parameter of type N_TYPE_BYTE [576]. Parameter value is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).
	ParameterPalmsMinMatchedCount [1748]	Identifier specifying minimal number of palmprints from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.
	ParameterPalmsMinMatchedCountThreshold [1748]	Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to Neurotec.Biometrics.NMatcher.ParameterPalmsMinMatchedCount [1748] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
	ParameterVoicesMatcherBase [1749]	Helper parameter for internal usage.
	ParameterVoicesUniquePhrasesOnly [1749]	Identifier specifying whether each user in the system always uses only his own phrases which are unique throughout the whole database of users.
	PartFaces [1749]	Identifier specifying that retrieving or setting parameter for faces matcher (NlsMatcher [143]) contained in NMatcher. See GetParameter [1557], SetParameter [1563], GetStaticParameter
	PartFingers [1749]	Identifier specifying that retrieving or setting parameter for fingers matcher (NfsMatcher) contained in NMatcher. See GetParameter [1557], SetParameter [1563], GetStaticParameter
	PartIrises [1749]	Identifier specifying that retrieving or setting parameter for irises matcher (NesMatcher) contained in NMatcher. See GetParameter [1557], SetParameter [1563], GetStaticParameter

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	PartNone [1749]	Identifier specifying that retrieving or setting parameter for NMatcher. See GetParameter [1557], SetParameter [1563], GetStaticParameter
	PartPalms [1749]	Identifier specifying that retrieving or setting parameter for palms matcher (NfsMatcher) contained in NMatcher. See GetParameter [1557], SetParameter [1563], GetStaticParameter

Methods

NMatcher Class

	Name	Description
	NMatcher [1742]	Initializes a new instance of the NMatcher class.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.
S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NMatcher Class

	Name	Description
	IdentifyEnd [1749]	Ends identification process.
	IdentifyNext [1750]	Compares the specified NTemplate [1762] with the one identification was started with.
	IdentifyStart [1751]	Starts identification with the specified packed NTemplate [1762].
	PrepareTemplate [1752]	Changes specified template for faster matching.
	Verify [1752]	Compares two packed templates.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMatcher Class

	Name	Description
	FacesMatchingSpeed [1754]	Gets or sets faces matching speed.
	FacesMatchingThresholdNew [1754]	Matters only if ParameterFusionType [1747] is SelectByFaceThenFuse(NMFusionType [1781]) set.
	FingersMatchingSpeed [1754]	Gets or sets fingers matching speed.
	FingersMaximalRotation [1754]	Gets or sets maximal rotation.
	FingersMinMatchedCount [1754]	Gets or sets minimal number of fingerprints from probe template to match parameter. The value is from zero to MaxRecordCount (NFTemplate).
	FingersMinMatchedCountThreshold [1755]	Gets or sets matching threshold (biggest allowed FAR) when matching is stopped (with respect to FingersMinMatchedFingerCount parameter value) parameter. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
	FingersMode [1755]	Gets or sets scanners mode.
	FusionType [1755]	Gets or sets fingerprints, irises and faces fusion type.
	IrisesMatchingSpeed [1755]	Gets or sets irises matching speed.
	IrisesMatchingThreshold [1755]	Matters only if ParameterFusionType [1747] is SelectByIrisThenFuse(NMFusionType [1781]) set.
	IrisesMaximalRotation [1755]	Gets or sets the maximal rotation allowed between two matched irises. It is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).
	IrisesMinMatchedCount [1756]	Gets or sets minimal number of irises from probe template to match parameter. The value is from zero to MaxRecordCount (NFTemplate).
	IrisesMinMatchedCountThreshold [1756]	Gets or sets matching threshold (biggest allowed FAR) when matching is stopped (with respect to IrisesMinMatchedCount [1756] parameter value) parameter. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
	MatchingThreshold [1756]	Gets or sets the matching threshold.
S	NativeType [1756]	Defines the native type of the object.
S	NMatchingSpeedNativeType [1756]	Defines the native type of the matching speed.
S	NMFusionTypeNativeType [1756]	Defines the native type of the fusion type.
	PalmsMatchingSpeed [1756]	Gets or sets palms matching speed.
	PalmsMaximalRotation [1756]	Gets or sets maximal rotation.
	PalmsMinMatchedCount [1757]	Gets or sets minimal number of palmprints from probe template to match parameter. The value is from zero to MaxRecordCount (NFTemplate).
	PalmsMinMatchedCountThreshold [1757]	Gets or sets matching threshold (biggest allowed FAR) when matching is stopped (with respect to PalmsMinMatchedCount [1757] parameter value) parameter. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
	VoicesUniquePhrasesOnly [1757]	Gets or sets identifier specifying whether each user in the system always uses only his own phrases which are unique throughout the whole database of users.

7.2.2.1.19.1 NMatcher.NMatcher Constructor

Initializes a new instance of the NMatcher class.

C#

```
public NMatcher();
```

7.2.2.1.19.2 NMatcher Fields

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.19.2.1 NMatcher.FingersModeAtmelFingerChip Field

Represents Atmel Finger Chip scanner.

C#

```
public const uint FingersModeAtmelFingerChip = 900;
```

7.2.2.1.19.2.2 NMatcher.FingersModeAuthentecAes2501B Field

Represents Authentec Aes 2501B scanner.

C#

```
public const uint FingersModeAuthentecAes2501B = 811;
```

7.2.2.1.19.2.3 NMatcher.FingersModeAuthentecAes4000 Field

Represents Authentec Aes 4000 scanner.

C#

```
public const uint FingersModeAuthentecAes4000 = 810;
```

7.2.2.1.19.2.4 NMatcher.FingersModeAuthentecAFS2 Field

Represents Authentec AFS2 scanner.

C#

```
public const uint FingersModeAuthentecAFS2 = 800;
```

7.2.2.1.19.2.5 NMatcher.FingersModeBiometrikaFX2000 Field

Represents Biometrika FX2000 scanner.

C#

```
public const uint FingersModeBiometrikaFX2000 = 200;
```

7.2.2.1.19.2.6 NMatcher.FingersModeBiometrikaFX3000 Field

Represents Biometrika FX3000 scanner.

C#

```
public const uint FingersModeBiometrikaFX3000 = 201;
```

7.2.2.1.19.2.7 NMatcher.FingersModeBmfBlp100 Field

Represents Bmf Blp 100 scanner.

C#

```
public const uint FingersModeBmfBlp100 = 1000;
```

7.2.2.1.19.2.8 NMatcher.FingersModeCrossMatchVerifier300 Field

Represents Cross Match Verifier 300 scanner.

C#

```
public const uint FingersModeCrossMatchVerifier300 = 1300;
```

7.2.2.1.19.2.9 NMatcher.FingersModeDigentIzzix Field

Represents Digent Izzix scanner.

C#

```
public const uint FingersModeDigentIzzix = 1600;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.19.2.10 NMatcher.FingersModeDigitalPersonaUareU Field

Represents Digital Persona U are U scanner.

C#

```
public const uint FingersModeDigitalPersonaUareU = 100;
```

7.2.2.1.19.2.11 NMatcher.FingersModeEthentica Field

Represents Ethentica scanner.

C#

```
public const uint FingersModeEthentica = 1200;
```

7.2.2.1.19.2.12 NMatcher.FingersModeFujitsuMbf200 Field

Represents Fujitsu Mbf200 scanner.

C#

```
public const uint FingersModeFujitsuMbf200 = 1800;
```

7.2.2.1.19.2.13 NMatcher.FingersModeFutronicFS80 Field

Represents Futronic FS80 scanner.

C#

```
public const uint FingersModeFutronicFS80 = 1900;
```

7.2.2.1.19.2.14 NMatcher.FingersModeGeneral Field

Represents general scanner.

C#

```
public const uint FingersModeGeneral = 0;
```

7.2.2.1.19.2.15 NMatcher.FingersModeIdenticatorTechnologyDF90 Field

Represents Identicator Technology DF90 scanner.

C#

```
public const uint FingersModeIdenticatorTechnologyDF90 = 700;
```

7.2.2.1.19.2.16 NMatcher.FingersModeIdentixDfr2090 Field

Represents Identix Dfr2090 scanner.

C#

```
public const uint FingersModeIdentixDfr2090 = 401;
```

7.2.2.1.19.2.17 NMatcher.FingersModeIdentixTouchView Field

Represents Identix TouchView scanner.

C#

```
public const uint FingersModeIdentixTouchView = 400;
```

7.2.2.1.19.2.18 NMatcher.FingersModeKeytronicSecureDesktop Field

Represents Keytronic Secure Desktop scanner.

C#

```
public const uint FingersModeKeytronicSecureDesktop = 300;
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.19.2.19 NMatcher.FingersModeLighTuningLttC500 Field

Represents Light Tuning LttC500 scanner.

C#

```
public const uint FingersModeLighTuningLttC500 = 2000;
```

7.2.2.1.19.2.20 NMatcher.FingersModeNitgenFingkeyHamster Field

Represents Nitgen Fingkey Hamster scanner.

C#

```
public const uint FingersModeNitgenFingkeyHamster = 1400;
```

7.2.2.1.19.2.21 NMatcher.FingersModePreciseBiometrics100CS Field

Represents Precise Biometrics [1654] 100CS scanner.

C#

```
public const uint FingersModePreciseBiometrics100CS = 500;
```

7.2.2.1.19.2.22 NMatcher.FingersModeSecugenHamster Field

Represents Secugen Hamster scanner.

C#

```
public const uint FingersModeSecugenHamster = 1100;
```

7.2.2.1.19.2.23 NMatcher.FingersModeStartekFM200 Field

Represents Startek FM200 scanner.

C#

```
public const uint FingersModeStartekFM200 = 1700;
```

7.2.2.1.19.2.24 NMatcher.FingersModeTacomaCmos Field

Represents Tacoma Cmos scanner.

C#

```
public const uint FingersModeTacomaCmos = 2100;
```

7.2.2.1.19.2.25 NMatcher.FingersModeTestechBioI Field

Represents Testech Bio I scanner.

C#

```
public const uint FingersModeTestechBioI = 1500;
```

7.2.2.1.19.2.26 NMatcher.FingersModeUpekTouchChip Field

Represents Upek Touch Chip scanner.

C#

```
public const uint FingersModeUpekTouchChip = 600;
```

7.2.2.1.19.2.27 NMatcher.ParameterFacesMatcherBase Field

Helper parameter for internal usage.

C#

```
public const ushort ParameterFacesMatcherBase = 11536;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.19.2.28 NMatcher.ParameterFacesMatchingSpeed Field

Specifies face templates matching speed. Can be Low or High matching speed.

C#

```
public const ushort ParameterFacesMatchingSpeed = ParameterFacesMatcherBase + 1;
```

Notes

High matching speed is equal to Medium.

7.2.2.1.19.2.29 NMatcher.ParameterFacesMatchingThresholdNew Field

Matters only if ParameterFusionType [1747] is SelectByFaceThenFuse(NMFusionType [1781]) set.

C#

```
public const ushort ParameterFacesMatchingThresholdNew = 215;
```

7.2.2.1.19.2.30 NMatcher.ParameterFingersFillFullMatchingDetails Field

Specifies whether full matching details are filled during the fingers matching.

C#

```
public const ushort ParameterFingersFillFullMatchingDetails = ParameterFingersMatcherBase +
900;
```

7.2.2.1.19.2.31 NMatcher.ParameterFingersMatcherBase Field

Helper parameter for internal usage.

C#

```
public const ushort ParameterFingersMatcherBase = 3536;
```

7.2.2.1.19.2.32 NMatcher.ParameterFingersMatchingSpeed Field

Specifies finger templates matching speed. Can be Low, Medium or High matching speed.

C#

```
public const ushort ParameterFingersMatchingSpeed = ParameterFingersMatcherBase + 220;
```

7.2.2.1.19.2.33 NMatcher.ParameterFingersMaximalRotation Field

Identifier specifying modulus of maximal rotation allowed between two matched NFRecords parameter of type N_TYPE_BYTE [576]. Parameter value is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).

C#

```
public const ushort ParameterFingersMaximalRotation = ParameterFingersMatcherBase + 201;
```

7.2.2.1.19.2.34 NMatcher.ParameterFingersMinMatchedCount Field

Identifier specifying minimal number of fingerprints from probe template to match parameter of type int. Parameter value is from zero to NFTemplate.MaxRecordCount.

C#

```
public const ushort ParameterFingersMinMatchedCount = ParameterFingersMatcherBase + 1201;
```

7.2.2.1.19.2.35 NMatcher.ParameterFingersMinMatchedCountThreshold Field

Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to Neurotec.Biometrics.NMatcher.ParameterFingersMinMatchedCount [1746] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public const ushort ParameterFingersMinMatchedCountThreshold = ParameterFingersMatcherBase
+ 1202;
```

7.2.2.1.19.2.36 NMatcher.ParameterFingersMode Field

Parameter specifying fingers mode (parameter value set). Parameter value can be one of the FingersModeXXX.

C#

```
public const ushort ParameterFingersMode = ParameterFingersMatcherBase + 1000;
```

7.2.2.1.19.2.37 NMatcher.ParameterFusionType Field

Identifier specifying fingerprints and faces fusion of type int. Parameter value can be one of the NMFusionType [1781] enumeration members.

C#

```
public const ushort ParameterFusionType = 200;
```

7.2.2.1.19.2.38 NMatcher.ParameterIrisesMatcherBase Field

Helper parameter for internal usage.

C#

```
public const ushort ParameterIrisesMatcherBase = 19536;
```

7.2.2.1.19.2.39 NMatcher.ParameterIrisesMatchingSpeed Field

Specifies iris templates matching speed. Can be Low, Medium or High matching speed.

C#

```
public const ushort ParameterIrisesMatchingSpeed = ParameterIrisesMatcherBase + 1;
```

Notes

High speed is equal to Medium.

7.2.2.1.19.2.40 NMatcher.ParameterIrisesMatchingThreshold Field

Matters only if ParameterFusionType [1747] is SelectByIrisThenFuse(NMFusionType [1781]) set.

C#

```
public const ushort ParameterIrisesMatchingThreshold = 220;
```

7.2.2.1.19.2.41 NMatcher.ParameterIrisesMaximalRotation Field

Defines the maximal rotation allowed between two matched irises. It is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).

C#

```
public const ushort ParameterIrisesMaximalRotation = ParameterIrisesMatcherBase + 2;
```

7.2.2.1.19.2.42 NMatcher.ParameterIrisesMinMatchedCount Field

Identifier specifying minimal number of irises from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NET_MAX_RECORD_COUNT.

C#

```
public const ushort ParameterIrisesMinMatchedCount = ParameterIrisesMatcherBase + 1201;
```

7.2.2.1.19.2.43 NMatcher.ParameterIrisesMinMatchedCountThreshold Field

Parameter specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Neurotec.Biometrics.NMatcher.ParameterIrisesMinMatchedCount [1747] parameter value). Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

C#

```
public const ushort ParameterIrisesMinMatchedCountThreshold = ParameterIrisesMatcherBase +
1202;
```

7.2.2.1.19.2.44 NMatcher.ParameterMatchingThreshold Field

Identifier specifying matching threshold (biggest allowed FAR) parameter of type int. Parameter value is equal to - 12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

C#

```
public const ushort ParameterMatchingThreshold = 100;
```

7.2.2.1.19.2.45 NMatcher.ParameterPalmsFillFullMatchingDetails Field

Specifies whether full matching details are filled during the palms matching.

C#

```
public const ushort ParameterPalmsFillFullMatchingDetails = ParameterPalmsMatcherBase + 900;
```

7.2.2.1.19.2.46 NMatcher.ParameterPalmsMatcherBase Field

Helper parameter for internal usage.

C#

```
public const ushort ParameterPalmsMatcherBase = 27536;
```

7.2.2.1.19.2.47 NMatcher.ParameterPalmsMatchingSpeed Field

Specifies palm templates matching speed. Can be Low, Medium or High matching speed.

C#

```
public const ushort ParameterPalmsMatchingSpeed = ParameterPalmsMatcherBase + 220;
```

7.2.2.1.19.2.48 NMatcher.ParameterPalmsMaximalRotation Field

Identifier specifying modulus of maximal rotation allowed between two matched NFRecords parameter of type N_TYPE_BYTE [576]. Parameter value is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).

C#

```
public const ushort ParameterPalmsMaximalRotation = ParameterPalmsMatcherBase + 201;
```

7.2.2.1.19.2.49 NMatcher.ParameterPalmsMinMatchedCount Field

Identifier specifying minimal number of palmprints from probe template to match parameter of type N_TYPE_INT [577]. Parameter value is from zero to NFT_MAX_RECORD_COUNT.

C#

```
public const ushort ParameterPalmsMinMatchedCount = ParameterPalmsMatcherBase + 1201;
```

7.2.2.1.19.2.50 NMatcher.ParameterPalmsMinMatchedCountThreshold Field

Identifier specifying matching threshold (biggest allowed FAR) when matching is stopped (with respect to Neurotec.Biometrics.NMatcher.ParameterPalmsMinMatchedCount [1748] parameter value) parameter of type N_TYPE_INT [577]. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

C#

```
public const ushort ParameterPalmsMinMatchedCountThreshold = ParameterPalmsMatcherBase +
1202;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.19.2.51 NMatcher.ParameterVoicesMatcherBase Field

Helper parameter for internal usage.

C#

```
public const ushort ParameterVoicesMatcherBase = 35536;
```

7.2.2.1.19.2.52 NMatcher.ParameterVoicesUniquePhrasesOnly Field

Identifier specifying whether each user in the system always uses only his own phrases which are unique throughout the whole database of users.

C#

```
public const ushort ParameterVoicesUniquePhrasesOnly = ParameterVoicesMatcherBase + 2;
```

7.2.2.1.19.2.53 NMatcher.PartFaces Field

Identifier specifying that retrieving or setting parameter for faces matcher (NlsMatcher [143]) contained in NMatcher [1737]. See GetParameter [1557], SetParameter [1563], GetStaticParameter

C#

```
public const ushort PartFaces = 20;
```

7.2.2.1.19.2.54 NMatcher.PartFingers Field

Identifier specifying that retrieving or setting parameter for fingers matcher (NfsMatcher) contained in NMatcher [1737]. See GetParameter [1557], SetParameter [1563], GetStaticParameter

C#

```
public const ushort PartFingers = 10;
```

7.2.2.1.19.2.55 NMatcher.PartIrises Field

Identifier specifying that retrieving or setting parameter for irises matcher (NesMatcher) contained in NMatcher [1737]. See GetParameter [1557], SetParameter [1563], GetStaticParameter

C#

```
public const ushort PartIrises = 30;
```

7.2.2.1.19.2.56 NMatcher.PartNone Field

Identifier specifying that retrieving or setting parameter for NMatcher [1737]. See GetParameter [1557], SetParameter [1563], GetStaticParameter

C#

```
public const ushort PartNone = 0;
```

7.2.2.1.19.2.57 NMatcher.PartPalms Field

Identifier specifying that retrieving or setting parameter for palms matcher (NfsMatcher) contained in NMatcher [1737]. See GetParameter [1557], SetParameter [1563], GetStaticParameter

C#

```
public const ushort PartPalms = 40;
```

7.2.2.1.19.3 NMatcher Methods

7.2.2.1.19.3.1 NMatcher.IdentifyEnd Method

Ends identification process.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public void IdentifyEnd();
```

7.2.2.1.19.3.2 IdentifyNext Method

7.2.2.1.19.3.2.1 NMatcher.IdentifyNext Method (NBuffer)

Compares the specified NTemplate [1762] with the one identification was started with.

C#

```
public int IdentifyNext(NBuffer template);
```

Parameters

Parameters	Description
NBuffer template	Memory buffer that contains packed NTemplate [1762].

Returns

The score of matched NTemplates.

Remarks

Method returns zero if similarity is less than matching threshold, i.e. two NTemplates do not match (see ParameterMatchingThreshold [1748] and SetParameter [1563] method),and is greater than or equal to matching threshold otherwise.

7.2.2.1.19.3.2.2 NMatcher.IdentifyNext Method (NBuffer, NMatchingDetails)

Compares the specified NTemplate [1762] with the one identification was started with.

C#

```
public int IdentifyNext(NBuffer template, NMatchingDetails matchingDetails);
```

Parameters

Parameters	Description
NBuffer template	Memory buffer that contains packed NTemplate [1762].
NMatchingDetails matchingDetails	Matching details which are filled with detailed matching info.

Returns

The score of matched NTemplates.

Remarks

Method returns zero if similarity is less than matching threshold, i.e. two NTemplates do not match (see ParameterMatchingThreshold [1748] and SetParameter [1563] method),and is greater than or equal to matching threshold otherwise.

7.2.2.1.19.3.2.3 NMatcher.IdentifyNext Method (byte[])

Compares the specified NTemplate [1762] with the one identification was started with.

C#

```
public int IdentifyNext(byte[] template);
```

Parameters

Parameters	Description
byte[] template	The byte array that contains packed NTemplate [1762].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

The score of matched NFTemplates.

Remarks

Method returns zero if similarity is less than matching threshold, i.e. two NTemplates do not match(see ParameterMatchingThreshold [1748] and SetParameter [1563] method),and is greater than or equal to matching threshold otherwise.

7.2.2.1.19.3.2.4 NMatcher.IdentifyNext Method (byte[], NMatchingDetails)

Compares the specified NTemplate [1762] with the one identification was started with.

C#

```
public int IdentifyNext(byte[] template, NMatchingDetails matchingDetails);
```

Parameters

Parameters	Description
byte[] template	The byte array that contains packed NTemplate [1762].
NMatchingDetails matchingDetails	Matching details which are filled with detailed matching info.

Returns

The score of matched NFTemplates.

Remarks

Method returns zero if similarity is less than matching threshold, i.e. two NTemplates do not match(see ParameterMatchingThreshold [1748] and SetParameter [1563] method),and is greater than or equal to matching threshold otherwise.

7.2.2.1.19.3.3 IdentifyStart Method

7.2.2.1.19.3.3.1 NMatcher.IdentifyStart Method (NBuffer)

Starts identification with the specified packed NTemplate [1762].

C#

```
public void IdentifyStart(NBuffer template);
```

Parameters

Parameters	Description
NBuffer template	Buffer that contains packed NTemplate [1762].

7.2.2.1.19.3.3.2 NMatcher.IdentifyStart Method (NBuffer, out NMatchingDetails)

Starts identification with the specified packed NTemplate [1762].

C#

```
public void IdentifyStart(NBuffer template, out NMatchingDetails matchingDetails);
```

Parameters

Parameters	Description
NBuffer template	Buffer that contains packed NTemplate [1762].
out NMatchingDetails matchingDetails	Matching details which provide details matching info.

7.2.2.1.19.3.3.3 NMatcher.IdentifyStart Method (byte[])

Starts identification with the specified packed NTemplate [1762].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public void IdentifyStart(byte[] template);
```

Parameters

Parameters	Description
byte[] template	The byte array with packed NFTemplate.

7.2.2.1.19.3.3.4 NMatcher.IdentifyStart Method (byte[], out NMatchingDetails)

Starts identification with the specified packed NTemplate [1762].

C#

```
public void IdentifyStart(byte[] template, out NMatchingDetails matchingDetails);
```

Parameters

Parameters	Description
byte[] template	The byte array with packed NFTemplate.
out NMatchingDetails matchingDetails	Matching details which provide details matching info.

7.2.2.1.19.3.4 NMatcher.PrepareTemplate Method

Changes specified template for faster matching.

C#

```
public void PrepareTemplate(NTemplate template);
```

Parameters

Parameters	Description
NTemplate template	Template to change for faster matching.

Remarks

This method is primarily used by cluster. It is not recommended to use it directly because template size can increase. When cluster is not used, this method can be used directly to prepare template for faster matching.

7.2.2.1.19.3.5 Verify Method

7.2.2.1.19.3.5.1 NMatcher.Verify Method (NBuffer, NBuffer)

Compares two packed templates.

C#

```
public int Verify(NBuffer template1, NBuffer template2);
```

Parameters

Parameters	Description
NBuffer template1	Buffer containing first packed template.
NBuffer template2	Buffer containing second packed template.

Returns

Verification score.

Remarks

Method returns zero if similarity is less than matching threshold, i.e. two templates do not match (see ParameterMatchingThreshold [1748] and SetParameter [1563] method), and is greater than or equal to matching threshold otherwise.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.19.3.5.2 NMatcher.Verify Method (NBuffer, NBuffer, out NMatchingDetails)

Compares two packed templates.

C#

```
public int Verify(NBuffer template1, NBuffer template2, out NMatchingDetails
matchingDetails);
```

Parameters

Parameters	Description
NBuffer template1	Buffer containing first packed template.
NBuffer template2	Buffer containing second packed template.
out NMatchingDetails matchingDetails	Detailed matching info.

Returns

Verification score.

Remarks

Method returns zero if similarity is less than matching threshold, i.e. two templates do not match (see ParameterMatchingThreshold [1748] and SetParameter [1563] method), and is greater than or equal to matching threshold otherwise.

7.2.2.1.19.3.5.3 NMatcher.Verify Method (byte[], byte[])

Compares two NTemplate [1762].

C#

```
public int Verify(byte[] template1, byte[] template2);
```

Parameters

Parameters	Description
byte[] template1	The byte array that contains packed NTemplate [1762].
byte[] template2	The byte array that contains packed NTemplate [1762].

Returns

Verification score.

Remarks

Method returns zero if similarity is less than matching threshold, i.e. two NTemplates do not match (see ParameterMatchingThreshold [1748] and SetParameter [1563] method), and is greater than or equal to matching threshold otherwise.

7.2.2.1.19.3.5.4 NMatcher.Verify Method (byte[], byte[], out NMatchingDetails)

Compares two NTemplate [1762].

C#

```
public int Verify(byte[] template1, byte[] template2, out NMatchingDetails matchingDetails);
```

Parameters

Parameters	Description
byte[] template1	The byte array that contains packed NTemplate [1762].
byte[] template2	The byte array that contains packed NTemplate [1762].
out NMatchingDetails matchingDetails	Detailed matching info.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Verification score.

Remarks

Method returns zero if similarity is less than matching threshold, i.e. two NTemplates do not match (see ParameterMatchingThreshold [1748] and SetParameter [1563] method), and is greater than or equal to matching threshold otherwise.

7.2.2.1.19.4 NMatcher Properties

7.2.2.1.19.4.1 NMatcher.FacesMatchingSpeed Property

Gets or sets faces matching speed.

C#

```
public NMatchingSpeed FacesMatchingSpeed;
```

Property value

One of the NMatchingSpeed [1781] values.

7.2.2.1.19.4.2 NMatcher.FacesMatchingThresholdNew Property

Matters only if ParameterFusionType [1747] is SelectByFaceThenFuse(NMFusionType [1781]) set.

C#

```
public int FacesMatchingThresholdNew;
```

Property value

The faces matching threshold.

7.2.2.1.19.4.3 NMatcher.FingersMatchingSpeed Property

Gets or sets fingers matching speed.

C#

```
public NMatchingSpeed FingersMatchingSpeed;
```

Property value

One of the NMatchingSpeed [1781] value.

7.2.2.1.19.4.4 NMatcher.FingersMaximalRotation Property

Gets or sets maximal rotation.

C#

```
public byte FingersMaximalRotation;
```

Property value

The finger maximal rotation.

7.2.2.1.19.4.5 NMatcher.FingersMinMatchedCount Property

Gets or sets minimal number of fingerprints from probe template to match parameter. The value is from zero to MaxRecordCount (NFTemplate).

C#

```
public int FingersMinMatchedCount;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.19.4.6 NMatcher.FingersMinMatchedCountThreshold Property

Gets or sets matching threshold (biggest allowed FAR) when matching is stopped (with respect to FingersMinMatchedFingerCount parameter value) parameter. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

C#

```
public int FingersMinMatchedCountThreshold;
```

7.2.2.1.19.4.7 NMatcher.FingersMode Property

Gets or sets scanners mode.

C#

```
public uint FingersMode;
```

Property value

One of the scanner types.

7.2.2.1.19.4.8 NMatcher.FusionType Property

Gets or sets fingerprints, irises and faces fusion type.

C#

```
public NMFusionType FusionType;
```

Property value

One of the NMFusionType [1781] values.

7.2.2.1.19.4.9 NMatcher.IrisesMatchingSpeed Property

Gets or sets irises matching speed.

C#

```
public NMatchingSpeed IrisesMatchingSpeed;
```

Property value

One of the NMatchingSpeed [1781] values.

7.2.2.1.19.4.10 NMatcher.IrisesMatchingThreshold Property

Matters only if ParameterFusionType [1747] is SelectByIrisThenFuse(NMFusionType [1781]) set.

C#

```
public int IrisesMatchingThreshold;
```

Property value

The faces matching threshold.

7.2.2.1.19.4.11 NMatcher.IrisesMaximalRotation Property

Gets or sets the maximal rotation allowed between two matched irises. It is specified in 180/128 degrees units and cannot be greater than 128 (180 degrees).

C#

```
public byte IrisesMaximalRotation;
```

Property value

The maximal rotation allowed between two matched irises.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.19.4.12 NMatcher.IrisesMinMatchedCount Property

Gets or sets minimal number of irises from probe template to match parameter. The value is from zero to MaxRecordCount (NFTemplate).

C#

```
public int IrisesMinMatchedCount;
```

7.2.2.1.19.4.13 NMatcher.IrisesMinMatchedCountThreshold Property

Gets or sets matching threshold (biggest allowed FAR) when matching is stopped (with respect to IrisesMinMatchedCount [1756] parameter value) parameter. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

C#

```
public int IrisesMinMatchedCountThreshold;
```

7.2.2.1.19.4.14 NMatcher.MatchingThreshold Property

Gets or sets the matching threshold.

C#

```
public int MatchingThreshold;
```

Property value

The matching speed.

7.2.2.1.19.4.15 NMatcher.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.2.1.19.4.16 NMatcher.NMatchingSpeedNativeType Property

Defines the native type of the matching speed.

C#

```
public static NType NMatchingSpeedNativeType;
```

7.2.2.1.19.4.17 NMatcher.NMFusionTypeNativeType Property

Defines the native type of the fusion type.

C#

```
public static NType NMFusionTypeNativeType;
```

7.2.2.1.19.4.18 NMatcher.PalmsMatchingSpeed Property

Gets or sets palms matching speed.

C#

```
public NMatchingSpeed PalmsMatchingSpeed;
```

Property value

One of the NMatchingSpeed [1781] value.

7.2.2.1.19.4.19 NMatcher.PalmsMaximalRotation Property

Gets or sets maximal rotation.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public byte PalmsMaximalRotation;
```

Property value

The palm maximal rotation.

7.2.2.1.19.4.20 NMatcher.PalmsMinMatchedCount Property

Gets or sets minimal number of palmprints from probe template to match parameter. The value is from zero to MaxRecordCount (NFTemplate).

C#

```
public int PalmsMinMatchedCount;
```

7.2.2.1.19.4.21 NMatcher.PalmsMinMatchedCountThreshold Property

Gets or sets matching threshold (biggest allowed FAR) when matching is stopped (with respect to PalmsMinMatchedCount [1757] parameter value) parameter. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

C#

```
public int PalmsMinMatchedCountThreshold;
```

7.2.2.1.19.4.22 NMatcher.VoicesUniquePhrasesOnly Property

Gets or sets identifier specifying whether each user in the system always uses only his own phrases which are unique throughout the whole database of users.

C#

```
public bool VoicesUniquePhrasesOnly;
```

Property value

true, if unique phrases are used; false, otherwise.

7.2.2.1.20 NMatchingDetails Class

Provides functionality for retrieving Neurotechnology templates matching details.

C#

```
public sealed class NMatchingDetails : NMatchingDetailsBase;
```

NMatchingDetails Classes

	Name	Description
	FaceCollection [1759]	Face templates matching details collection.
	FingerCollection [1759]	Finger templates matching details collection.
	IrisCollection [1759]	Iris templates matching details collection.
	PalmCollection [1760]	Palm templates matching details collection.
	VoiceCollection [1760]	Voice templates matching details collection.

Methods

	Name	Description
	NMatchingDetails [1758]	Creates new instance of NMatchingDetails class.

NMatchingDetails Properties

	Name	Description
	Faces [1760]	Gets collection of faces matching details.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	FacesMatchedIndex [1760]	Gets index of matched face in the second matched NLTemplate [1730] (or NTemplate [1762])).
	FacesScore [1760]	Gets similarity score of two matched face templates (NLTemplate [1730]).
	Fingers [1760]	Gets collection of fingers matching details.
	FingersScore [1760]	Gets similarity score of two matched finger templates (NFTemplate).
	Irises [1761]	Gets collection of irises matching details.
	IrisesScore [1761]	Gets similarity score of two matched iris templates (NETemplate).
S	NativeType [1761]	Defines the native type of the object.
	Palms [1761]	Gets collection of palms matching details.
	PalmsScore [1761]	Gets similarity score of two matched palm templates (NFTemplate).
	Voices [1761]	Gets collection of voices matching details.
	VoicesScore [1762]	Gets similarity score of two matched voice templates (NSTemplate).

7.2.2.1.20.1 NMatchingDetails Constructor

7.2.2.1.20.1.1 NMatchingDetails.NMatchingDetails Constructor (NBuffer)

Creates new instance of NMatchingDetails class.

C#

```
public NMatchingDetails(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	Memory buffer which contains templates matching details.

7.2.2.1.20.1.2 NMatchingDetails.NMatchingDetails Constructor (NBuffer, uint)

Creates new instance of NMatchingDetails class.

C#

```
public NMatchingDetails(NBuffer buffer, uint flags);
```

Parameters

Parameters	Description
NBuffer buffer	Memory buffer which contains templates matching details.
uint flags	Bitwise combination of zero or more flags that controls behavior of the constructor. Should be zero.

7.2.2.1.20.1.3 NMatchingDetails.NMatchingDetails Constructor (NBuffer, uint, out int)

Creates new instance of NMatchingDetails class.

C#

```
public NMatchingDetails(NBuffer buffer, uint flags, out int size);
```

Parameters

Parameters	Description
NBuffer buffer	Memory buffer which contains templates matching details.
uint flags	Bitwise combination of zero or more flags that controls behavior of the constructor. Should be zero.
out int size	Size of actually read buffer (in bytes).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.20.1.4 NMatchingDetails.NMatchingDetails Constructor (Stream)

Creates new instance of NMatchingDetails class.

C#

```
public NMatchingDetails(Stream stream);
```

Parameters

Parameters	Description
Stream stream	Stream containing matching details.

7.2.2.1.20.1.5 NMatchingDetails.NMatchingDetails Constructor (Stream, uint)

Creates new instance of NMatchingDetails class.

C#

```
public NMatchingDetails(Stream stream, uint flags);
```

Parameters

Parameters	Description
Stream stream	Stream containing matching details.
uint flags	Bitwise combination of zero or more flags that controls behavior of the constructor. Should be zero.

7.2.2.1.20.1.6 NMatchingDetails.NMatchingDetails Constructor (byte[])

Creates new instance of NMatchingDetails class.

C#

```
public NMatchingDetails(byte[] buffer);
```

Parameters

Parameters	Description
byte[] buffer	Byte buffer which contains templates matching details.

7.2.2.1.20.2 NMatchingDetails Classes

7.2.2.1.20.2.1 NMatchingDetails.FaceCollection Class

Face templates matching details collection.

C#

```
public sealed class FaceCollection : NObjectReadOnlyCollection<NLMatchingDetails>;
```

7.2.2.1.20.2.2 NMatchingDetails.FingerCollection Class

Finger templates matching details collection.

C#

```
public sealed class FingerCollection : NObjectReadOnlyCollection<NFMatchingDetails>;
```

7.2.2.1.20.2.3 NMatchingDetails.IrisCollection Class

Iris templates matching details collection.

C#

```
public sealed class IrisCollection : NObjectReadOnlyCollection<NEMatchingDetails>;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.20.2.4 NMatchingDetails.PalmCollection Class

Palm templates matching details collection.

C#

```
public sealed class PalmCollection : NObjectReadOnlyCollection<NFMatchingDetails>;
```

7.2.2.1.20.2.5 NMatchingDetails.VoiceCollection Class

Voice templates matching details collection.

C#

```
public sealed class VoiceCollection : NObjectReadOnlyCollection<NSMatchingDetails>;
```

7.2.2.1.20.3 NMatchingDetails Properties

7.2.2.1.20.3.1 NMatchingDetails.Faces Property

Gets collection of faces matching details.

C#

```
public FaceCollection Faces;
```

Property value

Faces matching details collection (FaceCollection [1759]).

7.2.2.1.20.3.2 NMatchingDetails.FacesMatchedIndex Property

Gets index of matched face in the second matched NLTemplate [1730] (or NTemplate [1762])).

C#

```
public int FacesMatchedIndex;
```

Property value

Index of matched face in the second template.

7.2.2.1.20.3.3 NMatchingDetails.FacesScore Property

Gets similarity score of two matched face templates (NLTemplate [1730]).

C#

```
public int FacesScore;
```

Property value

Similarity (matching) score.

7.2.2.1.20.3.4 NMatchingDetails.Fingers Property

Gets collection of fingers matching details.

C#

```
public FingerCollection Fingers;
```

Property value

Fingers matching details collection (FingerCollection [1759]).

7.2.2.1.20.3.5 NMatchingDetails.FingersScore Property

Gets similarity score of two matched finger templates (NFTemplate).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public int FingersScore;
```

Property value

Similarity (matching) score. of two fingerprint templates.

7.2.2.1.20.3.6 NMatchingDetails.Irises Property

Gets collection of irises matching details.

C#

```
public IrisCollection Irises;
```

Property value

Irises matching details collection (IrisCollection [1759]).

7.2.2.1.20.3.7 NMatchingDetails.IrisesScore Property

Gets similarity score of two matched iris templates (NETemplate).

C#

```
public int IrisesScore;
```

Property value

Similarity (matching) score. of two iris templates.

7.2.2.1.20.3.8 NMatchingDetails.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.2.1.20.3.9 NMatchingDetails.Palms Property

Gets collection of palms matching details.

C#

```
public PalmCollection Palms;
```

Property value

Palmprints matching details collection (PalmCollection [1760]).

7.2.2.1.20.3.10 NMatchingDetails.PalmsScore Property

Gets similarity score of two matched palm templates (NFTemplate).

C#

```
public int PalmsScore;
```

Property value

Similarity (matching) score. of two palm templates.

7.2.2.1.20.3.11 NMatchingDetails.Voices Property

Gets collection of voices matching details.

C#

```
public VoiceCollection Voices;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

Voices matching details collection (VoiceCollection [1760]).

7.2.2.1.20.3.12 NMatchingDetails.VoicesScore Property

Gets similarity score of two matched voice templates (NSTemplate).

C#

```
public int VoicesScore;
```

Property value

Similarity (matching) score. of two voice templates.

7.2.2.1.21 NTemplate Class

Provides functionality for packing, unpacking and editing Neurotechnology Templates (NTemplates).

C#

```
public sealed class NTemplate : NObject;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

Methods

NTemplate Class

	Name	Description
	NTemplate [1765]	Initializes a new instance of the NTemplate class.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.
S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NTemplate Class

	Name	Description
S	CalculateSize [1766]	Calculates the size of a packed NTemplate containing fingers, faces. irises and palms templates of the specified size.
S	Check [1766]	Checks if format of packed NTemplate format is correct.
	Clear [1767]	Removes all NFRecord objects.
S	GetSize [1767]	Gets the size of packed NTemplate.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTemplate Class

	Name	Description
	Faces [1768]	Gets NLTemplate [1730] object.
	Fingers [1768]	Gets NFTemplate object.
	Irises [1768]	Gets NETemplate object.
S	NativeType [1768]	Defines the native type of the object.
	Palms [1768]	Gets NFTemplate object.
	Voices [1768]	Gets NSTemplate object.

7.2.2.1.21.1 NTemplate Constructor

7.2.2.1.21.1.1 NTemplate.NTemplate Constructor ()

Initializes a new instance of the NTemplate class.

C#

```
public NTemplate();
```

7.2.2.1.21.1.2 NTemplate.NTemplate Constructor (NBuffer)

Initializes a new instance of the NTemplate class from byte array.

C#

```
public NTemplate(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] that contains packed NTemplate.

7.2.2.1.21.1.3 NTemplate.NTemplate Constructor (NBuffer, uint)

Initializes a new instance of the NTemplate class from byte array.

C#

```
public NTemplate(NBuffer buffer, uint flags);
```

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] that contains packed NTemplate.
uint flags	Reserved for future use. Must be zero.

7.2.2.1.21.1.4 NTemplate.NTemplate Constructor (NBuffer, uint, out int)

Initializes a new instance of the NTemplate class from byte array.

C#

```
public NTemplate(NBuffer buffer, uint flags, out int size);
```

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] that contains packed NTemplate.
uint flags	Reserved for future use. Must be zero.
out int size	[out] NTemplate size.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.21.1.5 NTemplate.NTemplate Constructor (byte[])

Initializes a new instance of the NTemplate class from byte array.

C#

```
public NTemplate(byte[] buffer);
```

Parameters

Parameters	Description
byte[] buffer	Byte array that contains packed NTemplate.

7.2.2.1.21.1.6 NTemplate.NTemplate Constructor (uint)

Initializes a new instance of the NTemplate class.

C#

```
public NTemplate(uint flags);
```

Parameters

Parameters	Description
uint flags	Bitwise combination of zero or more flags that controls behavior of the constructor. This parameter is reserved, must be zero.

7.2.2.1.21.2 NTemplate Methods

7.2.2.1.21.2.1 NTemplate.CalculateSize Method

Calculates the size of a packed NTemplate [1762] containing fingers, faces. irises and palms templates of the specified size.

C#

```
public static int CalculateSize(int fingersTemplateSize, int facesTemplateSize, int
irisesTemplateSize, int palmsTemplateSize, int voicesTemplateSize);
```

Parameters

Parameters	Description
int fingersTemplateSize	Size of packed NFTemplate.
int facesTemplateSize	Size of packed NLTemplate [1730].
int irisesTemplateSize	Size of packed NETemplate.
int palmsTemplateSize	Size of packed NFTemplate.
int voicesTemplateSize	Size of packed NSTemplate.

Returns

The value of NTemplate [1762] size.

7.2.2.1.21.2.2 Check Method

7.2.2.1.21.2.2.1 NTemplate.Check Method (NBuffer)

Checks if format of packed NTemplate [1762] format is correct.

C#

```
public static void Check(NBuffer buffer);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] that contains packed NTemplate [1762].

7.2.2.1.21.2.2.2 NTemplate.Check Method (byte[])

Checks if format of packed NTemplate [1762] format is correct.

C#

```
public static void Check(byte[] buffer);
```

Parameters

Parameters	Description
byte[] buffer	Memory buffer that contains packed NTemplate [1762].

Returns

True if NTemplate [1762] format is correct; false otherwise.

7.2.2.1.21.2.3 NTemplate.Clear Method

Removes all NFRecord objects.

C#

```
public void Clear();
```

7.2.2.1.21.2.4 GetSize Method

7.2.2.1.21.2.4.1 NTemplate.GetSize Method (NBuffer)

Gets the size of packed NTemplate [1762].

C#

```
public static int GetSize(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	Packed NTemplate [1762].

Returns

Size of NTemplate [1762].

7.2.2.1.21.2.4.2 NTemplate.GetSize Method (byte[])

Gets the size of packed NTemplate [1762].

C#

```
public static int GetSize(byte[] buffer);
```

Parameters

Parameters	Description
byte[] buffer	Packed NTemplate [1762].

Returns

Size of NTemplate [1762].

7.2.2.1.21.3 NTemplate Properties

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.1.21.3.1 NTemplate.Faces Property

Gets NLTemplate [1730] object.

C#

```
public NLTemplate Faces;
```

Property value

A NLTemplate [1730] object.

7.2.2.1.21.3.2 NTemplate.Fingers Property

Gets NFTemplate object.

C#

```
public NFTemplate Fingers;
```

Property value

A NFTemplate object.

7.2.2.1.21.3.3 NTemplate.Irises Property

Gets NETemplate object.

C#

```
public NETemplate Irises;
```

Property value

A NETemplate object.

7.2.2.1.21.3.4 NTemplate.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.2.1.21.3.5 NTemplate.Palms Property

Gets NFTemplate object.

C#

```
public NFTemplate Palms;
```

Property value

A NFTemplate object.

7.2.2.1.21.3.6 NTemplate.Voices Property

Gets NSTemplate object.

C#

```
public NSTemplate Voices;
```

Property value

A NSTemplate object.

7.2.2.2 Structs, Records, Enums

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.2.1 Neurotec.Biometrics.NBiometricStatus Enumeration

Enumerates biometric status values. These values are returned by NBiometricDevice [1142].

C#

```
public enum NBiometricStatus {
  None = 0,
  Ok = 1,
  Canceled = 2,
  Timeout = 3,
  CleaningNeeded = 10,
  ObjectsNotRemoved = 20,
  ObjectMissing = 21,
  ObjectNotFound = 22,
  TooFewObjects = 23,
  TooManyObjects = 24,
  BadObjectSequence = 25,
  SpoofDetected = 30,
  BadObject = 40,
  TooSoft = 51,
  TooHard = 52,
  BadPosition = 60,
  TooNorth = 61,
  TooEast = 62,
  TooSouth = 63,
  TooWest = 64,
  TooClose = 65,
  TooFar = 66,
  BadSpeed = 70,
  TooSlow = 71,
  TooFast = 72,
  BadSize = 80,
  TooShort = 81,
  TooLong = 82,
  TooNarrow = 83,
  TooWide = 84,
  TooSkewed = 91,
  WrongDirection = 92
}
```

Members

Members	Description
None = 0	Indicates that device is performing a concrete operation and can not return status value. E.g., fingerprint is swiped on a fingerprint scanner (operation is not finished yet and NBiometricDevice [1142] can not return any other value than None).
Ok = 1	Indicates succeeded operation.
Canceled = 2	Indicates canceled operation.
Timeout = 3	Indicates that operation timed-out.
CleaningNeeded = 10	Indicates that biometric device should be cleaned.
ObjectsNotRemoved = 20	Indicates that object(s) was not removed after NBiometricDevice [1142] finished operation.
ObjectMissing = 21	Indicates that object is missing.
ObjectNotFound = 22	Indicates that object(s) was not found.
TooFewObjects = 23	Indicates that too few objects was provided for device to succeed its operation. E.g., not enough fingerprints were scanned.
TooManyObjects = 24	Indicates that too many objects were provided for NBiometricDevice [1142] to perform operation.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

BadObjectSequence = 25	Indicates sequence of bad object(s).
SpoofDetected = 30	Indicates that spoof object was detected.
BadObject = 40	Indicates bad object.
TooSoft = 51	Indicates that fingerprint was placed too soft on a device.
TooHard = 52	Indicates that fingerprint was placed too hard on a device.
BadPosition = 60	Indicates bad fingerprint position.
TooNorth = 61	Indicated that object was too north from device.
TooEast = 62	Indicated that object was too east from device.
TooSouth = 63	Indicated that object was too south from device.
TooWest = 64	Indicated that object was too west from device.
TooClose = 65	Indicated that object was too close to device.
TooFar = 66	Indicated that object was too far from device.
BadSpeed = 70	Indicates too slow/fast moving object during operation. E.g., fingerprint was swiped too slow (too fast) when fingerprint scanning operation was performed on a scanner.
TooSlow = 71	Indicates that object was moving too slow. E.g., fingerprint was swiped through scanner too slow.
TooFast = 72	Indicates that object was moving too fast. E.g., fingerprint was swiped through scanner too fast.
BadSize = 80	Indicates object's bad size.
TooShort = 81	Indicates that object is too short for NBiometricDevice [1142] to perform operation.
TooLong = 82	Indicates that object is too long for NBiometricDevice [1142] to perform operation.
TooNarrow = 83	Indicates that object is too narrow for NBiometricDevice [1142] to perform operation.
TooWide = 84	Indicates that object is too wide for NBiometricDevice [1142] to perform operation.
TooSkewed = 91	Indicates that object is too skewed.
WrongDirection = 92	Indicates that object was moving to wrong direction.

7.2.2.2.2 Neurotec.Biometrics.NBiometricSubtype Enumeration

Specifies the subtype of biometric data.

C#

```
[Flags]
public enum NBiometricSubtype {
  None = 0x000000,
  Left = 0x000001,
  Right = 0x000002,
  LeftThumb = 0x000004,
  LeftPointerFinger = 0x000008,
  LeftMiddleFinger = 0x000010,
  LeftRingFinger = 0x000020,
  LeftLittleFinger = 0x000040,
  RightThumb = 0x000080,
  RightPointerFinger = 0x000100,
  RightMiddleFinger = 0x000200,
  RightRingFinger = 0x000400,
  RightLittleFinger = 0x000800,
  LeftPalm = 0x001000,
  LeftBackOfHand = 0x002000,
  LeftWrist = 0x004000,
  RightPalm = 0x008000,
  RightBackOfHand = 0x010000,
  RightWrist = 0x020000
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
}
```

Members

Members	Description
None = 0x000000	Undefined.
Left = 0x000001	Left.
Right = 0x000002	Right.
LeftThumb = 0x000004	Left thumb.
LeftPointerFinger = 0x000008	Left pointer finger.
LeftMiddleFinger = 0x000010	Left pointer finger.
LeftRingFinger = 0x000020	Left ring finger.
LeftLittleFinger = 0x000040	Left little finger.
RightThumb = 0x000080	Right thumb.
RightPointerFinger = 0x000100	Right pointer finger.
RightMiddleFinger = 0x000200	Right middle finger.
RightRingFinger = 0x000400	Right ring finger.
RightLittleFinger = 0x000800	Right little finger.
LeftPalm = 0x001000	Left palm.
LeftBackOfHand = 0x002000	Left back of hand.
LeftWrist = 0x004000	Left wrist.
RightPalm = 0x008000	Right palm.
RightBackOfHand = 0x010000	Right back of hand.
RightWrist = 0x020000	Right wrist.

7.2.2.2.3 Neurotec.Biometrics.NBiometricType Enumeration

Specifies the type of biometric data.

C#

```
[Flags]
public enum NBiometricType {
  None = 0x000000,
  MultipleBiometrics = 0x000001,
  Face = 0x000002,
  FacialFeatures = Face,
  Voice = 0x000004,
  Finger = 0x000008,
  Fingerprint = Finger,
  Iris = 0x000010,
  Retina = 0x000020,
  HandGeometry = 0x000040,
  SignatureOrSign = 0x000080,
  SignatureDynamics = SignatureOrSign,
  Keystroke = 0x000100,
  KeystrokeDynamics = Keystroke,
  LipMovement = 0x000200,
  ThermalFace = 0x000400,
  ThermalHand = 0x000800,
  ThermalFaceImage = ThermalFace,
  ThermalHandImage = ThermalHand,
  Gait = 0x001000,
  Scent = 0x002000,
  BodyOdor = Scent,
  Dna = 0x004000,
  Ear = 0x008000,
  EarShape = Ear,
  FingerGeometry = 0x010000,
  Palm = 0x020000,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
    PalmPrint = Palm,
    Vein = 0x040000,
    VeinPattern = Vein,
    Foot = 0x080000,
    FootPrint = Foot,
    PalmGeometry = 0x100000
  }
```

Members

Members	Description
None = 0x000000	Undefined.
MultipleBiometrics = 0x000001	Multiple biometrics.
Face = 0x000002	Face matching.
FacialFeatures = Face	Facial features (same as face).
Voice = 0x000004	Voice matching.
Finger = 0x000008	Finger matching.
Fingerprint = Finger	Fingerprint matching (same as finger).
Iris = 0x000010	Iris matching.
Retina = 0x000020	Retina mathcing.
HandGeometry = 0x000040	Hand geometry.
SignatureOrSign = 0x000080	Signature or sign.
SignatureDynamics = SignatureOrSign	Signature dynamics (same as signature or sign).
Keystroke = 0x000100	Keystroke matching.
KeystrokeDynamics = Keystroke	Keystroke dynamics (same as keystroke).
LipMovement = 0x000200	Lip movement.
ThermalFace = 0x000400	Thermal face matching.
ThermalHand = 0x000800	Thermal hand matching.
ThermalFaceImage = ThermalFace	Thermal face image (same as thermal face).
ThermalHandImage = ThermalHand	Thermal hand image (same as thermal hand).
Gait = 0x001000	Gait matching.
Scent = 0x002000	Scent matching.
BodyOdor = Scent	Body odor (same as scent).
Dna = 0x004000	DNA matching.
Ear = 0x008000	Ear matching.
EarShape = Ear	Ear shape (same as ear).
FingerGeometry = 0x010000	Finger geometry.
Palm = 0x020000	Palm mathcing.
PalmPrint = Palm	Palm print (same as palm).
Vein = 0x040000	Vein matching.
VeinPattern = Vein	Vein patter (same as vein).
Foot = 0x080000	Foot mathcing.
FootPrint = Foot	Foot print (same as foot).
PalmGeometry = 0x100000	Palm geometry.

7.2.2.2.4 Neurotec.Biometrics.NEPosition Enumeration

Holds information on iris image position.

C#

```
  public enum NEPosition {
    Unknown = 0,
    Right = 1,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
    Left = 2,
    Both = 3
  }
```

Members

Members	Description
Unknown = 0	Unknown iris position.
Right = 1	Right eye.
Left = 2	Left eye.
Both = 3	Both eyes.

Remarks

When both eyes are retuned (Both), it can be cropped using NImage.Crop method. Rectangle coordinates can be retrieved using NEAttributes.BoundingRect property.

7.2.2.2.5 Neurotec.Biometrics.NFImpressionType Enumeration

Specifies the impression types.

C#

```
  public enum NFImpressionType {
    LiveScanPlain = 0,
    LiveScanRolled = 1,
    NonliveScanPlain = 2,
    NonliveScanRolled = 3,
    LatentImpression = 4,
    LatentTracing = 5,
    LatentPhoto = 6,
    LatentLift = 7,
    LiveScanVerticalSwipe = 8,
    Swipe = LiveScanVerticalSwipe,
    LiveScanContactless = 9,
    LiveScanPalm = 10,
    NonliveScanPalm = 11,
    LatentPalmImpression = 12,
    LatentPalmTracing = 13,
    LatentPalmPhoto = 14,
    LatentPalmLift = 15,
    LiveScanOpticalContactPlain = 20,
    LiveScanOpticalContactRolled = 21,
    LiveScanNonOpticalContactPlain = 22,
    LiveScanNonOpticalContactRolled = 23,
    LiveScanOpticalContactlessPlain = 24,
    LiveScanOpticalContactlessRolled = 25,
    LiveScanNonOpticalContactlessPlain = 26,
    LiveScanNonOpticalContactlessRolled = 27,
    Other = 28,
    Unknown = 29
  }
```

Members

Members	Description
LiveScanPlain = 0	Live-scanned plain fingerprint.
LiveScanRolled = 1	Live-scanned rolled fingerprint.
NonliveScanPlain = 2	Nonlive-scanned (from paper) plain fingerprint.
NonliveScanRolled = 3	Nonlive-scanned (from paper) rolled fingerprint.
LatentImpression = 4	Latent impression fingerprint.
LatentTracing = 5	Latent tracing fingerprint.
LatentPhoto = 6	Latent photo fingerprint.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

LatentLift = 7	Latent lift fingerprint.
LiveScanVerticalSwipe = 8	Live-scanned fingerprint by sliding the finger across a "swipe" sensor.
Swipe = LiveScanVerticalSwipe	Live-scanned fingerprint by sliding the finger across a "swipe" sensor.
LiveScanContactless = 9	Live-scanned fingerprint using contactless device.
LiveScanPalm = 10	Live scanned palm.
NonliveScanPalm = 11	Non-live scanned palm.
LatentPalmImpression = 12	Latent palm impression.
LatentPalmTracing = 13	Latent palm tracing.
LatentPalmPhoto = 14	Latent palm photo.
LatentPalmLift = 15	Latent palm lift.
LiveScanOpticalContactPlain = 20	Live plain optical scan with contact.
LiveScanOpticalContactRolled = 21	Live rolled optical scan with contact.
LiveScanNonOpticalContactPlain = 22	Live plain non-optical scan with contact.
LiveScanNonOpticalContactRolled = 23	Live rolled non-optical scan with contact.
LiveScanOpticalContactlessPlain = 24	Live plain optical contactless scan.
LiveScanOpticalContactlessRolled = 25	Live rolled optical contactless scan.
LiveScanNonOpticalContactlessPlain = 26	Live plain non-optical contactless scan.
LiveScanNonOpticalContactlessRolled = 27	Live rolled non-optical contactless scan.
Other = 28	Other impression types.
Unknown = 29	Unknown impression type.

7.2.2.2.6 Neurotec.Biometrics.NFMinutiaFormat Enumeration

Specifies minutia format.

C#

```
[Flags]
public enum NFMinutiaFormat {
  None = 0,
  HasQuality = 1,
  HasCurvature = 2,
  HasG = 4
}
```

Members

Members	Description
None = 0	None of other values.
HasQuality = 1	Indicates that Neurotec.Biometrics.NFMinutia.Quality [1691] field contains meaningful value and is preserved during unpacking/packing of NFRecord.
HasCurvature = 2	Indicates that Neurotec.Biometrics.NFMinutia.Curvature [1690] field contains meaningful value and is preserved during unpacking/packing of NFRecord.
HasG = 4	Indicates that Neurotec.Biometrics.NFMinutia.G [1691] field contains meaningful value and is preserved during unpacking/packing of NFRecord.

7.2.2.2.7 Neurotec.Biometrics.NFMinutiaType Enumeration

Specifies types of minutia.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public enum NFMinutiaType {
  Unknown = 0,
  End = 1,
  Bifurcation = 2,
  Other = 3
}
```

Members

Members	Description
Unknown = 0	The type of the minutia is unknown.
End = 1	The minutia that is an end of a ridge.
Bifurcation = 2	The minutia that is a bifurcation of a ridge.
Other = 3	Other.

7.2.2.2.8 Neurotec.Biometrics.NFPatternClass Enumeration

Specifies pattern class of the fingerprint.

C#

```
public enum NFPatternClass {
  Unknown = 0,
  PlainArch = 1,
  TentedArch = 2,
  RadialLoop = 3,
  UlnarLoop = 4,
  PlainWhorl = 5,
  CentralPocketLoop = 6,
  DoubleLoop = 7,
  AccidentalWhorl = 8,
  Whorl = 9,
  RightSlantLoop = 10,
  LeftSlantLoop = 11,
  Scar = 12,
  Amputation = 15
}
```

Members

Members	Description
Unknown = 0	Unknown pattern class.
PlainArch = 1	Plain arch pattern class.
TentedArch = 2	Tented arch pattern class.
RadialLoop = 3	Radial loop pattern class.
UlnarLoop = 4	Ulnar loop pattern class.
PlainWhorl = 5	Plain whorl pattern class.
CentralPocketLoop = 6	Central pocket loop pattern class.
DoubleLoop = 7	Double loop pattern class.
AccidentalWhorl = 8	Accidental whorl pattern class.
Whorl = 9	Whorl pattern class.
RightSlantLoop = 10	Right slant loop pattern class.
LeftSlantLoop = 11	Left slant loop pattern class.
Scar = 12	Scar. Pattern class is not available.
Amputation = 15	Amputation. Pattern class is not available.

Remarks

This enumeration is implemented according to ANSI/NIST-ITL 1-2000 standard.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.2.2.9 Neurotec.Biometrics.NFPosition Enumeration

Specifies finger position.

C#

```
public enum NFPosition {
  Unknown = 0,
  RightThumb = 1,
  RightIndexFinger = 2,
  RightIndex = RightIndexFinger,
  RightMiddleFinger = 3,
  RightMiddle = RightMiddleFinger,
  RightRingFinger = 4,
  RightRing = RightRingFinger,
  RightLittleFinger = 5,
  RightLittle = RightLittleFinger,
  LeftThumb = 6,
  LeftIndexFinger = 7,
  LeftIndex = LeftIndexFinger,
  LeftMiddleFinger = 8,
  LeftMiddle = LeftMiddleFinger,
  LeftRingFinger = 9,
  LeftRing = LeftRingFinger,
  LeftLittleFinger = 10,
  LeftLittle = LeftLittleFinger,
  PlainRightThumb = 11,
  PlainLeftThumb = 12,
  PlainRightFourFingers = 13,
  PlainLeftFourFingers = 14,
  PlainThumbs = 15,
  UnknownPalm = 20,
  RightFullPalm = 21,
  RightWritersPalm = 22,
  LeftFullPalm = 23,
  LeftWritersPalm = 24,
  RightLowerPalm = 25,
  RightUpperPalm = 26,
  LeftLowerPalm = 27,
  LeftUpperPalm = 28,
  RightOther = 29,
  LeftOther = 30,
  RightInterdigital = 31,
  RightThenar = 32,
  RightHypothenar = 33,
  LeftInterdigital = 34,
  LeftThenar = 35,
  LeftHypothenar = 36,
  RightIndexMiddleFingers = 40,
  RightMiddleRingFingers = 41,
  RightRingLittleFingers = 42,
  LeftIndexMiddleFingers = 43,
  LeftMiddleRingFingers = 44,
  LeftRingLittleFingers = 45,
  RightIndexLeftIndexFingers = 46,
  RightIndexMiddleRingFingers = 47,
  RightMiddleRingLittleFingers = 48,
  LeftIndexMiddleRingFingers = 49,
  LeftMiddleRingLittleFingers = 50,
  UnknownTwoFingers = -2,
  UnknownThreeFingers = -3,
  UnknownFourFingers = -4
}
```

Members

Members	Description
Unknown = 0	Unknown finger.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

RightThumb = 1	Thumb of the right hand.
RightIndexFinger = 2	Index finger of the right hand.
RightIndex = RightIndexFinger	Index finger of the right hand.
RightMiddleFinger = 3	Middle finger of the right hand.
RightMiddle = RightMiddleFinger	Middle finger of the right hand.
RightRingFinger = 4	Ring finger of the right hand.
RightRing = RightRingFinger	Ring finger of the right hand.
RightLittleFinger = 5	Little finger of the right hand.
RightLittle = RightLittleFinger	LIttle finger of the right hand.
LeftThumb = 6	Thumb of the left hand.
LeftIndexFinger = 7	Index finger of the left hand.
LeftIndex = LeftIndexFinger	Index finger of the left hand.
LeftMiddleFinger = 8	Middle finger of the left hand.
LeftMiddle = LeftMiddleFinger	Middle finger of the left hand.
LeftRingFinger = 9	Ring finger of the left hand.
LeftRing = LeftRingFinger	Ring finger of the left hand.
LeftLittleFinger = 10	Little finger of the left hand.
LeftLittle = LeftLittleFinger	Little finger of the left hand.
PlainRightThumb = 11	Specifies plain right thumb.
PlainLeftThumb = 12	Specifies leftplain thumb.
PlainRightFourFingers = 13	Specifies plain right four fingers.
PlainLeftFourFingers = 14	Specifies left four fingers.
PlainThumbs = 15	Specifies plain thumbs.
UnknownPalm = 20	Unknown palm.
RightFullPalm = 21	Specifies full left palm.
RightWritersPalm = 22	Specifies right writers palm.
LeftFullPalm = 23	Specifies full left palm.
LeftWritersPalm = 24	Specifies left writers palm.
RightLowerPalm = 25	Specifies right lower palm.
RightUpperPalm = 26	Specifies right upper palm.
LeftLowerPalm = 27	Specifies lower palm.
LeftUpperPalm = 28	Specifies right upper palm.
RightOther = 29	Some other right palm.
LeftOther = 30	Some other left hand palm.
RightInterdigital = 31	Specifies right interdigital.
RightThenar = 32	Specifies right thenar.
RightHypothenar = 33	Specifies right hypothenar.
LeftInterdigital = 34	Specifies left interdigital.
LeftThenar = 35	Specifies left thenar.
LeftHypothenar = 36	Specifies left hypothenar.
RightIndexMiddleFingers = 40	Specifies right index and middle fingers.
RightMiddleRingFingers = 41	Specifies right middle and ring fingers.
RightRingLittleFingers = 42	Specifies right ring and little fingers.
LeftIndexMiddleFingers = 43	Specifies left index and middle fingers.
LeftMiddleRingFingers = 44	Specifies left middle and ring fingers.
LeftRingLittleFingers = 45	Specifies left ring and little fingers.
RightIndexLeftIndexFingers = 46	Specifies right and left index fingers.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

RightIndexMiddleRingFingers = 47	Specifies right index, middle and ring fingers.
RightMiddleRingLittleFingers = 48	Specifies right middle, ring and little fingers.
LeftIndexMiddleRingFingers = 49	Specifies left index, middle and ring fingers.
LeftMiddleRingLittleFingers = 50	Specifies left middle, ring and little fingers.
UnknownTwoFingers = -2	Specifies two unknown fingers.
UnknownThreeFingers = -3	Specifies three unknown fingers.
UnknownFourFingers = -4	Specifies four unknown fingers.

7.2.2.2.10 Neurotec.Biometrics.NFRidgeCountsType Enumeration

Specifies type of ridge counts contained in NFRecord.

C#

```
public enum NFRidgeCountsType {
  None = 0,
  FourNeighbors = 1,
  EightNeighbors = 2,
  FourNeighborsWithIndexes = 5,
  EightNeighborsWithIndexes = 6,
  Unspecified = 128+4
}
```

Members

Members	Description
None = 0	The NFRecord does not contain ridge counts.
FourNeighbors = 1	The NFRecord contains ridge counts to closest minutia in each of the four sectors of each minutia. First sector starts at minutia angle.
EightNeighbors = 2	The NFRecord contains ridge counts to closest minutia in each of the eight sectors of each minutia. First sector starts at minutia angle.
FourNeighborsWithIndexes = 5	The NFRecord contains ridge counts to four neighbors of each minutia.
EightNeighborsWithIndexes = 6	The NFRecord contains ridge counts to eight neighbors of each minutia.
Unspecified = 128+4	For internal use.

Remarks

Extracted template with EightNeighborsWithIndexes parameter is bigger than the template extracted with EightNeighbors parameter. Templates extracted with EightNeighborsWithIndexes parameter is faster than the templates extracted with EightNeighbors parameter. Extracted template with FourNeighborsWithIndexes parameter is bigger than the template extracted with FourNeighbors parameter. Templates extracted with FourNeighborsWithIndexes parameter is faster than the templates extracted with FourNeighbors parameter.

7.2.2.2.11 Neurotec.Biometrics.NGender Enumeration

Enumerates gender constants.

C#

```
public enum NGender {
  Unspecified = 0,
  Male = 1,
  Female = 2,
  Unknown = 255
}
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Members

Members	Description
Unspecified = 0	Unspecified gender.
Male = 1	Male.
Female = 2	Female.
Unknown = 255	Unknown.

Remarks

See NGender in Reference (C/C++) chapter.

7.2.2.2.12 Neurotec.Biometrics.NleExtractionStatus Enumeration

Specifies face features extraction status.

C#

```
public enum NleExtractionStatus {
  None = 0,
  TemplateCreated = 1,
  FaceNotDetected = 2,
  EyesNotDetected = 3,
  FaceTooCloseToImageBorder = 4,
  QualityCheckGrayscaleDensityFailed = 100,
  QualityCheckExposureFailed = 101,
  QualityCheckSharpnessFailed = 102,
  LivenessCheckFailed = 200,
  GeneralizationFailed = 300
}
```

Members

Members	Description
None = 0	No extraction status usage.
TemplateCreated = 1	Template created.
FaceNotDetected = 2	Face not detected.
EyesNotDetected = 3	Eyes not detected.
FaceTooCloseToImageBorder = 4	Face too close to image border.
QualityCheckGrayscaleDensityFailed = 100	Quality check grayscale density failed.
QualityCheckExposureFailed = 101	Quality check exposure failed.
QualityCheckSharpnessFailed = 102	Quality check sharpness failed.
LivenessCheckFailed = 200	Liveness check failed.
GeneralizationFailed = 300	Generalization failed.

7.2.2.2.13 Neurotec.Biometrics.NleTemplateSize Enumeration

Specifies NLRecord [1724] type by its size.

C#

```
public enum NleTemplateSize {
  Small = 0,
  Compact = 1,
  Medium = 64,
  Large = 128
}
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Members

Members	Description
Small = 0	The small NLRecord [1724] size (~4KB).
Compact = 1	The compact NLRecord [1724] size (~2.2KB, used for smartcards).
Medium = 64	The medium NLRecord [1724] size (~20KB).
Large = 128	The large NLRecord [1724] size (~36KB).

7.2.2.2.14 Neurotec.Biometrics.NLExpression Enumeration

Enumerates facial expression constants.

C#

```
public enum NLExpression {
  Unspecified = 0,
  Neutral = 1,
  Smile = 2,
  SmileOpenedJaw = 3,
  RaisedBrows = 4,
  EyesAway = 5,
  Squinting = 6,
  Frowning = 7,
  Unknown = 0xFFFF
}
```

Members

Members	Description
Unspecified = 0	Unspecified expression.
Neutral = 1	Neutral.
Smile = 2	Smiling.
SmileOpenedJaw = 3	Smiling with jaw open.
RaisedBrows = 4	Eyebrows raised.
EyesAway = 5	Eyes looking away.
Squinting = 6	Squinting.
Frowning = 7	Frowning.
Unknown = 0xFFFF	Unknown expression.

Remarks

See NLExpression in Reference (C/C++) chapter.

7.2.2.2.15 Neurotec.Biometrics.NLProperties Enumeration

Enumerates basic face properties constants

C#

```
[Flags]
public enum NLProperties {
  NotSpecified = 0,
  Specified = 0x000001,
  Glasses = 0x000002,
  Moustache = 0x000004,
  Beard = 0x000008,
  TeethVisible = 0x000010,
  Blink = 0x000020,
  MouthOpen = 0x000040,
  LeftEyePatch = 0x000080,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
    RightEyePatch = 0x000100,
    BothEyePatch = 0x000200,
    DarkGlasses = 0x000400,
    DistortingCondition = 0x000800,
    Hat = 0x01000000,
    Scarf = 0x02000000,
    NoEar = 0x04000000
  }
```

Members

Members	Description
NotSpecified = 0	Face properties not specified.
Specified = 0x000001	Face properties specified.
Glasses = 0x000002	Wearing glasses.
Moustache = 0x000004	Moustaches.
Beard = 0x000008	Beard.
TeethVisible = 0x000010	Teeth are visible.
Blink = 0x000020	Eye blink.
MouthOpen = 0x000040	Mouth is open.
LeftEyePatch = 0x000080	Patch on left eye.
RightEyePatch = 0x000100	Patch on right eye.
BothEyePatch = 0x000200	Patch on both eyes.
DarkGlasses = 0x000400	Wearing dark glasses.
DistortingCondition = 0x000800	Distorting conditons.
Hat = 0x01000000	Wearing a hat.
Scarf = 0x02000000	Wearing a scarf.
NoEar = 0x04000000	One ear missing.

Remarks

See NLProperties in Reference (C/C++) chapter.

7.2.2.2.16 Neurotec.Biometrics.NMatchingSpeed Enumeration

Specifies biometric templates matching speed.

C#

```
  public enum NMatchingSpeed {
    Low = 0,
    Medium = 128,
    High = 256
  }
```

Members

Members	Description
Low = 0	Low matching speed (slower but more accurate).
Medium = 128	Medium matching speed.
High = 256	High matching speed (faster but less accurate).

Remarks

Slow matcher has low speed and medium speed matching options. Fast matcher has low, medium and high speed options.

7.2.2.2.17 Neurotec.Biometrics.NMFusionType Enumeration

Specifies the fingerprints, palmprints, faces and irises matching fusion.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public enum NMFusionType {
  FuseAlways = 0,
  SelectByFaceThenFuse = 1,
  SelectByIrisThenFuse = 2
}
```

Members

Members	Description
FuseAlways = 0	Always matches fingerprints, palmprints, faces and irises. The results of are fused.
SelectByFaceThenFuse = 1	Matches faces. If the result is greater than ParameterFacesMatchingThresholdNew, then matches fingerprints, palms and irises also and fuses results; otherwise, score is zero.
SelectByIrisThenFuse = 2	Matches irises. If the result is greater than ParameterIrisesMatchingThreshold, then matches fingerprints, palms and faces also and fuses results; otherwise, score is zero.

7.2.3 Neurotec.Biometrics.Gui Namespace

Contains classes that provide infrastructure for Neurotechnology Biometrics [1654] components.

Classes

	Name	Description
	NEView [1782]	Provides functionality for managing iris scans.
	NFView [1784]	Provides functionality for managing fingerprint scans and templates.
	NLView [1787]	Provides functionality for managing face pictures.

Module

.NET Reference [1468]

7.2.3.1 Classes

7.2.3.1.1 NEView Class

Provides functionality for managing iris scans.

C#

```
public class NEView : Neurotec.Gui.NView;
```

Methods

	Name	Description
	NEView [1783]	Initializes a new instance of NEView class.

NEView Properties

	Name	Description
	Image [1783]	Gets or sets the current image.
	InnerBoundaryColor [1783]	Gets or sets the inner boundary color.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	InnerBoundaryWidth [1783]	Gets or sets the inner boundary width.
	OuterBoundaryColor [1783]	Gets or sets the outer boundary color.
	OuterBoundaryWidth [1783]	Gets or sets the outer boundary width.
	SegmentationDetails [1784]	Gets or sets the segmentation details.

7.2.3.1.1.1 NEView.NEView Constructor

Initializes a new instance of NEView class.

C#

```
public NEView();
```

7.2.3.1.1.2 NEView Properties

7.2.3.1.1.2.1 NEView.Image Property

Gets or sets the current image.

C#

```
public Bitmap Image;
```

Property value

The current image.

7.2.3.1.1.2.2 NEView.InnerBoundaryColor Property

Gets or sets the inner boundary color.

C#

```
public Color InnerBoundaryColor;
```

Property value

The inner boundary color.

7.2.3.1.1.2.3 NEView.InnerBoundaryWidth Property

Gets or sets the inner boundary width.

C#

```
public int InnerBoundaryWidth;
```

Property value

The inner boundary width.

7.2.3.1.1.2.4 NEView.OuterBoundaryColor Property

Gets or sets the outer boundary color.

C#

```
public Color OuterBoundaryColor;
```

Property value

The outer boundary color.

7.2.3.1.1.2.5 NEView.OuterBoundaryWidth Property

Gets or sets the outer boundary width.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public int OuterBoundaryWidth;
```

Property value

The outer boundary width.

7.2.3.1.1.2.6 NEView.SegmentationDetails Property

Gets or sets the segmentation details.

C#

```
public NeeSegmentationDetails SegmentationDetails;
```

Property value

The segmentation details.

7.2.3.1.2 NFView Class

Provides functionality for managing fingerprint scans and templates.

C#

```
public class NFView : NFingerView;
```

NFView Classes

	Name	Description
	PointerTool [1785]	Provides tools for pointer management.

NFView Events

	Name	Description
	HoveredDoubleCoreIndexChanged [1786]	Handler for event which occurs when index of the hovered double core has been changed.
	SelectedDoubleCoreIndexChanged [1786]	Handler for event which occurs when index of the selected double core has been changed.

NFView Methods

	Name	Description
	GetDoubleCoreAtScreenPoint [1785]	Gets the double core index at the specified point.
	OnHoveredDoubleCoreIndexChanged [1785]	Occurs when index of the hovered double core has been changed.
	OnSelectedDoubleCoreIndexChanged [1785]	Occurs when index of the selected double core has been changed.

NFView Properties

	Name	Description
	SelectedCoreIndex [1785]	Gets or sets the index of the selected core.
	SelectedDeltaIndex [1786]	Gets or sets the index of the selected delta.
	SelectedDoubleCoreIndex [1786]	Gets or sets the index of the selected double core.
	SelectedMinutiaIndex [1786]	Gets or sets the index of the selected minutia.
	Template [1786]	Gets or sets the template.

7.2.3.1.2.1 NFView Classes

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.3.1.2.1.1 NFView.PointerTool Class

Provides tools for pointer management.

C#

```
public new class PointerTool : NFingerView.PointerTool;
```

7.2.3.1.2.2 NFView Methods

7.2.3.1.2.2.1 NFView.GetDoubleCoreAtScreenPoint Method

Gets the double core index at the specified point.

C#

```
public int GetDoubleCoreAtScreenPoint(int x, int y);
```

Parameters

Parameters	Description
int x	X-coordinate of the point.
int y	Y-coordinate of the point.

Returns

Index of the double core.

7.2.3.1.2.2.2 NFView.OnHoveredDoubleCoreIndexChanged Method

Occurs when index of the hovered double core has been changed.

C#

```
public void OnHoveredDoubleCoreIndexChanged(EventArgs e);
```

Parameters

Parameters	Description
EventArgs e	An object that contains no event data.

7.2.3.1.2.2.3 NFView.OnSelectedDoubleCoreIndexChanged Method

Occurs when index of the selected double core has been changed.

C#

```
public virtual void OnSelectedDoubleCoreIndexChanged(EventArgs e);
```

Parameters

Parameters	Description
EventArgs e	An object that contains no event data.

7.2.3.1.2.3 NFView Properties

7.2.3.1.2.3.1 NFView.SelectedCoreIndex Property

Gets or sets the index of the selected core.

C#

```
[Browsable(false), DesignerSerializationVisibility(DesignerSerializationVisibility.Hidden)]
public override int SelectedCoreIndex;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

The index of the selected core.

7.2.3.1.2.3.2 NFView.SelectedDeltaIndex Property

Gets or sets the index of the selected delta.

C#

```
[Browsable(false), DesignerSerializationVisibility(DesignerSerializationVisibility.Hidden)]
public override int SelectedDeltaIndex;
```

Property value

The index of the selected delta.

7.2.3.1.2.3.3 NFView.SelectedDoubleCoreIndex Property

Gets or sets the index of the selected double core.

C#

```
[Browsable(false), DesignerSerializationVisibility(DesignerSerializationVisibility.Hidden)]
public int SelectedDoubleCoreIndex;
```

7.2.3.1.2.3.4 NFView.SelectedMinutiaIndex Property

Gets or sets the index of the selected minutia.

C#

```
[Browsable(false), DesignerSerializationVisibility(DesignerSerializationVisibility.Hidden)]
public override int SelectedMinutiaIndex;
```

Property value

The index of the selected minutia.

7.2.3.1.2.3.5 NFView.Template Property

Gets or sets the template.

C#

```
[Category("Behavior"), DefaultValue((string)null)]
public new NFRecord Template;
```

Property value

NFRecord object.

7.2.3.1.2.4 NFView Events

7.2.3.1.2.4.1 NFView.HoveredDoubleCoreIndexChanged Event

Handler for event which occurs when index of the hovered double core has been changed.

C#

```
public event EventHandler HoveredDoubleCoreIndexChanged;
```

7.2.3.1.2.4.2 NFView.SelectedDoubleCoreIndexChanged Event

Handler for event which occurs when index of the selected double core has been changed.

C#

```
public event EventHandler SelectedDoubleCoreIndexChanged;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.3.1.3 NLView Class

Provides functionality for managing face pictures.

C#

```
public class NLView : Neurotec.Gui.NView;
```

Methods

	Name	Description
	NLView [1787]	Initializes a new instance of the NLView class.

NLView Properties

	Name	Description
	BaseFeaturePointSize [1787]	Gets or sets the size of the feature points.
	DetectionDetails [1788]	Gets or sets the detection details.
	DrawConfidenceForEyes [1788]	Gets or sets draw confidence for the eyes.
	DrawFaceConfidence [1788]	Gets or sets draw confidence for the face.
	FaceIds [1788]	Gets or sets the face IDs.
	FaceRectangleColor [1788]	Gets or sets the color of the face-bounding rectangle.
	FaceRectangleWidth [1789]	Gets or sets the width of the face-bounding rectangle.
	FeaturePointSize [1789]	Gets or sets the size of the feature points.
	Image [1789]	Gets or sets the current image.
	ShowBaseFeaturePoints [1789]	Checks or sets whether base feature points are shown.
	ShowExpression [1789]	Checks or sets whether the expression is shown.
	ShowExpressionConfidence [1790]	Gets or sets show confidence for the expression.
	ShowEyes [1790]	Checks or sets whether the eyes are shown.
	ShowEyesConfidence [1790]	Gets or sets show confidence for the eyes.
	ShowFaceConfidence [1790]	Gets or sets show confidence for the face.
	ShowGender [1790]	Checks or sets whether the gender is shown.
	ShowGenderConfidence [1790]	Gets or sets show confidence for the gender.
	ShowMouth [1791]	Checks or sets whether the mouth is shown.
	ShowMouthConfidence [1791]	Gets or sets show confidence for the mouth.
	ShowNose [1791]	Checks or sets whether the nose is shown.
	ShowNoseConfidence [1791]	Gets or sets show confidence for the nose.
	ShowProperties [1791]	Checks or sets whether the properties(Glasses, Blink, etc) is shown.
	ShowPropertiesConfidence [1792]	Gets or sets show confidence for the properties.

7.2.3.1.3.1 NLView.NLView Constructor

Initializes a new instance of the NLView class.

C#

```
public NLView();
```

7.2.3.1.3.2 NLView Properties

7.2.3.1.3.2.1 NLView.BaseFeaturePointSize Property

Gets or sets the size of the feature points.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
[DefaultValue(2)]
public int BaseFeaturePointSize;
```

Property value

The size of the feature points.

7.2.3.1.3.2.2 NLView.DetectionDetails Property

Gets or sets the detection details.

C#

```
[Browsable(false), Description("Detection details for multiple faces.")]
public NleDetectionDetails DetectionDetails;
```

Property value

The detection details.

7.2.3.1.3.2.3 NLView.DrawConfidenceForEyes Property

Gets or sets draw confidence for the eyes.

C#

```
[Obsolete("Use ShowEyesConfidence instead.")]
[DefaultValue(false)]
public bool DrawConfidenceForEyes;
```

Property value

true if there is draw confidence for the eyes; else **false**.

7.2.3.1.3.2.4 NLView.DrawFaceConfidence Property

Gets or sets draw confidence for the face.

C#

```
[Obsolete("Use ShowFaceConfidence instead.")]
[DefaultValue(true)]
public bool DrawFaceConfidence;
```

Property value

true if there is draw confidence for the face; else **false**.

7.2.3.1.3.2.5 NLView.FaceIds Property

Gets or sets the face IDs.

C#

```
[Browsable(false), Description("Arbitrary strings that would be displayed near face.")]
public string FaceIds;
```

Property value

An array of strings containing face IDs.

7.2.3.1.3.2.6 NLView.FaceRectangleColor Property

Gets or sets the color of the face-bounding rectangle.

C#

```
[DefaultValue(typeof(Color), "Green")]
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
public Color FaceRectangleColor;
```

Property value

The color of the face-bounding rectangle.

7.2.3.1.3.2.7 NLView.FaceRectangleWidth Property

Gets or sets the width of the face-bounding rectangle.

C#

```
[DefaultValue(2)]
public int FaceRectangleWidth;
```

Property value

The width of the face-bounding rectangle.

7.2.3.1.3.2.8 NLView.FeaturePointSize Property

Gets or sets the size of the feature points.

C#

```
[DefaultValue(4)]
public int FeaturePointSize;
```

Property value

The size of the feature points.

7.2.3.1.3.2.9 NLView.Image Property

Gets or sets the current image.

C#

```
public Bitmap Image;
```

Property value

The current image.

7.2.3.1.3.2.10 NLView.ShowBaseFeaturePoints Property

Checks or sets whether base feature points are shown.

C#

```
[DefaultValue(true)]
public bool ShowBaseFeaturePoints;
```

Property value

true if base feature points are shown; else **false**.

7.2.3.1.3.2.11 NLView.ShowExpression Property

Checks or sets whether the expression is shown.

C#

```
[DefaultValue(true)]
public bool ShowExpression;
```

Property value

true if the expression is shown; else **false**.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.3.1.3.2.12 NLView.ShowExpressionConfidence Property

Gets or sets show confidence for the expression.

C#

```
[DefaultValue(false)]
public bool ShowExpressionConfidence;
```

Property value

true if there is show confidence for the expression; else **false**.

7.2.3.1.3.2.13 NLView.ShowEyes Property

Checks or sets whether the eyes are shown.

C#

```
[DefaultValue(true)]
public bool ShowEyes;
```

Property value

true if the eyes are shown; else **false**.

7.2.3.1.3.2.14 NLView.ShowEyesConfidence Property

Gets or sets show confidence for the eyes.

C#

```
[DefaultValue(false)]
public bool ShowEyesConfidence;
```

Property value

true if there is show confidence for the eyes; else **false**.

7.2.3.1.3.2.15 NLView.ShowFaceConfidence Property

Gets or sets show confidence for the face.

C#

```
[DefaultValue(true)]
public bool ShowFaceConfidence;
```

Property value

true if there is show confidence for the face; else **false**.

7.2.3.1.3.2.16 NLView.ShowGender Property

Checks or sets whether the gender is shown.

C#

```
[DefaultValue(true)]
public bool ShowGender;
```

Property value

true if the gender is shown; else **false**.

7.2.3.1.3.2.17 NLView.ShowGenderConfidence Property

Gets or sets show confidence for the gender.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
[DefaultValue(false)]
public bool ShowGenderConfidence;
```

Property value

true if there is show confidence for the gender; else **false**.

7.2.3.1.3.2.18 NLView.ShowMouth Property

Checks or sets whether the mouth is shown.

C#

```
[DefaultValue(true)]
public bool ShowMouth;
```

Property value

true if the mouth is shown; else **false**.

7.2.3.1.3.2.19 NLView.ShowMouthConfidence Property

Gets or sets show confidence for the mouth.

C#

```
[DefaultValue(false)]
public bool ShowMouthConfidence;
```

Property value

true if there is show confidence for the mouth; else **false**.

7.2.3.1.3.2.20 NLView.ShowNose Property

Checks or sets whether the nose is shown.

C#

```
[DefaultValue(true)]
public bool ShowNose;
```

Property value

true if the nose is shown; else **false**.

7.2.3.1.3.2.21 NLView.ShowNoseConfidence Property

Gets or sets show confidence for the nose.

C#

```
[DefaultValue(false)]
public bool ShowNoseConfidence;
```

Property value

true if there is show confidence for the nose; else **false**.

7.2.3.1.3.2.22 NLView.ShowProperties Property

Checks or sets whether the properties(Glasses, Blink, etc) is shown.

C#

```
[DefaultValue(true)]
public bool ShowProperties;
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

true if the properties is shown; else **false**.

7.2.3.1.3.2.23 NLView.ShowPropertiesConfidence Property

Gets or sets show confidence for the properties.

C#

```
[DefaultValue(false)]
public bool ShowPropertiesConfidence;
```

Property value

true if there is show confidence for the properties; else **false**.

7.2.4 Neurotec.Devices Namespace

Provides functionality for managing devices, like cameras, fingerprint or irises scanners.

Classes

	Name	Description
	NBiometricDevice [1792]	Provides methods to retrieve information about biometric device.
	NCamera [1797]	Provides methods for working with cameras.
	NCameraStillCapturedEventArgs [1803]	Contains NCamera [1797] still image capture event arguments.
	NCaptureDevice [1804]	Provides methods for working with capture devices.
	NDevice [1809]	Provides functionality for retrieving device information.
	NDeviceManager [1814]	Provides functionality for managing devices.
	NDeviceManagerDeviceEventArgs [1822]	Contains device manager device event arguements.
S	NDevices [1822]	Provides functionality for retrieving information about NDevices library.

Module

.NET Reference [1468]

Structs, Records, Enums

	Name	Description
	NCameraStatus [1823]	Enumerates camera status values.
	NDeviceType [1824]	Enumerates device type values.

7.2.4.1 Classes

7.2.4.1.1 NBiometricDevice Class

Provides methods to retrieve information about biometric device.

C#

```
public class NBiometricDevice : NDevice;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDevice Classes

NDevice Class

	Name	Description
	ChildCollection [1811]	Children [1812] collection of specified multimodal device.

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	FromObject [1554]	Creates a new NObject [1549] from the specified object.
	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NBiometricDevice Class

	Name	Description
	Cancel [1796]	Stops capturing and cancels NBiometricDevice (NFScanner or NIrisScanner)
	EndSequence [1796]	Ends multiple images capturing sequence.
	StartSequence [1796]	Starts sequence which prepares device for multiple capture operations.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NDevice Class

	Name	Description
	Children [1812]	Gets children collections of specified multimodal device.
	DeviceType [1812]	Gets the type of multimodal device.
	DisplayName [1812]	Gets device name
	Id [1812]	Gets device Id.
	IsAvailable [1812]	Checks if the device is available.
	IsDisconnectable [1812]	Checks whether the device is disconnectable.
	IsPrivate [1813]	Checks if the device is private.
	Make [1813]	Gets device manufacturer.
	Model [1813]	Gets model of current biometric device.
S	NativeType [1813]	Defines the native type of the object.
S	NDeviceTypeNativeType [1813]	Defines the native type of the device type.
	Owner [1813]	Gets the owner of the device manager.
	Parent [1813]	Gets the parent of specified multimodal device module.
	Plugin [1814]	Gets device plugin.
	SerialNumber [1814]	Gets serial number of current biometric device.

NBiometricDevice Class

	Name	Description
	BiometricType [1796]	Gets the type of biometric device.
	IsSpoofDetectionSupported [1796]	Checks if the device is capable to detect spoof objects.
S	NativeType [1797]	Defines the native type of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	ProductId [1797]	Gets product device Id [1812].
	SpoofDetection [1797]	Gets or sets spoof detection value for biometric device.
	VendorId [1797]	Gets device vendor Id [1812].

7.2.4.1.1.1 NBiometricDevice Methods

7.2.4.1.1.1.1 NBiometricDevice.Cancel Method

Stops capturing and cancels NBiometricDevice [1792] (NFScanner or NIrisScanner)

C#

```
public void Cancel();
```

Remarks

Also EndSequence [1796] method can be called which cancels device and stops sequence.

7.2.4.1.1.1.2 NBiometricDevice.EndSequence Method

Ends multiple images capturing sequence.

C#

```
public void EndSequence();
```

7.2.4.1.1.1.3 NBiometricDevice.StartSequence Method

Starts sequence which prepares device for multiple capture operations.

C#

```
public void StartSequence();
```

Remarks

Some devices do not support multiple capturing (sequences). When called for such devices, this method does nothing.

StartSequence method does not start capturing. Capturing should be started using Capture methods. This method should be called for each fingerprint/palmprint or iris image.

If capturing sequence was started, after capturing it should be ended by calling EndSequence [1796] method.

7.2.4.1.1.2 NBiometricDevice Properties

7.2.4.1.1.2.1 NBiometricDevice.BiometricType Property

Gets the type of biometric device.

C#

```
public NBiometricType BiometricType;
```

Property value

Device biometric type.

7.2.4.1.1.2.2 NBiometricDevice.IsSpoofDetectionSupported Property

Checks if the device is capable to detect spoof objects.

C#

```
public bool IsSpoofDetectionSupported;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

true if device is capable to detect spoof objects; otherwise false.

7.2.4.1.1.2.3 NBiometricDevice.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.4.1.1.2.4 NBiometricDevice.ProductId Property

Gets product device Id [1812].

C#

```
public ushort ProductId;
```

Property value

Product Id [1812].

7.2.4.1.1.2.5 NBiometricDevice.SpoofDetection Property

Gets or sets spoof detection value for biometric device.

C#

```
public bool SpoofDetection;
```

Property value

true if device is used to detect spoof objects; otherwise false.

7.2.4.1.1.2.6 NBiometricDevice.VendorId Property

Gets device vendor Id [1812].

C#

```
public ushort VendorId;
```

Property value

Device vendor Id [1812].

7.2.4.1.2 NCamera Class

Provides methods for working with cameras.

C#

```
public sealed class NCamera : NCaptureDevice;
```

NDevice Classes

NDevice Class

	Name	Description
	ChildCollection [1811]	Children [1812] collection of specified multimodal device.

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NCaptureDevice Class

	Name	Description
	IsCapturingChanged [1808]	Gets or sets

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	FromObject [1554]	Creates a new NObject [1549] from the specified object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.
S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.
S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCaptureDevice Class

	Name	Description
	GetCurrentFormat [1807]	Gets current media format of the capture device.
	GetFormats [1807]	Gets all supported formats by the capture device.
	SetCurrentFormat [1807]	Sets current media format for the capture device.
	StartCapturing [1808]	Starts capturing.
	StopCapturing [1808]	Stops capturing.

NCamera Class

	Name	Description
	CaptureStill [1801]	Gets a still image from capture device (camera).
	Focus [1801]	Adjusts camera focus.
	GetCurrentStillFormat [1801]	Gets the current still image format.
	GetFrame [1801]	Gets current frame from the capture device.
	GetStillFormats [1802]	Gets still image formats supported by specified camera.
	ResetFocus [1802]	Resets camera focus.
	SetCurrentStillFormat [1802]	Sets the current still image format.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NDevice Class

	Name	Description
	Children [1812]	Gets children collections of specified multimodal device.
	DeviceType [1812]	Gets the type of multimodal device.
	DisplayName [1812]	Gets device name
	Id [1812]	Gets device Id.
	IsAvailable [1812]	Checks if the device is available.
	IsDisconnectable [1812]	Checks whether the device is disconnectable.
	IsPrivate [1813]	Checks if the device is private.
	Make [1813]	Gets device manufacturer.
	Model [1813]	Gets model of current biometric device.
S	NativeType [1813]	Defines the native type of the object.
S	NDeviceTypeNativeType [1813]	Defines the native type of the device type.
	Owner [1813]	Gets the owner of the device manager.
	Parent [1813]	Gets the parent of specified multimodal device module.
	Plugin [1814]	Gets device plugin.
	SerialNumber [1814]	Gets serial number of current biometric device.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCaptureDevice Class

	Name	Description
	IsCapturing [1808]	Checks if the device is capturing.
	MediaType [1808]	Gets the media type.
S	NativeType [1808]	Defines the native type of the object.

NCamera Class

	Name	Description
	FocusRegion [1802]	Gets or sets the focus region.
	IsFocusRegionSupported [1802]	Checks whether the focus region is supported.
	IsFocusSupported [1802]	Checks whether the focus is supported.
	IsStillCaptureSupported [1803]	Checks whether still capture is supported.
S	NativeType [1803]	Defines the native type of the object.
S	NCameraStatusNativeType [1803]	Defines the native type of the camera status.

7.2.4.1.2.1 NCamera Methods

7.2.4.1.2.1.1 NCamera.CaptureStill Method

Gets a still image from capture device (camera).

C#

```
public NCameraStatus CaptureStill();
```

Returns

NCameraStatus [1823] object.

Remarks

Captured still image (NImage [1232]) is retrieved by calling NCameraAddStillCapturedCallback [1147] (user defined function).

7.2.4.1.2.1.2 NCamera.Focus Method

Adjusts camera focus.

C#

```
public NCameraStatus Focus();
```

Returns

NCameraStatus [1823] object.

7.2.4.1.2.1.3 NCamera.GetCurrentStillFormat Method

Gets the current still image format.

C#

```
public NVideoFormat GetCurrentStillFormat();
```

Returns

One of NVideoFormat [1395] values.

7.2.4.1.2.1.4 NCamera.GetFrame Method

Gets current frame from the capture device.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NImage GetFrame();
```

Returns

Frame image.

7.2.4.1.2.1.5 NCamera.GetStillFormats Method

Gets still image formats supported by specified camera.

C#

```
public NVideoFormat[] GetStillFormats();
```

Returns

The allocated array of NVideoFormat [1395] structures describing all supported formats for the current capture device (camera).

7.2.4.1.2.1.6 NCamera.ResetFocus Method

Resets camera focus.

C#

```
public void ResetFocus();
```

7.2.4.1.2.1.7 NCamera.SetCurrentStillFormat Method

Sets the current still image format.

C#

```
public void SetCurrentStillFormat(NVideoFormat format);
```

Parameters

Parameters	Description
NVideoFormat format	One of NVideoFormat [1395] values.

7.2.4.1.2.2 NCamera Properties

7.2.4.1.2.2.1 NCamera.FocusRegion Property

Gets or sets the focus region.

C#

```
public Nullable<RectangleF> FocusRegion;
```

7.2.4.1.2.2.2 NCamera.IsFocusRegionSupported Property

Checks whether the focus region is supported.

C#

```
public bool IsFocusRegionSupported;
```

7.2.4.1.2.2.3 NCamera.IsFocusSupported Property

Checks whether the focus is supported.

C#

```
public bool IsFocusSupported;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.4.1.2.2.4 NCamera.IsStillCaptureSupported Property

Checks whether still capture is supported.

C#

```
public bool IsStillCaptureSupported;
```

7.2.4.1.2.2.5 NCamera.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.4.1.2.2.6 NCamera.NCameraStatusNativeType Property

Defines the native type of the camera status.

C#

```
public static NType NCameraStatusNativeType;
```

7.2.4.1.3 NCameraStillCapturedEventArgs Class

Contains NCamera [1797] still image capture event arguments.

C#

```
public class NCameraStillCapturedEventArgs : EventArgs;
```

Methods

	Name	Description
	NCameraStillCapturedEventArgs [1803]	Creates a new instance of NCameraStillCapturedEventArgs class.

NCameraStillCapturedEventArgs Properties

	Name	Description
	Id [1804]	Retrieves the camera ID.
	Properties [1804]	Retrieves the event properties.
	Stream [1804]	Retrieves the data stream.

7.2.4.1.3.1 NCameraStillCapturedEventArgs.NCameraStillCapturedEventArgs Constructor

Creates a new instance of NCameraStillCapturedEventArgs class.

C#

```
public NCameraStillCapturedEventArgs(Stream stream, object id, NPropertyBag properties);
```

Parameters

Parameters	Description
Stream stream	Data stream.
object id	Camera ID.
NPropertyBag properties	Event properties.

7.2.4.1.3.2 NCameraStillCapturedEventArgs Properties

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.4.1.3.2.1 NCameraStillCapturedEventArgs.Id Property

Retrieves the camera ID.

C#

```
public object Id;
```

7.2.4.1.3.2.2 NCameraStillCapturedEventArgs.Properties Property

Retrieves the event properties.

C#

```
public NPropertyBag Properties;
```

7.2.4.1.3.2.3 NCameraStillCapturedEventArgs.Stream Property

Retrieves the data stream.

C#

```
public Stream Stream;
```

7.2.4.1.4 NCaptureDevice Class

Provides methods for working with capture devices.

C#

```
public class NCaptureDevice : NDevice;
```

NDevice Classes

NDevice Class

	Name	Description
	ChildCollection [1811]	Children [1812] collection of specified multimodal device.

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NCaptureDevice Class

	Name	Description
	IsCapturingChanged [1808]	Gets or sets

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.
S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NCaptureDevice Class

	Name	Description
	GetCurrentFormat [1807]	Gets current media format of the capture device.
	GetFormats [1807]	Gets all supported formats by the capture device.
	SetCurrentFormat [1807]	Sets current media format for the capture device.
	StartCapturing [1808]	Starts capturing.
	StopCapturing [1808]	Stops capturing.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NDevice Class

	Name	Description
	Children [1812]	Gets children collections of specified multimodal device.
	DeviceType [1812]	Gets the type of multimodal device.
	DisplayName [1812]	Gets device name
	Id [1812]	Gets device Id.
	IsAvailable [1812]	Checks if the device is available.
	IsDisconnectable [1812]	Checks whether the device is disconnectable.
	IsPrivate [1813]	Checks if the device is private.
	Make [1813]	Gets device manufacturer.
	Model [1813]	Gets model of current biometric device.
S	NativeType [1813]	Defines the native type of the object.
S	NDeviceTypeNativeType [1813]	Defines the native type of the device type.
	Owner [1813]	Gets the owner of the device manager.
	Parent [1813]	Gets the parent of specified multimodal device module.
	Plugin [1814]	Gets device plugin.
	SerialNumber [1814]	Gets serial number of current biometric device.

NCaptureDevice Class

	Name	Description
	IsCapturing [1808]	Checks if the device is capturing.
	MediaType [1808]	Gets the media type.
S	NativeType [1808]	Defines the native type of the object.

7.2.4.1.4.1 NCaptureDevice Methods

7.2.4.1.4.1.1 NCaptureDevice.GetCurrentFormat Method

Gets current media format of the capture device.

C#

```
public NMediaFormat GetCurrentFormat();
```

Returns

Current media format of capture device.

7.2.4.1.4.1.2 NCaptureDevice.GetFormats Method

Gets all supported formats by the capture device.

C#

```
public NMediaFormat[] GetFormats();
```

Returns

Array of supported media formats for the current capture device.

7.2.4.1.4.1.3 NCaptureDevice.SetCurrentFormat Method

Sets current media format for the capture device.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public void SetCurrentFormat(NMediaFormat format);
```

Parameters

Parameters	Description
NMediaFormat format	Current media format to set.

7.2.4.1.4.1.4 NCaptureDevice.StartCapturing Method

Starts capturing.

C#

```
public void StartCapturing();
```

7.2.4.1.4.1.5 NCaptureDevice.StopCapturing Method

Stops capturing.

C#

```
public void StopCapturing();
```

7.2.4.1.4.2 NCaptureDevice Properties

7.2.4.1.4.2.1 NCaptureDevice.IsCapturing Property

Checks if the device is capturing.

C#

```
public bool IsCapturing;
```

Property value

bool type value indicating if the device is capturing.

7.2.4.1.4.2.2 NCaptureDevice.MediaType Property

Gets the media type.

C#

```
public NMediaType MediaType;
```

7.2.4.1.4.2.3 NCaptureDevice.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.4.1.4.3 NCaptureDevice Events

7.2.4.1.4.3.1 NCaptureDevice.IsCapturingChanged Event

Gets or sets

C#

```
public event EventHandler<EventArgs> IsCapturingChanged;
```

Property value

Value

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.4.1.5 NDevice Class

Provides functionality for retrieving device information.

C#

```
public class NDevice : NObject;
```

NDevice Classes

NDevice Class

	Name	Description
	ChildCollection [1811]	Children [1812] collection of specified multimodal device.

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.
S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.
S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NDevice Class

	Name	Description
	Children [1812]	Gets children collections of specified multimodal device.
	DeviceType [1812]	Gets the type of multimodal device.
	DisplayName [1812]	Gets device name
	Id [1812]	Gets device Id.
	IsAvailable [1812]	Checks if the device is available.
	IsDisconnectable [1812]	Checks whether the device is disconnectable.
	IsPrivate [1813]	Checks if the device is private.
	Make [1813]	Gets device manufacturer.
	Model [1813]	Gets model of current biometric device.
S	NativeType [1813]	Defines the native type of the object.
S	NDeviceTypeNativeType [1813]	Defines the native type of the device type.
	Owner [1813]	Gets the owner of the device manager.
	Parent [1813]	Gets the parent of specified multimodal device module.
	Plugin [1814]	Gets device plugin.
	SerialNumber [1814]	Gets serial number of current biometric device.

7.2.4.1.5.1 NDevice Classes

7.2.4.1.5.1.1 NDevice.ChildCollection Class

Children [1812] collection of specified multimodal device.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public class ChildCollection : NObjectReadOnlyCollection<NDevice>;
```

7.2.4.1.5.2 NDevice Properties

7.2.4.1.5.2.1 NDevice.Children Property

Gets children collections of specified multimodal device.

C#

```
public ChildCollection Children;
```

Property value

Device children collection.

7.2.4.1.5.2.2 NDevice.DeviceType Property

Gets the type of multimodal device.

C#

```
public NDeviceType DeviceType;
```

Property value

Device type value.

7.2.4.1.5.2.3 NDevice.DisplayName Property

Gets device name

C#

```
public string DisplayName;
```

Property value

String containing device display name.

7.2.4.1.5.2.4 NDevice.Id Property

Gets device Id.

C#

```
public string Id;
```

Property value

String containing device Id.

7.2.4.1.5.2.5 NDevice.IsAvailable Property

Checks if the device is available.

C#

```
public bool IsAvailable;
```

Property value

Boolean value indicating if the device is available.

7.2.4.1.5.2.6 NDevice.IsDisconnectable Property

Checks whether the device is disconnectable.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public bool IsDisconnectable;
```

7.2.4.1.5.2.7 NDevice.IsPrivate Property

Checks if the device is private.

C#

```
public bool IsPrivate;
```

Property value

Boolean value indicating if the device is private.

7.2.4.1.5.2.8 NDevice.Make Property

Gets device manufacturer.

C#

```
public string Make;
```

Property value

String containing device manufacturer.

7.2.4.1.5.2.9 NDevice.Model Property

Gets model of current biometric device.

C#

```
public string Model;
```

Property value

String containing device model name.

7.2.4.1.5.2.10 NDevice.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.4.1.5.2.11 NDevice.NDeviceTypeNativeType Property

Defines the native type of the device type.

C#

```
public static NType NDeviceTypeNativeType;
```

7.2.4.1.5.2.12 NDevice.Owner Property

Gets the owner of the device manager.

C#

```
public new NDeviceManager Owner;
```

7.2.4.1.5.2.13 NDevice.Parent Property

Gets the parent of specified multimodal device module.

C#

```
public NDevice Parent;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

Parent of multimodal device.

7.2.4.1.5.2.14 NDevice.Plugin Property

Gets device plugin.

C#

```
public NPlugin Plugin;
```

Property value

Device plugin.

7.2.4.1.5.2.15 NDevice.SerialNumber Property

Gets serial number of current biometric device.

C#

```
public string SerialNumber;
```

Property value

String containing device serial number.

7.2.4.1.6 NDeviceManager Class

Provides functionality for managing devices.

C#

```
public sealed class NDeviceManager : NObject;
```

NDeviceManager Classes

NDeviceManager Class

	Name	Description
	DeviceCollection [1818]	Holds all devices data of the current NDeviceManager object.

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NDeviceManager Class

	Name	Description
	DeviceAdded [1821]	Represents the method to handle device added event.
	DeviceRemoved [1821]	Represents the method to handle device removed event.
	DevicesChanged [1821]	Represents the method to handle device changed event.
	DevicesChanging [1822]	Represents the method to handle device changing event.
	DevicesRefreshed [1822]	Represents the method to handle devices refreshed event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

Methods

NDeviceManager Class

	Name	Description
	NDeviceManager [1817]	Initializes a new instance of the NDeviceManager class.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.
S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.
S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDeviceManager Class

	Name	Description
	ConnectToDevice [1819]	Connects to specified device using NPlugin [1863] and NPropertyBag [1576] parameters.
	DisconnectFromDevice [1819]	Disconnects from specified device.
S	GetConnectToDeviceParameters [1820]	Gets connection parameters for a device from a specified plugin.
S	IsConnectToDeviceSupported [1820]	Checks whether the specified plugin supports connection to device.
	Refresh [1820]	Refreshes plugged devices list.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NDeviceManager Class

	Name	Description
	AutoUpdate [1820]	Checks whether NDeviceManager automatically updates devices list.
	Devices [1821]	Gets devices collection of the current NDeviceManager object.
	DeviceTypes [1821]	Gets supported device types by NDeviceManager.
S	NativeType [1821]	Defines the native type of the object.
S	PluginManager [1821]	Gets the plugin manager of the current object.

7.2.4.1.6.1 NDeviceManager Constructor

7.2.4.1.6.1.1 NDeviceManager.NDeviceManager Constructor ()

Initializes a new instance of the NDeviceManager class.

C#

```
public NDeviceManager();
```

7.2.4.1.6.1.2 NDeviceManager.NDeviceManager Constructor (NDeviceType)

Initializes a new instance of the NDeviceManager class.

C#

```
public NDeviceManager(NDeviceType deviceTypes);
```

Parameters

Parameters	Description
NDeviceType deviceTypes	Device types supported by NDeviceManager. If value Any is specified, new NDeviceManager will support all device types.

7.2.4.1.6.1.3 NDeviceManager.NDeviceManager Constructor (NDeviceType, bool)

Initializes a new instance of the NDeviceManager class.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NDeviceManager(NDeviceType deviceTypes, bool autoPlug);
```

Parameters

Parameters	Description
NDeviceType deviceTypes	Device types supported by NDeviceManager. If value Any is specified, new NDeviceManager will support all device types.
bool autoPlug	Bool value for automatic devices listing. If set to true, NDeviceManager automatically plugs-in devices and adds them to a list.

7.2.4.1.6.1.4 NDeviceManager.NDeviceManager Constructor (NDeviceType, bool, bool)

Initializes a new instance of the NDeviceManager class.

C#

```
public NDeviceManager(NDeviceType deviceTypes, bool autoPlug, bool autoUpdate);
```

Parameters

Parameters	Description
NDeviceType deviceTypes	Device types supported by NDeviceManager. If value Any is specified, new NDeviceManager will support all device types.
bool autoPlug	Bool value for automatic devices listing. If set to true, NDeviceManager automatically plugs-in devices and adds them to a list.
bool autoUpdate	Bool value for automatic devices update. If set to true, NDeviceManager will automatically update devices list when new device is added or removed.

7.2.4.1.6.2 NDeviceManager Classes

7.2.4.1.6.2.1 NDeviceManager.DeviceCollection Class

Holds all devices data of the current NDeviceManager object.

C#

```
public sealed class DeviceCollection : NObjectReadOnlyCollection<NDevice>;
```

DeviceCollection Methods

	Name	Description
	IndexOf [1818]	Retrieves device's index in a collection by specified device Id.
	this [1819]	Gets NDevice [1809] object specified by Id.

7.2.4.1.6.2.1.1 DeviceCollection Methods

7.2.4.1.6.2.1.1.1 NDeviceManager.DeviceCollection.IndexOf Method

Retrieves device's index in a collection by specified device Id.

C#

```
public int IndexOf(string id);
```

Parameters

Parameters	Description
string id	Device identifier.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Device index in the collection.

7.2.4.1.6.2.1.1.2 NDeviceManager.DeviceCollection.this Indexer

Gets NDevice [1809] object specified by Id.

C#

```
public NDevice this[string id];
```

Parameters

Parameters	Description
string id	Device Id.

Returns

NDevice [1809] object.

7.2.4.1.6.3 NDeviceManager Methods

7.2.4.1.6.3.1 ConnectToDevice Method

7.2.4.1.6.3.1.1 NDeviceManager.ConnectToDevice Method (NPlugin, NPropertyBag)

Connects to specified device using NPlugin [1863] and NPropertyBag [1576] parameters.

C#

```
public NDevice ConnectToDevice(NPlugin plugin, NPropertyBag parameters);
```

Parameters

Parameters	Description
NPlugin plugin	Device plugin used to establish a connection.
NPropertyBag parameters	Properties used for connection.

Returns

NDevice [1809] to which a connection was established.

7.2.4.1.6.3.1.2 NDeviceManager.ConnectToDevice Method (string, string)

Connects to a device using specified plugin and connection parameters.

C#

```
public NDevice ConnectToDevice(string plugin, string parameters);
```

Parameters

Parameters	Description
string plugin	String containing plugin name (e.g. "Media").
string parameters	String containing camera parameters (e.g. "FileName=C:video.avi;MediaType=nmtVideo").

Returns

NDevice [1809] to which a connection was established.

7.2.4.1.6.3.2 NDeviceManager.DisconnectFromDevice Method

Disconnects from specified device.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public void DisconnectFromDevice(NDevice device);
```

Parameters

Parameters	Description
NDevice device	The device to disconnect.

7.2.4.1.6.3.3 NDeviceManager.GetConnectToDeviceParameters Method

Gets connection parameters for a device from a specified plugin.

C#

```
public static NParameterDescriptor[] GetConnectToDeviceParameters(NPlugin plugin);
```

Parameters

Parameters	Description
NPlugin plugin	The device plugin.

Returns

An array of connection parameters supported by specified plugin.

7.2.4.1.6.3.4 NDeviceManager.IsConnectToDeviceSupported Method

Checks whether the specified plugin supports connection to device.

C#

```
public static bool IsConnectToDeviceSupported(NPlugin plugin);
```

Parameters

Parameters	Description
NPlugin plugin	The device plugin.

Returns

true if connection is supported; else false.

7.2.4.1.6.3.5 NDeviceManager.Refresh Method

Refreshes plugged devices list.

C#

```
public void Refresh();
```

Remarks

When NDeviceManager object was created using false value for autoUpdate parameter, user should call this method to refresh devices list. When autoUpdate was set to true, Refresh method is called automatically by NDeviceManager.

7.2.4.1.6.4 NDeviceManager Properties

7.2.4.1.6.4.1 NDeviceManager.AutoUpdate Property

Checks whether NDeviceManager automatically updates devices list.

C#

```
public bool AutoUpdate;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

true, if NDevice [1809] manager automatically updates devices list. otherwise false.

7.2.4.1.6.4.2 NDeviceManager.Devices Property

Gets devices collection of the current NDeviceManager object.

C#

```
public DeviceCollection Devices;
```

Property value

Device collection containing all devices of the current NDeviceManager.

7.2.4.1.6.4.3 NDeviceManager.DeviceTypes Property

Gets supported device types by NDeviceManager.

C#

```
public NDeviceType DeviceTypes;
```

Property value

Supported device types.

7.2.4.1.6.4.4 NDeviceManager.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.4.1.6.4.5 NDeviceManager.PluginManager Property

Gets the plugin manager of the current object.

C#

```
public static NPluginManager PluginManager;
```

7.2.4.1.6.5 NDeviceManager Events

7.2.4.1.6.5.1 NDeviceManager.DeviceAdded Event

Represents the method to handle device added event.

C#

```
public event EventHandler<NDeviceManagerDeviceEventArgs> DeviceAdded;
```

7.2.4.1.6.5.2 NDeviceManager.DeviceRemoved Event

Represents the method to handle device removed event.

C#

```
public event EventHandler<NDeviceManagerDeviceEventArgs> DeviceRemoved;
```

7.2.4.1.6.5.3 NDeviceManager.DevicesChanged Event

Represents the method to handle device changed event.

C#

```
public event EventHandler<EventArgs> DevicesChanged;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.4.1.6.5.4 NDeviceManager.DevicesChanging Event

Represents the method to handle device changing event.

C#

```
public event EventHandler<EventArgs> DevicesChanging;
```

7.2.4.1.6.5.5 NDeviceManager.DevicesRefreshed Event

Represents the method to handle devices refreshed event.

C#

```
public event EventHandler<EventArgs> DevicesRefreshed;
```

7.2.4.1.7 NDeviceManagerDeviceEventArgs Class

Contains device manager device event arguements.

C#

```
public class NDeviceManagerDeviceEventArgs : EventArgs;
```

Methods

	Name	Description
	NDeviceManagerDeviceEventArgs [1822]	Initializes a new instance of NDeviceManagerDeviceEventArgs class.

NDeviceManagerDeviceEventArgs Properties

	Name	Description
	Device [1822]	Gets the device.

7.2.4.1.7.1 NDeviceManagerDeviceEventArgs.NDeviceManagerDeviceEventArgs Constructor

Initializes a new instance of NDeviceManagerDeviceEventArgs class.

C#

```
public NDeviceManagerDeviceEventArgs(NDevice device);
```

Parameters

Parameters	Description
NDevice device	The device.

7.2.4.1.7.2 NDeviceManagerDeviceEventArgs Properties

7.2.4.1.7.2.1 NDeviceManagerDeviceEventArgs.Device Property

Gets the device.

C#

```
public NDevice Device;
```

7.2.4.1.8 NDevices Class

Provides functionality for retrieving information about NDevices library.

C#

```
public static class NDevices;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDevices Fields

	Name	Description
	DllName [1823]	NDevices library's Dll name.

NDevices Properties

	Name	Description
S	NativeModule [1823]	Defines the native module of the object.

7.2.4.1.8.1 NDevices Fields

7.2.4.1.8.1.1 NDevices.DllName Field

NDevices [1822] library's Dll name.

C#

```
public const string DllName = "NDevices";
```

7.2.4.1.8.2 NDevices Properties

7.2.4.1.8.2.1 NDevices.NativeModule Property

Defines the native module of the object.

C#

```
public static NModule NativeModule;
```

7.2.4.2 Structs, Records, Enums

7.2.4.2.1 Neurotec.Devices.NCameraStatus Enumeration

Enumerates camera status values.

C#

```
public enum NCameraStatus {
  None = 0,
  Ok = 1,
  AutoFocusFailure = 2,
  LensClosed = 3,
  MirrorUp = 4,
  SensorCleaning = 5,
  SilentOperation = 6,
  StroboCharge = 7,
  NoLens = 8
}
```

Members

Members	Description
None = 0	Reserved for future use.
Ok = 1	Indicates that camera is ready to use.
AutoFocusFailure = 2	Indicates that camera failed to auto focus an object.
LensClosed = 3	Indicates that camera lens are closed.
MirrorUp = 4	Indicates that camera mirror is up (when DSLR is used).
SensorCleaning = 5	Indicates that camera is performing sensor cleaning.
SilentOperation = 6	Indicates that camera is operating in silent mode.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

StroboCharge = 7	Indicates that camera strobe is charging.
NoLens = 8	Indicates that no lens were attached to the camera.

7.2.4.2.2 Neurotec.Devices.NDeviceType Enumeration

Enumerates device type values.

C#

```
[Flags]
public enum NDeviceType {
  None = 0,
  Camera = 0x01,
  BiometricDevice = 0x02,
  FScanner = 0x04,
  FingerScanner = 0x08,
  PalmScanner = 0x10,
  IrisScanner = 0x20,
  CaptureDevice = 0x40,
  Microphone = 0x80,
  Any = unchecked((int)0x80000000)
}
```

Members

Members	Description
None = 0	This value is used for primitive devices (e.g. for parent devices). NDeviceManager has such devices only if NDeviceManager was created using *Any* value for NDeviceType.
Camera = 0x01	Indicates that device type is camera.
BiometricDevice = 0x02	Indicates FingerScanner and IrisScanner values of this enumeration. For instance, if this value is returned by DeviceTypes property, it means that NDeviceManager type is one of the following: FingerScanner or IrisScanner.
FScanner = 0x04	Indicates that device is a scanner which can be used for friction ridges (fingers or palms) scanning. If FScanner value is passed to on of NDeviceManager constructors, it means that device which supports all friction ridge scanning devices will be created.
FingerScanner = 0x08	Indicates that device type is fingerprints scanner (FScanner scanner which can also scan fingerprints).
PalmScanner = 0x10	Indicates that device type is palmprints scanner (FScanner scanner which can also scan palmprints).
IrisScanner = 0x20	Indicates that device type is iris scanner.
CaptureDevice = 0x40	Indicates that device type is capture device.
Microphone = 0x80	Indicates that device type is microphone.
Any = unchecked((int)0x80000000)	This value indicates that NDeviceManager supports all available device types. For instance, when Any value is passed to one of NDeviceManager constructors, new NDeviceManager which supports all device modules (cameras, fingerprint, iris, palmprint scanners, etc.) is created. Also this value is returned when NDeviceManager does not recognize device's type (e.g. device's module version is newer than NDeviceManager). NDeviceManager type can be retrieved using DeviceTypes property.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.5 Neurotec.IO Namespace

Classes under this namespace provides infrastructure for Neurotechnology components.

Classes

	Name	Description
	NBuffer [1825]	Provides buffer management functionality.
	NMemoryStream [1837]	Provides access to unmanaged blocks of memory from managed code.
	NStream [1844]	This class supports internal Neurotechnology libraries infrastructure and should not be used directly in your code.

Module

.NET Reference [1468]

7.2.5.1 Classes

7.2.5.1.1 NBuffer Class

Provides buffer management functionality.

C#

```
public sealed class NBuffer : NObject;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

Methods

NBuffer Class

	Name	Description
	NBuffer [1828]	

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.
S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NBuffer Class

	Name	Description
S	Copy [1832]	Copies bytes from one buffer into another with specified offsets into both buffers and size.
	CopyFrom [1833]	Copies bytes from the specified buffer into the current buffer.
	CopyTo [1834]	Copies bytes from the current buffer into the buffer specified by pointer with specified size.
	ToArray [1836]	Copies the elements of the NBuffer to a new byte array.
	WriteTo [1836]	Writes the contents of NBuffer to memory stream.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NBuffer Class

	Name	Description
S	Empty [1837]	Gets or sets an empty NBuffer.
	LongSize [1837]	Gets a 64-bit integer that represents the length of NBuffer.
	NativeSize [1837]	Gets UIntPtr that represents the length of NBuffer.
S	NativeType [1837]	Defines the native type of the object.
	Ptr [1837]	Gets a representation of NBuffer pointer.
	Size [1837]	Gets the length of NBuffer.

7.2.5.1.1.1 NBuffer Constructor

7.2.5.1.1.1.1 NBuffer.NBuffer Constructor (IntPtr, UIntPtr)

C#

```
public NBuffer(IntPtr ptr, UIntPtr size);
```

Parameters

Parameters	Description
IntPtr ptr	_nt_
UIntPtr size	_nt_

7.2.5.1.1.1.2 NBuffer.NBuffer Constructor (IntPtr, UIntPtr, NMemoryType)

C#

```
public NBuffer(IntPtr ptr, UIntPtr size, NMemoryType ptrType);
```

Parameters

Parameters	Description
IntPtr ptr	_nt_
UIntPtr size	_nt_
NMemoryType ptrType	_nt_

7.2.5.1.1.1.3 NBuffer.NBuffer Constructor (IntPtr, UIntPtr, bool)

C#

```
public NBuffer(IntPtr ptr, UIntPtr size, bool ownsPtr);
```

Parameters

Parameters	Description
IntPtr ptr	_nt_
UIntPtr size	_nt_
bool ownsPtr	_nt_

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.5.1.1.1.4 NBuffer.NBuffer Constructor (IntPtr, int)

Initializes a new instance of the NBuffer class.

C#

```
public NBuffer(IntPtr ptr, int size);
```

Parameters

Parameters	Description
IntPtr ptr	A pointer to buffer.
int size	Size [1837] of a buffer.

7.2.5.1.1.1.5 NBuffer.NBuffer Constructor (IntPtr, int, NMemoryType)

C#

```
public NBuffer(IntPtr ptr, int size, NMemoryType ptrType);
```

Parameters

Parameters	Description
IntPtr ptr	_nt_
int size	_nt_
NMemoryType ptrType	_nt_

7.2.5.1.1.1.6 NBuffer.NBuffer Constructor (IntPtr, int, bool)

C#

```
public NBuffer(IntPtr ptr, int size, bool ownsPtr);
```

Parameters

Parameters	Description
IntPtr ptr	_nt_
int size	_nt_
bool ownsPtr	_nt_

7.2.5.1.1.1.7 NBuffer.NBuffer Constructor (IntPtr, long)

C#

```
public NBuffer(IntPtr ptr, long size);
```

Parameters

Parameters	Description
IntPtr ptr	_nt_
long size	_nt_

7.2.5.1.1.1.8 NBuffer.NBuffer Constructor (IntPtr, long, NMemoryType)

C#

```
public NBuffer(IntPtr ptr, long size, NMemoryType ptrType);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
IntPtr ptr	_nt_
long size	_nt_
NMemoryType ptrType	_nt_

7.2.5.1.1.1.9 NBuffer.NBuffer Constructor (IntPtr, long, bool)

C#

```
public NBuffer(IntPtr ptr, long size, bool ownsPtr);
```

Parameters

Parameters	Description
IntPtr ptr	_nt_
long size	_nt_
bool ownsPtr	_nt_

7.2.5.1.1.1.10 NBuffer.NBuffer Constructor (NBuffer, UIntPtr, UIntPtr)

C#

```
public NBuffer(NBuffer srcBuffer, UIntPtr offset, UIntPtr size);
```

Parameters

Parameters	Description
NBuffer srcBuffer	_nt_
UIntPtr offset	_nt_
UIntPtr size	_nt_

7.2.5.1.1.1.11 NBuffer.NBuffer Constructor (NBuffer, int, int)

C#

```
public NBuffer(NBuffer srcBuffer, int offset, int size);
```

Parameters

Parameters	Description
NBuffer srcBuffer	_nt_
int offset	_nt_
int size	_nt_

7.2.5.1.1.1.12 NBuffer.NBuffer Constructor (NBuffer, long, long)

C#

```
public NBuffer(NBuffer srcBuffer, long offset, long size);
```

Parameters

Parameters	Description
NBuffer srcBuffer	_nt_

Copyright © 2009-2014 Neurotechnology. All rights reserved.

long offset	_nt_
long size	_nt_

7.2.5.1.1.1.13 NBuffer.NBuffer Constructor (UIntPtr)

C#

```
public NBuffer(UIntPtr size);
```

Parameters

Parameters	Description
UIntPtr size	_nt_

7.2.5.1.1.1.14 NBuffer.NBuffer Constructor (byte[])

C#

```
public NBuffer(byte[] array);
```

Parameters

Parameters	Description
byte[] array	_nt_

7.2.5.1.1.1.15 NBuffer.NBuffer Constructor (byte[], int, int)

C#

```
public NBuffer(byte[] array, int arrayIndex, int arrayCount);
```

Parameters

Parameters	Description
byte[] array	_nt_
int arrayIndex	_nt_
int arrayCount	_nt_

7.2.5.1.1.1.16 NBuffer.NBuffer Constructor (int)

C#

```
public NBuffer(int size);
```

Parameters

Parameters	Description
int size	_nt_

7.2.5.1.1.1.17 NBuffer.NBuffer Constructor (long)

C#

```
public NBuffer(long size);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
long size	_nt_

7.2.5.1.1.2 NBuffer Methods

7.2.5.1.1.2.1 Copy Method

7.2.5.1.1.2.1.1 NBuffer.Copy Method (NBuffer, UIntPtr, NBuffer, UIntPtr, UIntPtr)

Copies bytes from one buffer into another with specified offsets into both buffers and size.

C#

```
public static void Copy(NBuffer srcBuffer, UIntPtr srcOffset, NBuffer dstBuffer, UIntPtr
dstOffset, UIntPtr size);
```

Parameters

Parameters	Description
NBuffer srcBuffer	The source buffer.
UIntPtr srcOffset	Offset into the source buffer.
NBuffer dstBuffer	The destination buffer.
UIntPtr dstOffset	Offset into the destination buffer.
UIntPtr size	The buffer size.

7.2.5.1.1.2.1.2 NBuffer.Copy Method (NBuffer, int, NBuffer, int, int)

Copies bytes from one buffer into another with specified offsets into both buffers and size.

C#

```
public static void Copy(NBuffer srcBuffer, int srcOffset, NBuffer dstBuffer, int dstOffset,
int size);
```

Parameters

Parameters	Description
NBuffer srcBuffer	The source buffer.
int srcOffset	Offset into the source buffer.
NBuffer dstBuffer	The destination buffer.
int dstOffset	Offset into the destination buffer.
int size	The buffer size.

7.2.5.1.1.2.1.3 NBuffer.Copy Method (NBuffer, long, NBuffer, long, long)

Copies bytes from one buffer into another with specified offsets into both buffers and size.

C#

```
public static void Copy(NBuffer srcBuffer, long srcOffset, NBuffer dstBuffer, long
dstOffset, long size);
```

Parameters

Parameters	Description
NBuffer srcBuffer	The source buffer.
long srcOffset	Offset into the source buffer.
NBuffer dstBuffer	The destination buffer.
long dstOffset	Offset into the destination buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

long size	The buffer size.

7.2.5.1.1.2.2 CopyFrom Method

7.2.5.1.1.2.2.1 NBuffer.CopyFrom Method (IntPtr, UIntPtr, UIntPtr)

Copies bytes from the specified buffer into the current buffer.

C#

```
public void CopyFrom(IntPtr pSrcBuffer, UIntPtr srcBufferSize, UIntPtr offset);
```

Parameters

Parameters	Description
IntPtr pSrcBuffer	Pointer to the source buffer.
UIntPtr srcBufferSize	Size [1837] of the source buffer.
UIntPtr offset	Offset into the current buffer.

7.2.5.1.1.2.2.2 NBuffer.CopyFrom Method (IntPtr, int, int)

Copies bytes from the specified buffer into the current buffer.

C#

```
public void CopyFrom(IntPtr pSrcBuffer, int srcBufferSize, int offset);
```

Parameters

Parameters	Description
IntPtr pSrcBuffer	Pointer to the source buffer.
int srcBufferSize	Size [1837] of the source buffer.
int offset	Offset into the current buffer.

7.2.5.1.1.2.2.3 NBuffer.CopyFrom Method (IntPtr, long, long)

Copies bytes from the specified buffer into the current buffer.

C#

```
public void CopyFrom(IntPtr pSrcBuffer, long srcBufferSize, long offset);
```

Parameters

Parameters	Description
IntPtr pSrcBuffer	Pointer to the source buffer.
long srcBufferSize	Size [1837] of the source buffer.
long offset	Offset into the current buffer.

7.2.5.1.1.2.2.4 NBuffer.CopyFrom Method (byte[], int, int, int)

Copies bytes from the specified byte array at specified index to the current buffer.

C#

```
public void CopyFrom(byte[] array, int arrayIndex, int arrayCount, int offset);
```

Parameters

Parameters	Description
byte[] array	The source byte array.
int arrayIndex	Index in the source array.
int arrayCount	Size [1837] of the source array.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

int offset	Offset into the current buffer.

7.2.5.1.1.2.3 CopyTo Method

7.2.5.1.1.2.3.1 NBuffer.CopyTo Method (IntPtr, UIntPtr)

Copies bytes from the current buffer into the buffer specified by pointer with specified size.

C#

```
public void CopyTo(IntPtr pDstBuffer, UIntPtr dstBufferSize);
```

Parameters

Parameters	Description
IntPtr pDstBuffer	Pointer to the destination buffer.
UIntPtr dstBufferSize	Size [1837] if the destination buffer.

7.2.5.1.1.2.3.2 NBuffer.CopyTo Method (IntPtr, int)

Copies bytes from the current buffer into the buffer specified by pointer with specified size.

C#

```
public void CopyTo(IntPtr pDstBuffer, int dstBufferSize);
```

Parameters

Parameters	Description
IntPtr pDstBuffer	Pointer to the destination buffer.
int dstBufferSize	Size [1837] if the destination buffer.

7.2.5.1.1.2.3.3 NBuffer.CopyTo Method (IntPtr, long)

Copies bytes from the current buffer into the buffer specified by pointer with specified size.

C#

```
public void CopyTo(IntPtr pDstBuffer, long dstBufferSize);
```

Parameters

Parameters	Description
IntPtr pDstBuffer	Pointer to the destination buffer.
long dstBufferSize	Size [1837] if the destination buffer.

7.2.5.1.1.2.3.4 NBuffer.CopyTo Method (NBuffer, UIntPtr)

Copies bytes from the current buffer into the specified buffer with specified offset.

C#

```
public void CopyTo(NBuffer dstBuffer, UIntPtr dstOffset);
```

Parameters

Parameters	Description
NBuffer dstBuffer	The destination buffer.
UIntPtr dstOffset	The destination offset.

7.2.5.1.1.2.3.5 NBuffer.CopyTo Method (NBuffer, int)

Copies bytes from the current buffer into the specified buffer with specified offset.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public void CopyTo(NBuffer dstBuffer, int dstOffset);
```

Parameters

Parameters	Description
NBuffer dstBuffer	The destination buffer.
int dstOffset	The destination offset.

7.2.5.1.1.2.3.6 NBuffer.CopyTo Method (NBuffer, long)

Copies bytes from the current buffer into the specified buffer with specified offset.

C#

```
public void CopyTo(NBuffer dstBuffer, long dstOffset);
```

Parameters

Parameters	Description
NBuffer dstBuffer	The destination buffer.
long dstOffset	The destination offset.

7.2.5.1.1.2.3.7 NBuffer.CopyTo Method (UIntPtr, IntPtr, UIntPtr, UIntPtr)

Copies bytes from the current buffer into a different buffer with specified size and offset.

C#

```
public void CopyTo(UIntPtr offset, IntPtr pDstBuffer, UIntPtr dstBufferSize, UIntPtr size);
```

Parameters

Parameters	Description
UIntPtr offset	Offset into the current buffer.
IntPtr pDstBuffer	The destination buffer.
UIntPtr dstBufferSize	Size [1837] of the destination buffer.
UIntPtr size	Size [1837] of the current buffer.

7.2.5.1.1.2.3.8 NBuffer.CopyTo Method (byte[], int)

Copies bytes from the current buffer into a byte array at specified index.

C#

```
public void CopyTo(byte[] array, int arrayIndex);
```

Parameters

Parameters	Description
byte[] array	The destination byte array.
int arrayIndex	The specified index.

7.2.5.1.1.2.3.9 NBuffer.CopyTo Method (int, IntPtr, int, int)

Copies bytes from the current buffer into a different buffer with specified size and offset.

C#

```
public void CopyTo(int offset, IntPtr pDstBuffer, int dstBufferSize, int size);
```

Parameters

Parameters	Description
int offset	Offset into the current buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

IntPtr pDstBuffer	The destination buffer.
int dstBufferSize	Size [1837] of the destination buffer.
int size	Size [1837] of the current buffer.

7.2.5.1.1.2.3.10 NBuffer.CopyTo Method (int, byte[], int, int)

Copies bytes from the current buffer into a byte array at specified index.

C#

```
public void CopyTo(int offset, byte[] array, int arrayIndex, int size);
```

Parameters

Parameters	Description
int offset	Offset into the current buffer.
byte[] array	The detination byte array.
int arrayIndex	The index in the array.
int size	Sized of the current buffer.

7.2.5.1.1.2.3.11 NBuffer.CopyTo Method (long, IntPtr, long, long)

Copies bytes from the current buffer into a different buffer with specified size and offset.

C#

```
public void CopyTo(long offset, IntPtr pDstBuffer, long dstBufferSize, long size);
```

Parameters

Parameters	Description
long offset	Offset into the current buffer.
IntPtr pDstBuffer	The destination buffer.
long dstBufferSize	Size [1837] of the destination buffer.
long size	Size [1837] of the current buffer.

7.2.5.1.1.2.4 NBuffer.ToArray Method

Copies the elements of the NBuffer [1825] to a new byte array.

C#

```
public byte[] ToArray();
```

Returns

A byte array containing copies of the elements of the NBuffer [1825].

7.2.5.1.1.2.5 NBuffer.WriteTo Method

Writes the contents of NBuffer [1825] to memory stream.

C#

```
public void WriteTo(Stream stream);
```

Parameters

Parameters	Description
Stream stream	The stream to write the contents of NBuffer [1825] to.

7.2.5.1.1.3 NBuffer Properties

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.5.1.1.3.1 NBuffer.Empty Property

Gets or sets an empty NBuffer [1825].

C#

```
public static NBuffer Empty;
```

Property value

Empty NBuffer [1825].

7.2.5.1.1.3.2 NBuffer.LongSize Property

Gets a 64-bit integer that represents the length of NBuffer [1825].

C#

```
public long LongSize;
```

7.2.5.1.1.3.3 NBuffer.NativeSize Property

Gets UIntPtr that represents the length of NBuffer [1825].

C#

```
public UIntPtr NativeSize;
```

7.2.5.1.1.3.4 NBuffer.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.5.1.1.3.5 NBuffer.Ptr Property

Gets a representation of NBuffer [1825] pointer.

C#

```
public IntPtr Ptr;
```

7.2.5.1.1.3.6 NBuffer.Size Property

Gets the length of NBuffer [1825].

C#

```
public int Size;
```

Property value

The length of NBuffer [1825].

7.2.5.1.2 NMemoryStream Class

Provides access to unmanaged blocks of memory from managed code.

C#

```
public sealed class NMemoryStream : NStream;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

Methods

NMemoryStream Class

	Name	Description
	NMemoryStream [1841]	Initializes a new, empty instance of the NMemoryStream class.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.

7

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	FromObject [1554]	Creates a new NObject [1549] from the specified object.
	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NStream Class

	Name	Description
	AsStream [1847]	Converts the current NStream [1844] into a data stream.
	CopyTo [1847]	Copies the current stream to another stream.
	Flush [1847]	Clears all buffers for this stream and causes any buffered data to be written to the underlying device.
S	FromStream [1847]	Creates new NStream [1844] from Stream object.
	Read [1848]	Reads a sequence of bytes from the current stream.
	ReadByte [1848]	Reads a byte from the stream.
	Seek [1848]	Sets the position within the current stream.
	SetLength [1849]	Sets the length of the current stream.
S	Synchronized [1849]	Creates a synchronized NStream [1844] from a Stream object.
	Write [1849]	Writes a content of a buffer to the current stream.
	WriteByte [1850]	Writes a byte to the current position in the stream.

NMemoryStream Class

	Name	Description
	GetBuffer [1843]	Retrieves the buffer.
	ToArray [1843]	Saves stream to byte array.
	WriteTo [1843]	Writes NMemoryStream to Stream.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NStream Class

	Name	Description
	CanRead [1850]	Gets a value indicating whether the current stream supports reading.
	CanSeek [1850]	Gets a value indicating whether the current stream supports seeking.
	CanWrite [1850]	Gets a value indicating whether the current stream supports writing.
	Length [1850]	Gets size of a stream in bytes.
S	NativeType [1850]	Defines the native type of the object.
S	Null [1850]	Gets an empty NStream [1844].
	Position [1851]	Gets number of bytes from the start of the data to the current position in a stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMemoryStream Class

	Name	Description
	Capacity [1843]	Gets the stream size of memory assigned to a stream (capacity).
	LongCapacity [1843]	Gets the stream size of memory assigned to a stream (capacity), as a long number.
	NativeCapacity [1843]	Gets the stream size of memory assigned to a stream (capacity), as an unsigned intiger.
S	NativeType [1843]	Defines the native type of the object.
	PositionPtr [1844]	Gets the pointer to a certain position.

7.2.5.1.2.1 NMemoryStream Constructor

7.2.5.1.2.1.1 NMemoryStream.NMemoryStream Constructor ()

Initializes a new, empty instance of the NMemoryStream class.

C#

```
public NMemoryStream();
```

7.2.5.1.2.1.2 NMemoryStream.NMemoryStream Constructor (NBuffer)

Initializes a new instance of the NMemoryStream class using the specified buffer.

C#

```
public NMemoryStream(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] specified.

7.2.5.1.2.1.3 NMemoryStream.NMemoryStream Constructor (NBuffer, FileAccess, bool)

Initializes a new instance of the NMemoryStream class using the specified buffer, file access and exposability.

C#

```
public NMemoryStream(NBuffer buffer, FileAccess access, bool bufferExposable);
```

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] specified.
FileAccess access	File access.
bool bufferExposable	true if the buffer is exposable; else false.

7.2.5.1.2.1.4 NMemoryStream.NMemoryStream Constructor (NBuffer, bool)

Initializes a new instance of the NMemoryStream class using the specified buffer and writing permission.

C#

```
public NMemoryStream(NBuffer buffer, bool canWrite);
```

Parameters

Parameters	Description
NBuffer buffer	NBuffer [1825] specified.
bool canWrite	true if writing is permitted; else false.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.5.1.2.1.5 NMemoryStream.NMemoryStream Constructor (UIntPtr)

Initializes a new instance of the NMemoryStream class using the specified memory length.

C#

```
public NMemoryStream(UIntPtr capacity);
```

Parameters

Parameters	Description
UIntPtr capacity	The length of the memory to use.

7.2.5.1.2.1.6 NMemoryStream.NMemoryStream Constructor (byte[])

Initializes a new instance of the NMemoryStream class using the specified buffer.

C#

```
public NMemoryStream(byte[] buffer);
```

Parameters

Parameters	Description
byte[] buffer	Byte buffer specified.

7.2.5.1.2.1.7 NMemoryStream.NMemoryStream Constructor (byte[], bool)

Initializes a new instance of the NMemoryStream class using the specified buffer and writing permission.

C#

```
public NMemoryStream(byte[] buffer, bool canWrite);
```

Parameters

Parameters	Description
byte[] buffer	Byte buffer specified.
bool canWrite	true if writing is permitted; else false.

7.2.5.1.2.1.8 NMemoryStream.NMemoryStream Constructor (int)

Initializes a new instance of the NMemoryStream class using the specified memory length.

C#

```
public NMemoryStream(int capacity);
```

Parameters

Parameters	Description
int capacity	The length of the memory to use.

7.2.5.1.2.1.9 NMemoryStream.NMemoryStream Constructor (long)

Initializes a new instance of the NMemoryStream class using the specified memory length.

C#

```
public NMemoryStream(long capacity);
```

Parameters

Parameters	Description
long capacity	The length of the memory to use.

7.2.5.1.2.2 NMemoryStream Methods

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.5.1.2.2.1 NMemoryStream.GetBuffer Method

Retrieves the buffer.

C#

```
public NBuffer GetBuffer();
```

Returns

NBuffer [1825] object.

7.2.5.1.2.2.2 NMemoryStream.ToArray Method

Saves stream to byte array.

C#

```
public byte[] ToArray();
```

Returns

Byte array.

7.2.5.1.2.2.3 NMemoryStream.WriteTo Method

Writes NMemoryStream [1837] to Stream.

C#

```
public void WriteTo(Stream stream);
```

Parameters

Parameters	Description
Stream stream	Stream object.

7.2.5.1.2.3 NMemoryStream Properties

7.2.5.1.2.3.1 NMemoryStream.Capacity Property

Gets the stream size of memory assigned to a stream (capacity).

C#

```
public int Capacity;
```

7.2.5.1.2.3.2 NMemoryStream.LongCapacity Property

Gets the stream size of memory assigned to a stream (capacity), as a long number.

C#

```
public long LongCapacity;
```

7.2.5.1.2.3.3 NMemoryStream.NativeCapacity Property

Gets the stream size of memory assigned to a stream (capacity), as an unsigned intiger.

C#

```
public UIntPtr NativeCapacity;
```

7.2.5.1.2.3.4 NMemoryStream.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.5.1.2.3.5 NMemoryStream.PositionPtr Property

Gets the pointer to a certain position.

C#

```
public IntPtr PositionPtr;
```

7.2.5.1.3 NStream Class

This class supports internal Neurotechnology libraries infrastructure and should not be used directly in your code.

C#

```
public class NStream : NObject;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.
S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.
S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NStream Class

	Name	Description
	AsStream [1847]	Converts the current NStream into a data stream.
	CopyTo [1847]	Copies the current stream to another stream.
	Flush [1847]	Clears all buffers for this stream and causes any buffered data to be written to the underlying device.
S	FromStream [1847]	Creates new NStream from Stream object.
	Read [1848]	Reads a sequence of bytes from the current stream.
	ReadByte [1848]	Reads a byte from the stream.
	Seek [1848]	Sets the position within the current stream.
	SetLength [1849]	Sets the length of the current stream.
S	Synchronized [1849]	Creates a synchronized NStream from a Stream object.
	Write [1849]	Writes a content of a buffer to the current stream.
	WriteByte [1850]	Writes a byte to the current position in the stream.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NStream Class

	Name	Description
	CanRead [1850]	Gets a value indicating whether the current stream supports reading.
	CanSeek [1850]	Gets a value indicating whether the current stream supports seeking.
	CanWrite [1850]	Gets a value indicating whether the current stream supports writing.
	Length [1850]	Gets size of a stream in bytes.
S	NativeType [1850]	Defines the native type of the object.
S	Null [1850]	Gets an empty NStream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	Position [1851]	Gets number of bytes from the start of the data to the current position in a stream.

7.2.5.1.3.1 NStream Methods

7.2.5.1.3.1.1 NStream.AsStream Method

Converts the current NStream [1844] into a data stream.

C#

```
public Stream AsStream();
```

Returns

The data stream.

7.2.5.1.3.1.2 CopyTo Method

7.2.5.1.3.1.2.1 NStream.CopyTo Method (NStream)

Copies the current stream to another stream.

C#

```
public void CopyTo(NStream dstStream);
```

Parameters

Parameters	Description
NStream dstStream	Destination stream.

7.2.5.1.3.1.2.2 NStream.CopyTo Method (NStream, int)

Copies the current stream to another stream with specified buffer size.

C#

```
public void CopyTo(NStream dstStream, int bufferSize);
```

Parameters

Parameters	Description
NStream dstStream	Destination stream.
int bufferSize	Buffer size.

7.2.5.1.3.1.3 NStream.Flush Method

Clears all buffers for this stream and causes any buffered data to be written to the underlying device.

C#

```
public void Flush();
```

7.2.5.1.3.1.4 NStream.FromStream Method

Creates new NStream [1844] from Stream object.

C#

```
public static NStream FromStream(Stream stream);
```

Parameters

Parameters	Description
Stream stream	Memory stream.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NStream [1844] object.

7.2.5.1.3.1.5 Read Method

7.2.5.1.3.1.5.1 NStream.Read Method (NBuffer)

Reads a sequence of bytes from the current stream.

C#

```
public int Read(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	Memory buffer.

Returns

Number of bytes successfully read.

7.2.5.1.3.1.5.2 NStream.Read Method (byte[], int, int)

Reads a sequence of bytes from the current stream.

C#

```
public int Read(byte[] buffer, int offset, int count);
```

Parameters

Parameters	Description
byte[] buffer	Destination array. The locations with index range from offset to (offset+count-1) will be replaced by the bytes read from the current source.
int offset	The zero-based byte offset in buffer at which to begin storing the data read from the current stream.
int count	Number of bytes to be read from the current stream.

Returns

Number of bytes successfully read.

7.2.5.1.3.1.6 NStream.ReadByte Method

Reads a byte from the stream.

C#

```
public int ReadByte();
```

Returns

Value of byte at current position in a stream.

7.2.5.1.3.1.7 NStream.Seek Method

Sets the position within the current stream.

C#

```
public void Seek(long offset, SeekOrigin origin);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
long offset	A byte offset relative to the origin parameter.
SeekOrigin origin	A value indicating the reference point used to obtain the new position.

Returns

The new position within the current stream.

7.2.5.1.3.1.8 NStream.SetLength Method

Sets the length of the current stream.

C#

```
public void SetLength(long value);
```

Parameters

Parameters	Description
long value	New desired length of the current stream in bytes.

7.2.5.1.3.1.9 NStream.Synchronized Method

Creates a synchronized NStream [1844] from a Stream object.

C#

```
public static NStream Synchronized(Stream stream);
```

Parameters

Parameters	Description
Stream stream	Memory stream.

Returns

NStream [1844] object.

7.2.5.1.3.1.10 Write Method

7.2.5.1.3.1.10.1 NStream.Write Method (NBuffer)

Writes a content of a buffer to the current stream.

C#

```
public void Write(NBuffer buffer);
```

Parameters

Parameters	Description
NBuffer buffer	Memory buffer.

7.2.5.1.3.1.10.2 NStream.Write Method (byte[], int, int)

Writes a sequence of bytes to the current stream.

C#

```
public void Write(byte[] buffer, int offset, int count);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
byte[] buffer	An array of bytes.
int offset	The zero-based byte offset in buffer at which to begin copying bytes to the current stream.
int count	Number of bytes to be written to the current stream.

7.2.5.1.3.1.11 NStream.WriteByte Method

Writes a byte to the current position in the stream.

C#

```
public void WriteByte(byte value);
```

Parameters

Parameters	Description
byte value	The byte to write to the stream.

7.2.5.1.3.2 NStream Properties

7.2.5.1.3.2.1 NStream.CanRead Property

Gets a value indicating whether the current stream supports reading.

C#

```
public bool CanRead;
```

7.2.5.1.3.2.2 NStream.CanSeek Property

Gets a value indicating whether the current stream supports seeking.

C#

```
public bool CanSeek;
```

7.2.5.1.3.2.3 NStream.CanWrite Property

Gets a value indicating whether the current stream supports writing.

C#

```
public bool CanWrite;
```

7.2.5.1.3.2.4 NStream.Length Property

Gets size of a stream in bytes.

C#

```
public long Length;
```

7.2.5.1.3.2.5 NStream.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.5.1.3.2.6 NStream.Null Property

Gets an empty NStream [1844].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public static NStream Null;
```

7.2.5.1.3.2.7 NStream.Position Property

Gets number of bytes from the start of the data to the current position in a stream.

C#

```
public long Position;
```

7.2.6 Neurotec.Licensing Namespace

Provides functionality for getting, releasing licenses.

Classes

	Name	Description
	NLicense [1851]	Provides functionality for getting, releasing licenses.
	NLicenseInfo [1858]	Provides information about NLicensing library.
	NLicensing [1860]	Provides methods for retrieving information about the NLicensing library.
	NLicensingService [1860]	Manages licensing services.

Module

.NET Reference [1468]

Structs, Records, Enums

	Name	Description
	NLicensingServiceStatus [1862]	Enumerates licensing service status values.

7.2.6.1 Classes

7.2.6.1.1 NLicense Class

Provides functionality for getting, releasing licenses.

C#

```
public static class NLicense;
```

NLicense Methods

	Name	Description
	ActivateOnline [1852]	Activates specified Id (that was obtained through calling GenerateId [1855] method or using ActivationWizardDotNet) and obtains license file for this Id via internet.
	DeactivateOnline [1852]	Deactivates specified license via internet.
	DeactivateOnlineWithId [1853]	Deactivates specified license via internet using specified deactivation Id.
	GenerateDeactivationId [1853]	Generates deactivation Id for specified provided product on current computer.
	GenerateDeactivationIdForLicense [1854]	Generates deactivation Id for specified license on current computer.
	GenerateId [1855]	Generates Id for activation from serial number for a particular machine.
	GetInfo [1855]	Retrieves license information of specified product.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	GetInfoForComponent [1856]	Retrieves license information of specified component.
	IsComponentActivated [1856]	Checks if component specified by name is activated.
	Obtain [1856]	Obtains license.
	ObtainComponents [1857]	Obtains component license from the specified server.
	Release [1858]	Releases license.
	ReleaseComponents [1858]	Releases component license.

NLicense Properties

	Name	Description
	NativeType [1858]	Defines the native type of the object.

7.2.6.1.1.1 NLicense Methods

7.2.6.1.1.1.1 NLicense.ActivateOnline Method

Activates specified Id (that was obtained through calling GenerateId [1855] method or using ActivationWizardDotNet) and obtains license file for this Id via internet.

C#

```
public static string ActivateOnline(string id);
```

Parameters

Parameters	Description
string id	Id generated from serial number.

Returns

String containing activated license data.

7.2.6.1.1.1.2 DeactivateOnline Method

7.2.6.1.1.1.2.1 NLicense.DeactivateOnline Method (string)

Deactivates specified license via internet.

C#

```
public static void DeactivateOnline(string license);
```

Parameters

Parameters	Description
string license	String that contains activated license's full product name OR license file content.

7.2.6.1.1.1.2.2 NLicense.DeactivateOnline Method (string, string)

Deactivates specified license via internet using specified Id generator.

C#

```
public static void DeactivateOnline(string idGenPath, string license);
```

Parameters

Parameters	Description
string idGenPath	Path to Id generator program
string license	Strin that contains activated license name.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.6.1.1.1.3 NLicense.DeactivateOnlineWithId Method

Deactivates specified license via internet using specified deactivation Id.

C#

```
public static void DeactivateOnlineWithId(string license, string deactivationId);
```

Parameters

Parameters	Description
string license	String that contains activated license name.
string deactivationId	Deactivation Id.

Remarks

Important! Both license and generated deactivation Id should be sent to support@neurotechnology.com in order to complete deactivation process. Deactivation on Neurotechnology side is done manually.

Deactivation must be performed on the same computer supplied license was generated for. If license is deactivated on another computer, although call to this function will succeed, but deactivation might be refused by support@neurotechnology.com.

If method succeeds - license will be deactivated on this computer and will not be usable.

7.2.6.1.1.1.4 GenerateDeactivationId Method

7.2.6.1.1.1.4.1 NLicense.GenerateDeactivationId Method (string)

Generates deactivation Id for specified provided product on current computer.

C#

```
public static string GenerateDeactivationId(string productName);
```

Parameters

Parameters	Description
string productName	Long product name (i.e. Face Extractor, Finger Matcher and etc.).

Returns

Generated deactivation Id.

Remarks

Important! Both license and generated deactivation Id should be sent to support@neurotechnology.com in order to complete deactivation process. Deactivation on Neurotechnology side is done manually.

Important! Deactivation must be performed on the same computer supplied license was generated for. If license is deactivated on another computer, although call to this function will succeed, but deactivation might be refused by support@neurotechnology.com.

Important! If method succeeds - license will be deactivated on this computer and will not be usable.

7.2.6.1.1.1.4.2 NLicense.GenerateDeactivationId Method (string, string)

Generates deactivation Id for specified provided product on current computer.

C#

```
public static string GenerateDeactivationId(string idGenPath, string productName);
```


Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
string idGenPath	Path to Id generator program.
string productName	Long product name (i.e. Face Extractor, Finger Matcher and etc.).

Returns

Generated deactivation Id.

Remarks

Important! Both license and generated deactivation Id should be sent to support@neurotechnology.com in order to complete deactivation process. Deactivation on Neurotechnology side is done manually.

Important! Deactivation must be performed on the same computer supplied license was generated for. If license is deactivated on another computer, although call to this function will succeed, but deactivation might be refused by support@neurotechnology.com.

Important! If method succeeds - license will be deactivated on this computer and will not be usable.

7.2.6.1.1.1.5 GenerateDeactivationIdForLicense Method

7.2.6.1.1.1.5.1 NLicense.GenerateDeactivationIdForLicense Method (string)

Generates deactivation Id for specified license on current computer.

C#

```
public static string GenerateDeactivationIdForLicense(string license);
```

Parameters

Parameters	Description
string license	License to be deactivated.

Returns

Generated deactivation Id.

Remarks

Important! Internet connection is required for this function to succeed.

Important! Both license and generated deactivation Id should be sent to support@neurotechnology.com in order to complete deactivation process. Deactivation on Neurotechnology side is done manually.

Important! Deactivation must be performed on the same computer supplied license was generated for. If license is deactivated on another computer, although call to this function will succeed, but deactivation might be refused by support@neurotechnology.com.

7.2.6.1.1.1.5.2 NLicense.GenerateDeactivationIdForLicense Method (string, string)

Generates deactivation Id for specified license on current computer.

C#

```
public static string GenerateDeactivationIdForLicense(string idGenPath, string license);
```

Parameters

Parameters	Description
string idGenPath	Path to Id generator program.
string license	License to be deactivated.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

Generated deactivation Id.

Remarks

Important! Internet connection is required for this function to succeed.

Important! Both license and generated deactivation Id should be sent to support@neurotechnology.com in order to complete deactivation process. Deactivation on Neurotechnology side is done manually.

Important! Deactivation must be performed on the same computer supplied license was generated for. If license is deactivated on another computer, although call to this function will succeed, but deactivation might be refused by support@neurotechnology.com.

7.2.6.1.1.1.6 GenerateId Method

7.2.6.1.1.1.6.1 NLicense.GenerateId Method (string)

Generates Id for activation from serial number for a particular machine.

C#

```
public static string GenerateId(string serialNumber);
```

Parameters

Parameters	Description
string serialNumber	Serial number used for generating Id.

Returns

Generated Id.

7.2.6.1.1.1.6.2 NLicense.GenerateId Method (string, string)

Generates Id for activation from serial number for a particular machine.

C#

```
public static string GenerateId(string idGenPath, string serialNumber);
```

Parameters

Parameters	Description
string idGenPath	Path to Id generator program.
string serialNumber	Serial number used for generating Id.

Returns

Generated Id.

7.2.6.1.1.1.7 NLicense.GetInfo Method

Retrieves license information of specified product.

C#

```
public static NLicenseInfo GetInfo(string product);
```

Parameters

Parameters	Description
string product	String that contains name of a product to retrieve license information.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Returns

NLicenseInfo [1858] object that contains properties for retrieving license information.

7.2.6.1.1.1.8 NLicense.GetInfoForComponent Method

Retrieves license information of specified component.

C#

```
public static NLicenseInfo GetInfoForComponent(string component);
```

Parameters

Parameters	Description
string component	String that contains name of a component to retrieve license information. See ObtainComponents method remarks for the list of supported component names for this parameter.

Returns

NLicenseInfo [1858] object that contains properties for retrieving license information.

7.2.6.1.1.1.9 NLicense.IsComponentActivated Method

Checks if component specified by name is activated.

C#

```
public static bool IsComponentActivated(string component);
```

Parameters

Parameters	Description
string component	String which contains component name. See ObtainComponents method remarks for the list of supported component names for this parameter.

Returns

Bool value indicating if the specified component was activated.

7.2.6.1.1.1.10 Obtain Method

7.2.6.1.1.1.10.1 NLicense.Obtain Method (string, int, string)

Obtains license.

C#

```
public static bool Obtain(string address, int port, string products);
```

Parameters

Parameters	Description
string address	Server address where license manager is installed as a server. "/local" is used for this parameter when license is tied with the computer using serial number.
int port	License manager server port.
string products	String which contains license name.

Returns

Returns true if license available; otherwise, false.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.6.1.1.1.10.2 NLicense.Obtain Method (string, string, string)

Obtains license.

C#

```
public static bool Obtain(string address, string port, string products);
```

Parameters

Parameters	Description
string address	Server address where license manager is installed as a server. "/local" is used for this parameter when license is tied with the computer using serial number.
string port	License manager server port.
string products	String which contains license name.

Returns

Returns true if license available; otherwise, false.

7.2.6.1.1.1.11 ObtainComponents Method

7.2.6.1.1.1.11.1 NLicense.ObtainComponents Method (string, int, string)

Obtains component license from the specified server.

C#

```
public static bool ObtainComponents(string address, int port, string components);
```

Parameters

Parameters	Description
string address	Server address where license manager is installed as a server. "/local" is used for this parameter when license is tied with the computer using serial number.
int port	License manager server port (integer value).
string components	String which contains component license name.

Returns

Returns true if component license available; otherwise, false.

7.2.6.1.1.1.11.2 NLicense.ObtainComponents Method (string, string, string)

Obtains component license from the specified server.

C#

```
public static bool ObtainComponents(string address, string port, string components);
```

Parameters

Parameters	Description
string address	Server address where license manager is installed as a server. "/local" is used for this parameter when license is tied with the computer using serial number.
string port	License manager server port (string value).
string components	String which contains component license name.

Returns

Returns true if component license available; otherwise, false.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.6.1.1.1.12 NLicense.Release Method

Releases license.

C#

```
public static void Release(string products);
```

Parameters

Parameters	Description
string products	Name of a product to release its license.

Returns

Returns true if license available; otherwise, false.

7.2.6.1.1.1.13 NLicense.ReleaseComponents Method

Releases component license.

C#

```
public static void ReleaseComponents(string components);
```

Parameters

Parameters	Description
string components	Name of a component to release its license. See ObtainComponents method remarks for the list of supported component names for this parameter.

Remarks

This method should be called as many times as method ObtainComponents [1857] was called.

7.2.6.1.1.2 NLicense Properties

7.2.6.1.1.2.1 NLicense.NativeType Property

Defines the native type of the object.

C#

```
public static NType NativeType;
```

7.2.6.1.2 NLicenseInfo Structure

Provides information about NLicensing library.

C#

```
[StructLayout(LayoutKind.Sequential)]
public struct NLicenseInfo {
}
```

Methods

	Name	Description
	NLicenseInfo [1859]	Creates a new instance of NLicenseInfo class.

NLicenseInfo Properties

	Name	Description
	DistributorId [1859]	Gets distributor id.
	IsObtained [1859]	Gets license status.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	NativeType [1859]	Defines the native type of the object.
	SerialNumber [1859]	Gets serial number of a license.

7.2.6.1.2.1 NLicenseInfo.NLicenseInfo Constructor

Creates a new instance of NLicenseInfo class.

C#

```
public NLicenseInfo(bool isObtained, int distributorId, int serialNumber);
```

Parameters

Parameters	Description
bool isObtained	*true* if the license is obtained, else *false*.
int distributorId	Distributor ID.
int serialNumber	Serial number.

7.2.6.1.2.2 NLicenseInfo Properties

7.2.6.1.2.2.1 NLicenseInfo.DistributorId Property

Gets distributor id.

C#

```
public int DistributorId;
```

Property value

Distributor id.

7.2.6.1.2.2.2 NLicenseInfo.IsObtained Property

Gets license status.

C#

```
public bool IsObtained;
```

Property value

True if license is obtained; otherwise, false.

7.2.6.1.2.2.3 NLicenseInfo.NativeType Property

Defines the native type of the object.

C#

```
public static NType NativeType;
```

7.2.6.1.2.2.4 NLicenseInfo.SerialNumber Property

Gets serial number of a license.

C#

```
public int SerialNumber;
```

Returns

Serial number.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.6.1.3 NLicensing Class

Provides methods for retrieving information about the NLicensing library.

C#

```
public static class NLicensing;
```

NLicensing Fields

	Name	Description
	DllName [1860]	Name of DLL containing unmanaged part of this class.

NLicensing Properties

	Name	Description
	NativeModule [1860]	Defines the native module of the object.

7.2.6.1.3.1 NLicensing Fields

7.2.6.1.3.1.1 NLicensing.DllName Field

Name of DLL containing unmanaged part of this class.

C#

```
public const string DllName = "NLicensing";
```

7.2.6.1.3.2 NLicensing Properties

7.2.6.1.3.2.1 NLicensing.NativeModule Property

Defines the native module of the object.

C#

```
public static NModule NativeModule;
```

7.2.6.1.4 NLicensingService Class

Manages licensing services.

C#

```
public class NLicensingService;
```

NLicensingService Methods

	Name	Description
	GetBinPath [1861]	Gets the binary path (path to pg.exe) to the license.
	GetConfPath [1861]	Gets path to licensing service configuration file (pgd.conf).
	GetStatus [1861]	Gets the status of the licensing service.
	Install [1861]	Installs licensing service.
	IsTrial [1862]	Checks whether the licensing service is trial.
	Start [1862]	Starts licensing service.
	Stop [1862]	Stops licensing service.
	Uninstall [1862]	Uninstalls licensing service.

NLicensingService Properties

	Name	Description
	NativeType [1862]	Defines the native type of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	NLicensingServiceStatusNativeType [1862]	Defines the native type of the licensing service status.

7.2.6.1.4.1 NLicensingService Methods

7.2.6.1.4.1.1 NLicensingService.GetBinPath Method

Gets the binary path (path to pg.exe) to the license.

C#

```
public static string GetBinPath();
```

Returns

Path to licensing service.

7.2.6.1.4.1.2 NLicensingService.GetConfPath Method

Gets path to licensing service configuration file (pgd.conf).

C#

```
public static string GetConfPath();
```

Returns

Path to licensing service configuration file.

7.2.6.1.4.1.3 NLicensingService.GetStatus Method

Gets the status of the licensing service.

C#

```
public static NLicensingServiceStatus GetStatus();
```

Returns

Licensing [1851] service status value.

7.2.6.1.4.1.4 Install Method

7.2.6.1.4.1.4.1 NLicensingService.Install Method (string)

Installs licensing service.

C#

```
public static void Install(string binPath);
```

Parameters

Parameters	Description
string binPath	Path to licensing service file (pg.exe).

7.2.6.1.4.1.4.2 NLicensingService.Install Method (string, string)

Installs licensing service.

C#

```
public static void Install(string binPath, string confPath);
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Parameters

Parameters	Description
string binPath	Path to licensing service file (pg.exe).
string confPath	Path to licensing service configuration file (pgd.conf).

7.2.6.1.4.1.5 NLicensingService.IsTrial Method

Checks whether the licensing service is trial.

C#

```
public static bool IsTrial();
```

Returns

true if the licensing service is trial; otherwise false.

7.2.6.1.4.1.6 NLicensingService.Start Method

Starts licensing service.

C#

```
public static void Start();
```

7.2.6.1.4.1.7 NLicensingService.Stop Method

Stops licensing service.

C#

```
public static void Stop();
```

7.2.6.1.4.1.8 NLicensingService.Uninstall Method

Uninstalls licensing service.

C#

```
public static void Uninstall();
```

7.2.6.1.4.2 NLicensingService Properties

7.2.6.1.4.2.1 NLicensingService.NativeType Property

Defines the native type of the object.

C#

```
public static NType NativeType;
```

7.2.6.1.4.2.2 NLicensingService.NLicensingServiceStatusNativeType Property

Defines the native type of the licensing service status.

C#

```
public static NType NLicensingServiceStatusNativeType;
```

7.2.6.2 Structs, Records, Enums

7.2.6.2.1 Neurotec.Licensing.NLicensingServiceStatus Enumeration

Enumerates licensing service status values.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public enum NLicensingServiceStatus {
  NotInstalled = 0,
  Stopped = 0x00000001,
  StartPending = 0x00000002,
  StopPending = 0x00000003,
  Running = 0x00000004,
  ContinuePending = 0x00000005,
  PausePending = 0x00000006,
  Paused = 0x00000007
}
```

Members

Members	Description
NotInstalled = 0	Indicates that licensing service is nor installed.
Stopped = 0x00000001	Indicates that licensing service is stopped.
StartPending = 0x00000002	Indicates that licensing service is pending to be started.
StopPending = 0x00000003	Indicates that licensing service is pending to be stopped.
Running = 0x00000004	Indicates that licensing service is running.
ContinuePending = 0x00000005	Indicates that licensing service was continued after pause.
PausePending = 0x00000006	Indicates that licensing service is pending to be paused.
Paused = 0x00000007	Indicates that licensing service is paused.

7.2.7 Neurotec.Plugins Namespace

Classes under this namespace provides plugins management framework for use in other Neurotechnology components.

Classes

	Name	Description
	NPlugin [1863]	Plug [1867]-in management framework for use in other Neurotechnology components.
	NPluginManager [1869]	Provides functions for working with plugins.
	NPluginManagerPluginEventArgs [1876]	Base class for holding NPluginManager [1869] events data.
	NPluginModule [1877]	The module providing one or more plugins. Provides methods to retrieve information about plugins.

Module

.NET Reference [1468]

Structs, Records, Enums

	Name	Description
	NPluginState [1882]	Enumerates NPlugin [1863] state values. The plugin state show how much of initialization the plugin has performed and can be used to identify common problems of plugin loading.

7.2.7.1 Classes

7.2.7.1.1 NPlugin Class

Plug [1867]-in management framework for use in other Neurotechnology components.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public sealed class NPlugin : NObject;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NPlugin Class

	Name	Description
	Changed [1869]	Adds or removes a Changed callback to/from NPlugin.
	Changing [1869]	Adds or removes a Changing callback to/from NPlugin.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.
S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.
S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NPlugin Class

	Name	Description
	Disable [1866]	Disables NPlugin
	Enable [1867]	Enables NPlugin.
	GetError [1867]	Returns the most recent exception of NPlugin.
	Plug [1867]	Plugs specified plug-in to the system.
	Unplug [1867]	Unplugs specified plug-in from the system.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NPlugin Class

	Name	Description
	FileName [1867]	Gets plug-in library name.
	LoadTime [1867]	Gets plug-in load time.
	Module [1868]	Gets NPluginModule [1877] object.
S	NativeType [1868]	Defines the native type of the object.
S	NPluginStateNativeType [1868]	Defines the native type of the plugin state.
	Owner [1868]	Gets the manager of the plugin.
	PlugTime [1868]	Gets plug-in plug time.
	SelectedInterfaceVersion [1868]	Gets current version of plugin interface.
	State [1868]	Gets the state of specified plugin.

7.2.7.1.1.1 NPlugin Methods

7.2.7.1.1.1.1 NPlugin.Disable Method

Disables NPlugin [1863]

C#

```
public void Disable();
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.7.1.1.1.2 NPlugin.Enable Method

Enables NPlugin [1863].

C#

```
public void Enable();
```

7.2.7.1.1.1.3 NPlugin.GetError Method

Returns the most recent exception of NPlugin [1863].

C#

```
public Exception GetError();
```

Returns

The most recent exception from NPlugin [1863].

7.2.7.1.1.1.4 NPlugin.Plug Method

Plugs specified plug-in to the system.

C#

```
public void Plug();
```

Remarks

The requested plugin might not be plugged into the system immediately. If there is no potential users (device managers) for provided device types, the plugin's state will be set to Unused [1882]. The plugin will be automatically plugged later, once device manager of appropriate type is created.

7.2.7.1.1.1.5 NPlugin.Unplug Method

Unplugs specified plug-in from the system.

C#

```
public void Unplug();
```

Remarks

- The function is not mandatory as plugin will be unplugged automatically when it becomes unused. Not calling a function would have effect that plugin would be later plugged automatically when needed.
- If the function is called the plugin's state will be set to Unplugged [1882] and plugin will not be automatically plugged later.
- The function can be used to force unplug even if devices provided by the plugin are still in use.

7.2.7.1.1.2 NPlugin Properties

7.2.7.1.1.2.1 NPlugin.FileName Property

Gets plug-in library name.

C#

```
public string FileName;
```

Property value

String which contains plug-in file name.

7.2.7.1.1.2.2 NPlugin.LoadTime Property

Gets plug-in load time.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public TimeSpan LoadTime;
```

Property value

TimeSpan structure with plugin load time.

7.2.7.1.1.2.3 NPlugin.Module Property

Gets NPluginModule [1877] object.

C#

```
public NPluginModule Module;
```

7.2.7.1.1.2.4 NPlugin.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.7.1.1.2.5 NPlugin.NPluginStateNativeType Property

Defines the native type of the plugin state.

C#

```
public static NType NPluginStateNativeType;
```

7.2.7.1.1.2.6 NPlugin.Owner Property

Gets the manager of the plugin.

C#

```
public new NPluginManager Owner;
```

Property value

Reference to NPluginManager [1869]

7.2.7.1.1.2.7 NPlugin.PlugTime Property

Gets plug-in plug time.

C#

```
public TimeSpan PlugTime;
```

Property value

TimeSpan structure with plugin load time.

7.2.7.1.1.2.8 NPlugin.SelectedInterfaceVersion Property

Gets current version of plugin interface.

C#

```
public NVersion SelectedInterfaceVersion;
```

Property value

NVersion [1640] value.

7.2.7.1.1.2.9 NPlugin.State Property

Gets the state of specified plugin.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NPluginState State;
```

Property value

NPluginState [1882] object.

7.2.7.1.1.3 NPlugin Events

7.2.7.1.1.3.1 NPlugin.Changed Event

Adds or removes a Changed callback to/from NPlugin [1863].

C#

```
public event EventHandler<EventArgs> Changed;
```

7.2.7.1.1.3.2 NPlugin.Changing Event

Adds or removes a Changing callback to/from NPlugin [1863].

C#

```
public event EventHandler<EventArgs> Changing;
```

7.2.7.1.2 NPluginManager Class

Provides functions for working with plugins.

C#

```
public sealed class NPluginManager : NObject;
```

NPluginManager Classes

NPluginManager Class

	Name	Description
	DisabledPluginCollection [1872]	Represents collection of plugin names for plugins that will be disabled.
	PluginCollection [1872]	Represents collection of plugins.

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NPluginManager Class

	Name	Description
	DisabledPluginsChanged [1876]	Adds arguments for "disabled plugins changed" event handler.
	DisabledPluginsChanging [1876]	Adds arguments for "disabled plugins changing" event handler.
	PluginAdded [1876]	Represents the method to handle device added event.
	PluginsChanged [1876]	Adds arguments for "plugins changed" event handler.
	PluginsChanging [1876]	Adds arguments for "plugins changing" event handler.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	FromObject [1554]	Creates a new NObject [1549] from the specified object.
	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NPluginManager Class

	Name	Description
	EnsurePluginSearchPerformed [1874]	Checks if initial plugin search was performed and if not calls Refresh [1874]
	GetInstances [1874]	Gets instance of the objects.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	GetInterfaceVersions [1874]	Returns the range of supported interface versions by specified NPluginManager.
	PlugAll [1874]	Plugs all plugins into the system.
	Refresh [1874]	Refreshes plugins list
	UnplugAll [1874]	Unplugs all plugins.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NPluginManager Class

	Name	Description
	AllowsUnplug [1875]	Checks whether PlugAll [1874] and UnplugAll [1874] methods work
	DisabledPlugins [1875]	Gets disabled plugins collection.
	InterfaceType [1875]	Gets NPluginManager interface type name.
S	NativeType [1875]	Defines the native type of the object.
	Plugins [1875]	Gets collection of all plugins in NPluginManager.
	PluginSearchPath [1875]	Gets or sets plugins and its dependencies search path.

7.2.7.1.2.1 NPluginManager Classes

7.2.7.1.2.1.1 NPluginManager.DisabledPluginCollection Class

Represents collection of plugin names for plugins that will be disabled.

C#

```
public class DisabledPluginCollection : StringCollection;
```

Remarks

To view or modify the collection use the methods from the base class: StringCollection.

7.2.7.1.2.1.2 NPluginManager.PluginCollection Class

Represents collection of plugins.

C#

```
public sealed class PluginCollection : NObjectReadOnlyCollection<NPlugin>;
```

Remarks

To iterate over the collection use the interfaces supported by the base class: NObjectReadOnlyCollection.

PluginCollection Methods

	Name	Description
	Add [1873]	Adds plugin and its dependencies (if exist) from specified directory.
	IndexOf [1873]	Retrieves the index of the plugin specified by name.
	this [1873]	Retrieves a plugin specified by name.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.7.1.2.1.2.1 PluginCollection Methods

7.2.7.1.2.1.2.1.1 Add Method

7.2.7.1.2.1.2.1.1.1 NPluginManager.PluginCollection.Add Method (NPluginModule, string)

Adds plugin and its dependencies (if exist) from specified directory.

C#

```
public NPlugin Add(NPluginModule module, string directory);
```

Parameters

Parameters	Description
NPluginModule module	Plugin module.
string directory	Directory where plugin and its dependencies (if exist) are saved.

Returns

Added plugin.

7.2.7.1.2.1.2.1.1.2 NPluginManager.PluginCollection.Add Method (string)

Adds plugin from user specified file.

C#

```
public NPlugin Add(string fileName);
```

Parameters

Parameters	Description
string fileName	File name where plugin is saved.

Returns

Added plugin.

Remarks

This method automatically searches for library dependencies in the directory of the plugin library (specified in f*ileName*).

7.2.7.1.2.1.2.1.2 NPluginManager.PluginCollection.IndexOf Method

Retrieves the index of the plugin specified by name.

C#

```
public int IndexOf(string name);
```

Parameters

Parameters	Description
string name	The name of the plugin.

Returns

The index of the plugin.

7.2.7.1.2.1.2.1.3 NPluginManager.PluginCollection.this Indexer

Retrieves a plugin specified by name.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NPlugin this[string name];
```

Parameters

Parameters	Description
string name	Plugin name.

Returns

NPlugin [1863] object.

7.2.7.1.2.2 NPluginManager Methods

7.2.7.1.2.2.1 NPluginManager.EnsurePluginSearchPerformed Method

Checks if initial plugin search was performed and if not calls Refresh [1874]

C#

```
public void EnsurePluginSearchPerformed();
```

7.2.7.1.2.2.2 NPluginManager.GetInstances Method

Gets instance of the objects.

C#

```
public static NPluginManager[] GetInstances();
```

Returns

Array of object instqances.

7.2.7.1.2.2.3 NPluginManager.GetInterfaceVersions Method

Returns the range of supported interface versions by specified NPluginManager [1869].

C#

```
public NVersionRange[] GetInterfaceVersions();
```

Returns

NVersionRange [1645] array with all supported interface version values.

7.2.7.1.2.2.4 NPluginManager.PlugAll Method

Plugs all plugins into the system.

C#

```
public void PlugAll();
```

7.2.7.1.2.2.5 NPluginManager.Refresh Method

Refreshes plugins list

C#

```
public void Refresh();
```

7.2.7.1.2.2.6 NPluginManager.UnplugAll Method

Unplugs all plugins.

C#

```
public void UnplugAll();
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.7.1.2.3 NPluginManager Properties

7.2.7.1.2.3.1 NPluginManager.AllowsUnplug Property

Checks whether PlugAll [1874] and UnplugAll [1874] methods work

C#

```
public bool AllowsUnplug;
```

Property value

Bool value.

7.2.7.1.2.3.2 NPluginManager.DisabledPlugins Property

Gets disabled plugins collection.

C#

```
public DisabledPluginCollection DisabledPlugins;
```

Property value

Collection of names of plugins that are disabled.

7.2.7.1.2.3.3 NPluginManager.InterfaceType Property

Gets NPluginManager [1869] interface type name.

C#

```
public string InterfaceType;
```

Property value

String with NPluginManager [1869] interface name.

7.2.7.1.2.3.4 NPluginManager.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.7.1.2.3.5 NPluginManager.Plugins Property

Gets collection of all plugins in NPluginManager [1869].

C#

```
public PluginCollection Plugins;
```

Property value

Plugins collection.

7.2.7.1.2.3.6 NPluginManager.PluginSearchPath Property

Gets or sets plugins and its dependencies search path.

C#

```
public string PluginSearchPath;
```

Property value

Plugin search path.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.7.1.2.4 NPluginManager Events

7.2.7.1.2.4.1 NPluginManager.DisabledPluginsChanged Event

Adds arguments for "disabled plugins changed" event handler.

C#

```
public event EventHandler<EventArgs> DisabledPluginsChanged;
```

Remarks

"Disabled plugins changed" event is raised when disabled plugins collection was changed.

7.2.7.1.2.4.2 NPluginManager.DisabledPluginsChanging Event

Adds arguments for "disabled plugins changing" event handler.

C#

```
public event EventHandler<EventArgs> DisabledPluginsChanging;
```

Remarks

"Disabled plugins changing" event is raised when disabled plugins collection is being changed.

7.2.7.1.2.4.3 NPluginManager.PluginAdded Event

Represents the method to handle device added event.

C#

```
public event EventHandler<NPluginManagerPluginEventArgs> PluginAdded;
```

7.2.7.1.2.4.4 NPluginManager.PluginsChanged Event

Adds arguments for "plugins changed" event handler.

C#

```
public event EventHandler<EventArgs> PluginsChanged;
```

7.2.7.1.2.4.5 NPluginManager.PluginsChanging Event

Adds arguments for "plugins changing" event handler.

C#

```
public event EventHandler<EventArgs> PluginsChanging;
```

7.2.7.1.3 NPluginManagerPluginEventArgs Class

Base class for holding NPluginManager [1869] events data.

C#

```
public class NPluginManagerPluginEventArgs : EventArgs;
```

Methods

	Name	Description
	NPluginManagerPluginEventArgs [1877]	Plugin [1877] constructor.

NPluginManagerPluginEventArgs Properties

	Name	Description
	Plugin [1877]	Gets Plugin associated with event raised by PluginManager.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.7.1.3.1 NPluginManagerPluginEventArgs.NPluginManagerPluginEventArgs Constructor

Plugin [1877] constructor.

C#

```
public NPluginManagerPluginEventArgs(NPlugin plugin);
```

Parameters

Parameters	Description
NPlugin plugin	The plugin associated with event.

7.2.7.1.3.2 NPluginManagerPluginEventArgs Properties

7.2.7.1.3.2.1 NPluginManagerPluginEventArgs.Plugin Property

Gets Plugin associated with event raised by PluginManager.

C#

```
public NPlugin Plugin;
```

Property value

NPlugin [1863] object.

7.2.7.1.4 NPluginModule Class

The module providing one or more plugins.

Provides methods to retrieve information about plugins.

C#

```
public sealed class NPluginModule : NModule;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

Methods

NModule Class

	Name	Description
	NModule [1546]	Initializes a new instance of NModule class.

NPluginModule Class

	Name	Description
	NPluginModule [1880]	Creates a new instance of NPluginModule class.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NModule Class

	Name	Description
	CheckInit [1546]	Checks whether the current module is initialized and throws and exception if it is not.
	GetDependences [1546]	Retrieves dependencies of the module.
	GetLoadedModules [1547]	Gets all the loaded modules that are part of this assembly.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPluginModule Class

	Name	Description
	GetInterfaceVersions [1881]	Returns the range of supported interface versions by current NPluginModule object.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NModule Class

	Name	Description
	Activated [1547]	Checks if the module is registered.
	Company [1547]	Retrieves company name.
	Copyright [1547]	Retrieves copyright notice of this module.
	Name [1547]	Retrieves module name.
S	NativeType [1547]	Defines the native type of the object.
S	NModuleOptionsNativeType [1548]	Defines the native type of the module options.
	Options [1548]	Retrieves module options.
	Product [1548]	Retrieves product name.
	Title [1548]	Retrieves module's title.
	Version [1548]	Retrieves module version number.
	VersionBuild [1548]	Retrieves the build of the current version.
	VersionMajor [1548]	Retrieves the major version value.
	VersionMinor [1548]	Retrieves the minor version value.
	VersionRevision [1548]	Retrieves revision of the current version.

NPluginModule Class

	Name	Description
	IncompatiblePlugins [1881]	Gets the list of incompatible plugins.
	InterfaceType [1881]	Gets plugin interface type name.
S	NativeType [1881]	Defines the native type of the object.
	PluginName [1881]	Gets plugin name.
	Priority [1882]	Gets plugin priority over other plugins.

7.2.7.1.4.1 NPluginModule.NPluginModule Constructor

Creates a new instance of NPluginModule class.

C#

```
public NPluginModule();
```

7.2.7.1.4.2 NPluginModule Methods

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.7.1.4.2.1 NPluginModule.GetInterfaceVersions Method

Returns the range of supported interface versions by current NPluginModule [1877] object.

C#

```
public NVersionRange[] GetInterfaceVersions();
```

Returns

NVersionRange [1645] array with all supported interface version values.

Notes

NVersionRange [1645]* and NVersion [1640]* macros can be used to parse version range.

7.2.7.1.4.3 NPluginModule Properties

7.2.7.1.4.3.1 NPluginModule.IncompatiblePlugins Property

Gets the list of incompatible plugins.

C#

```
public string IncompatiblePlugins;
```

Remarks

This property retrieves only the list of incompatible plug-in names. Plug-in file names can be retrieved using FileName property.

Property value

String with incompatible plugin names. Plugin names are separated by semicolons (;).

7.2.7.1.4.3.2 NPluginModule.InterfaceType Property

Gets plugin interface type name.

C#

```
public string InterfaceType;
```

Property value

String with plugin interface type name.

7.2.7.1.4.3.3 NPluginModule.NativeType Property

Defines the native type of the object.

C#

```
public static new NType NativeType;
```

7.2.7.1.4.3.4 NPluginModule.PluginName Property

Gets plugin name.

C#

```
public string PluginName;
```

Property value

String with plugin name.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.7.1.4.3.5 NPluginModule.Priority Property

Gets plugin priority over other plugins.

C#

```
public int Priority;
```

Remarks

Priority influences plugins adding to the system. If two or more plugins are incompatible or duplicate plugins exist plugin with higher priority will be added to the system. Priority is set during plugin compilation.

Default plugin priority is 0.

Property value

Plugin priority over other plugins.

7.2.7.2 Structs, Records, Enums

7.2.7.2.1 Neurotec.Plugins.NPluginState Enumeration

Enumerates NPlugin [1863] state values. The plugin state show how much of initialization the plugin has performed and can be used to identify common problems of plugin loading.

C#

```
public enum NPluginState {
  None = 0,
  LoadError = 1,
  NotRecognized = 2,
  InvalidModule = 3,
  InterfaceTypeMismatch = 4,
  InterfaceVersionMismatch = 5,
  InvalidInterface = 6,
  Unplugged = 64,
  Unused = 65,
  Disabled = 66,
  Duplicate = 67,
  IncompatibleWithOtherPlugins = 68,
  PluggingError = 69,
  Plugged = 128
}
```

Members

Members	Description
None = 0	Indicates uninitialized variable. This value should never be observed by the API user.
LoadError = 1	Indicates that plugin's dynamic library (dll) failed to load.
NotRecognized = 2	Indicates that function returning plugin module handle was not found. If the file name of plugin library is "<*Name*>.dll", the expected name of the function is "<*Name*>ModuleOf".
InvalidModule = 3	Indicates invalid plugin module. The plugin's ModuleOf function (see: NotRecognized) was found, but the returned handle was wrong.
InterfaceTypeMismatch = 4	Indicates that plugin's interface type differs from expected type. This means that the plugin is for other product/application. Interface type can be retrieved using NPluginModule.InterfaceType property.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

InterfaceVersionMismatch = 5	Indicates that plug-in interface version mismatches. This means that plugin is of the right type, but has version incompatible with versions supported by current plugin loader. Interface version can be retrieved using NPlugin.SelectedInterfaceVersion property.
InvalidInterface = 6	Indicates that the plugin's interface didn't pass the validation check. This happens when some mandatory functions required for plugin are missing or return wrong value.
Unplugged = 64	Indicates that plugin is not connected into the system. The plugin will not be used by the system unless activated with NPlugin.Plug [1867] or NPluginManager.PlugAll [1874]/NPluginManager.Refresh [1874] methods.
Unused = 65	Indicates that plugin is activated (with NPlugin.Plug [1867] or NPluginManager.PlugAll [1874]/NPluginManager.Refresh [1874] methods) but not currently used (not plugged into the system because there are no potential users of the plugin). The plugin will be automatically plugged once device manager of appropriate type is created (see: Plugged).
Disabled = 66	Indicates that plugin is disabled.
Duplicate = 67	Indicates that NPluginManager [1869] already contains plug-in with the same name.
IncompatibleWithOtherPlugins = 68	Indicates that the current plugin is incompatible with other plugin in the system. Each plugin has a priority value which determines which of the incompatible plugins will be loaded. The plugin with the highest priority will be loaded, and other plugins will fail with npsIncompatibleWithOtherPlugins state.
PluggingError = 69	Indicates that an error occurred when plugging the plug-in. NPlugin.GetError method can be used to retrieve exception.
Plugged = 128	Indicates that plugin is activated and the provided functionality can be used by the system. If all the device managers enumerating the devices provided by the current plugin are terminated, the plugin will be automatically unplugged and it's state changed to Unused.

7.2.8 Neurotec.Surveillance Namespace

Provides functionality for real-time faces identification.

Classes

	Name	Description
	NsedMatch [1884]	Structure defining a Score [1884]-Id pair of a match in internal database.
	NSurveillance [1885]	Base Surveillance class.
	NSurveillanceEventArgs [1902]	A class that is passed to event handler and through which event data can be accessed.
	NSurveillanceEventDetails [1903]	A class that provides access to event details data.

Module

.NET Reference [1468]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Structs, Records, Enums

	Name	Description
	NSurveillanceEventType [1908]	Enumeration type for different Surveillance events. Different information is available inside EventDetails field of NSurveillanceEventArgs [1902]. Events are asynchronous, so no assumption on the order of events should be made.

7.2.8.1 Classes

7.2.8.1.1 NsedMatch Structure

Structure defining a Score [1884]-Id pair of a match in internal database.

C#

```
public struct NsedMatch {
}
```

NsedMatch Properties

	Name	Description
S	NativeType [1884]	Defines the native type of the object.
	Score [1884]	Gets or sets the matching score.
	TemplateId [1884]	Gets or sets the Id of matched template.

7.2.8.1.1.1 NsedMatch Properties

7.2.8.1.1.1.1 NsedMatch.NativeType Property

Defines the native type of the object.

C#

```
public static NType NativeType;
```

7.2.8.1.1.1.2 NsedMatch.Score Property

Gets or sets the matching score.

C#

```
public int Score;
```

Property value

Integer value that represents matching score.

7.2.8.1.1.1.3 NsedMatch.TemplateId Property

Gets or sets the Id of matched template.

C#

```
public object TemplateId;
```

Property value

Object that contains template Id.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.8.1.2 NSurveillance Class

Base Surveillance [1883] class.

C#

```
public sealed class NSurveillance : NObject;
```

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

NSurveillance Class

	Name	Description
	Any [1901]	This callback is called on every event.
	FacesDetected [1901]	This callback is called when faces are detected.
	ImageCaptured [1902]	This event is fired when a new image is captured.
	IsRunningChanged [1902]	This event is called when Surveillance [1883] "running" task changed.
	SubjectAppeared [1902]	This event is fired when a new subject appears in the field of view of the camera.
	SubjectDisappeared [1902]	This event is fired when a subject being tracked leaves the field of view of the camera.
	SubjectTrack [1902]	This event is fired when successfull identification is performed on a tracked subject.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NSurveillance Class

	Name	Description
	DllName [1891]	Name of the library containing unmanaged part of this class.
	ParameterDetectionDetectMoreFaces [1891]	Identifier used to choose different face detection procedures which result in high recall.
	ParameterDetectionFaceConfidenceThreshold [1891]	Specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100]. The default value is 52.
	ParameterDetectionFaceDetectionScaleCount [1891]	Identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate. Values are in range [1...16].
	ParameterDetectionFaceRollAngleBase [1891]	Identifier specifying a base angle for faces roll angle in the image to search. Values are in range [-180...180]. The default value is 0. Identifier is used to set base angle to support mobile devices with orientation sensor.
	ParameterDetectionMaxIod [1891]	Identifier is specifying maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 4000.
	ParameterDetectionMaxRollAngleDeviation [1891]	Identifier specifying the maximum roll angle deviation from frontal face in degrees. The value of this parameter can be in range [0..180]. The default value is 15.
	ParameterDetectionMaxYawAngleDeviation [1892]	Identifier specifying the maximum yaw angle deviation from frontal face in degrees. The value of this parameter can be in range [0..90]. The default value is 15.
	ParameterDetectionMinIod [1892]	Identifier is specifying minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 40.
	ParameterExtractionBlinkConfidenceThreshold [1892]	Closed eyes absence/presence confidence threshold.
	ParameterExtractionDarkGlassesConfidenceThreshold [1892]	Dark glasses on a face absence/presence confidence threshold.
	ParameterExtractionDetectBlink [1892]	Identifier used to detect closed eyes.
	ParameterExtractionDetectDarkGlasses [1892]	Identifier used to detect faces with dark glasses.
	ParameterExtractionDetectExpression [1892]	Identifier used to detect face expressions.
	ParameterExtractionDetectGender [1892]	Identifier specifying whether to use gender detection during extraction phase.
	ParameterExtractionDetectGlasses [1893]	Identifier used to detect faces with glasses.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	ParameterExtractionDetectMouthOpen [1893]	Identifier used to detect faces with mouth open.
	ParameterExtractionEnrollmentTemplateSize [1893]	Identifier specifying enrollment template size parameter. It is used in generating a template from a tracked subject, usually, on SubjectDisappeared [1902] event. Parameter value can be one of the NleTemplateSize [1779] enumeration members. The default value is NleTemplateSize.Medium.
	ParameterExtractionExpressionConfidenceThreshold [1893]	Face expressions absence/presence confidence threshold.
	ParameterExtractionFaceQualityThreshold [1893]	Specifies the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.
	ParameterExtractionGenderConfidenceThreshold [1893]	Identifier specifying gender detection confidence threshold.
	ParameterExtractionGlassesConfidenceThreshold [1894]	Glasses on a face absence/presence confidence threshold.
	ParameterExtractionIdentificationTemplateSize [1894]	Identifier specifying identification template size parameter. It is used in generating a template, which is further used in identification, from each detected face. Parameter value can be one of the NleTemplateSize [1779] enumeration members. The default value is NleTemplateSize.Medium.
	ParameterExtractionMaxRecordsPerTemplate [1894]	Sets the maximum number of records an extraction function can return in one NLTemplate [1730]. The value of this parameter can be in range [1..16]. The default value is 5.
	ParameterExtractionMouthOpenConfidenceThreshold [1894]	Face with mouth open absence/presence confidence threshold.
	ParameterIdentificationMatchingThreshold [1894]	Identifier specifying identification matching threshold (biggest allowed FAR) parameter of type UShort. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%). The default value is 48.
	ParameterIdentificationMaxBestMatchCount [1895]	Maximum length of the list which is returned by each identification. The default value is 10.
	ParameterMaxThreadCount [1895]	Id of the parameter used to set maximum number of used threads. The default value is 5.

Methods

NSurveillance Class

	Name	Description
	NSurveillance [1890]	Initializes a new instance the Surveillance [1883] class.

NDisposable Methods

	Name	Description
	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	FromObject [1554]	Creates a new NObject [1549] from the specified object.
	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
	GetSizeMany [1558]	Returns the size of objects in the array.
	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	Invoke [1559]	Invoke Event.
	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.
	SaveMany [1562]	Saves an array of objects into a buffer.
	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NSurveillance Class

	Name	Description
	AddEvent [1895]	Adds event to class.
	AddTemplate [1895]	Adds a template to the internal database.
	GetTemplateIds [1896]	Returns array of template Ids that internal database contains.
	RemoveAllTemplates [1896]	Clears internal template database.
	RemoveEvent [1896]	Removes the event from Surveillance [1883] framework.
	RemoveTemplate [1896]	Removes a template with specified Id from internal database.
	Start [1896]	Starts surveillance.
	Stop [1897]	Stops surveillance.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NSurveillance Class

	Name	Description
	DetectionDetectMoreFaces [1897]	
	DetectionFaceConfidenceThreshold [1897]	Gets or sets the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100].
	DetectionFaceDetectionScaleCount [1897]	
	DetectionFaceRollAngleBase [1897]	
	DetectionMaxIod [1897]	Gets or sets maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000].
	DetectionMaxRollAngleDeviation [1897]	Gets or sets maximum roll angle deviation from frontal face in degrees. The value of this parameter can be in range [0..180]. The default value is 15.
	DetectionMaxYawAngleDeviation [1898]	

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	DetectionMinIod [1898]	Gets or sets minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000].
	DeviceManager [1898]	Gets Device Manager.
	ExtractionBlinkConfidenceThreshold [1898]	Gets or sets closed eyes absence/presence confidence threshold.
	ExtractionDarkGlassesConfidenceThreshold [1898]	Gets or sets dark glasses confidence threshold.
	ExtractionDetectBlink [1898]	Gets or sets identifier used to detect closed eyes.
	ExtractionDetectDarkGlasses [1899]	Gets or sets identifier used to detect faces with dark glasses.
	ExtractionDetectExpression [1899]	Gets or sets identifier used to detect face expressions.
	ExtractionDetectGender [1899]	Gets or sets gender detection flag parameter.
	ExtractionDetectGlasses [1899]	Gets or sets identifier used to detect faces with glasses.
	ExtractionDetectMouthOpen [1899]	Gets or sets identifier used to detect faces with mouth open.
	ExtractionEnrollmentTemplateSize [1899]	Gets or sets enrollment template size parameter.
	ExtractionExpressionConfidenceThreshold [1900]	Gets or sets face expressions absence/presence confidence threshold.
	ExtractionFaceQualityThreshold [1900]	Gets or sets the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.
	ExtractionGenderConfidenceThreshold [1900]	Gets or sets gender detection confidence threshold parameter.
	ExtractionGlassesConfidenceThreshold [1900]	Gets or sets glasses on a face absence/presence confidence threshold.
	ExtractionIdentificationTemplateSize [1900]	Gets or sets identification template size parameter.
	ExtractionMaxRecordsPerTemplate [1900]	Gets or sets the maximum number of records an extraction function can return in one NLTemplate [1730]. The value of this parameter can be in range [1..16]. The default value is 5.
	ExtractionMouthOpenConfidenceThreshold [1901]	Gets or sets face with mouth open absence/presence confidence threshold.
	IdentificationMatchingThreshold [1901]	Gets or sets matching threshold (biggest allowed FAR) parameter of type UShort. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).
	IdentificationMaxBestMatchCount [1901]	Gets or sets maximum length of the list which is returned by each identification.
	IsRunning [1901]	Checks if Surveillance [1883] is running (performing operations).
	MaxThreadCount [1901]	Gets or sets Id of the parameter used to set number of detection threads.
S	NativeType [1901]	Neurotechnology framework type of the object.

7.2.8.1.2.1 NSurveillance.NSurveillance Constructor

Initializes a new instance the Surveillance [1883] class.

C#

```
public NSurveillance();
```

7.2.8.1.2.2 NSurveillance Fields

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.8.1.2.2.1 NSurveillance.DllName Field

Name of the library containing unmanaged part of this class.

C#

```
public const string DllName = "NSurveillance";
```

7.2.8.1.2.2.2 NSurveillance.ParameterDetectionDetectMoreFaces Field

Identifier used to choose different face detection procedures which result in high recall.

C#

```
public const ushort ParameterDetectionDetectMoreFaces = 10107;
```

7.2.8.1.2.2.3 NSurveillance.ParameterDetectionFaceConfidenceThreshold Field

Specifies the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100]. The default value is 52.

C#

```
public const ushort ParameterDetectionFaceConfidenceThreshold = 10104;
```

Notes

Parameter cannot be changed while surveillance is running (cannot be changed between Start [1896] and Stop [1897] calls).

7.2.8.1.2.2.4 NSurveillance.ParameterDetectionFaceDetectionScaleCount Field

Identifier specifying number of scales in one octave when constructing scale space for face detection. Higher values mean slower face detection but higher recall rate. Values are in range [1...16].

C#

```
public const ushort ParameterDetectionFaceDetectionScaleCount = 10108;
```

7.2.8.1.2.2.5 NSurveillance.ParameterDetectionFaceRollAngleBase Field

Identifier specifying a base angle for faces roll angle in the image to search. Values are in range [-180...180]. The default value is 0. Identifier is used to set base angle to support mobile devices with orientation sensor.

C#

```
public const ushort ParameterDetectionFaceRollAngleBase = 10109;
```

7.2.8.1.2.2.6 NSurveillance.ParameterDetectionMaxIod Field

Identifier is specifying maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 4000.

C#

```
public const ushort ParameterDetectionMaxIod = 10103;
```

Notes

Parameter cannot be changed while surveillance is running (cannot be changed between Start [1896] and Stop [1897] calls).

7.2.8.1.2.2.7 NSurveillance.ParameterDetectionMaxRollAngleDeviation Field

Identifier specifying the maximum roll angle deviation from frontal face in degrees. The value of this parameter can be in range [0..180]. The default value is 15.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public const ushort ParameterDetectionMaxRollAngleDeviation = 10105;
```

7.2.8.1.2.2.8 NSurveillance.ParameterDetectionMaxYawAngleDeviation Field

Identifier specifying the maximum yaw angle deviation from frontal face in degrees. The value of this parameter can be in range [0..90]. The default value is 15.

C#

```
public const ushort ParameterDetectionMaxYawAngleDeviation = 10106;
```

7.2.8.1.2.2.9 NSurveillance.ParameterDetectionMinIod Field

Identifier is specifying minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000]. The default value is 40.

C#

```
public const ushort ParameterDetectionMinIod = 10102;
```

Notes

Parameter cannot be changed while surveillance is running (cannot be changed between Start [1896] and Stop [1897] calls).

7.2.8.1.2.2.10 NSurveillance.ParameterExtractionBlinkConfidenceThreshold Field

Closed eyes absence/presence confidence threshold.

C#

```
public const ushort ParameterExtractionBlinkConfidenceThreshold = 10215;
```

7.2.8.1.2.2.11 NSurveillance.ParameterExtractionDarkGlassesConfidenceThreshold Field

Dark glasses on a face absence/presence confidence threshold.

C#

```
public const ushort ParameterExtractionDarkGlassesConfidenceThreshold = 10221;
```

7.2.8.1.2.2.12 NSurveillance.ParameterExtractionDetectBlink Field

Identifier used to detect closed eyes.

C#

```
public const ushort ParameterExtractionDetectBlink = 10214;
```

7.2.8.1.2.2.13 NSurveillance.ParameterExtractionDetectDarkGlasses Field

Identifier used to detect faces with dark glasses.

C#

```
public const ushort ParameterExtractionDetectDarkGlasses = 10220;
```

7.2.8.1.2.2.14 NSurveillance.ParameterExtractionDetectExpression Field

Identifier used to detect face expressions.

C#

```
public const ushort ParameterExtractionDetectExpression = 10212;
```

7.2.8.1.2.2.15 NSurveillance.ParameterExtractionDetectGender Field

Identifier specifying whether to use gender detection during extraction phase.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public const ushort ParameterExtractionDetectGender = 10210;
```

Notes

Parameter cannot be changed while surveillance is running (cannot be changed between Start [1896] and Stop [1897] calls).

7.2.8.1.2.2.16 NSurveillance.ParameterExtractionDetectGlasses Field

Identifier used to detect faces with glasses.

C#

```
public const ushort ParameterExtractionDetectGlasses = 10218;
```

7.2.8.1.2.2.17 NSurveillance.ParameterExtractionDetectMouthOpen Field

Identifier used to detect faces with mouth open.

C#

```
public const ushort ParameterExtractionDetectMouthOpen = 10216;
```

7.2.8.1.2.2.18 NSurveillance.ParameterExtractionEnrollmentTemplateSize Field

Identifier specifying enrollment template size parameter. It is used in generating a template from a tracked subject, usually, on SubjectDisappeared [1902] event. Parameter value can be one of the NleTemplateSize [1779] enumeration members. The default value is NleTemplateSize.Medium.

C#

```
public const ushort ParameterExtractionEnrollmentTemplateSize = 10204;
```

Notes

Enrollment template size cannot be larger than identification template size.

Parameter cannot be changed while surveillance is running (cannot be changed between Start [1896] and Stop [1897] calls).

7.2.8.1.2.2.19 NSurveillance.ParameterExtractionExpressionConfidenceThreshold Field

Face expressions absence/presence confidence threshold.

C#

```
public const ushort ParameterExtractionExpressionConfidenceThreshold = 10213;
```

7.2.8.1.2.2.20 NSurveillance.ParameterExtractionFaceQualityThreshold Field

Specifies the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.

C#

```
public const ushort ParameterExtractionFaceQualityThreshold = 10202;
```

Notes

Parameter cannot be changed while surveillance is running (cannot be changed between Start [1896] and Stop [1897] calls).

7.2.8.1.2.2.21 NSurveillance.ParameterExtractionGenderConfidenceThreshold Field

Identifier specifying gender detection confidence threshold.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public const ushort ParameterExtractionGenderConfidenceThreshold = 10211;
```

Notes

Parameter cannot be changed while surveillance is running (cannot be changed between Start [1896] and Stop [1897] calls).

7.2.8.1.2.2.22 NSurveillance.ParameterExtractionGlassesConfidenceThreshold Field

Glasses on a face absence/presence confidence threshold.

C#

```
public const ushort ParameterExtractionGlassesConfidenceThreshold = 10219;
```

7.2.8.1.2.2.23 NSurveillance.ParameterExtractionIdentificationTemplateSize Field

Identifier specifying identification template size parameter. It is used in generating a template, which is further used in identification, from each detected face. Parameter value can be one of the NleTemplateSize [1779] enumeration members. The default value is NleTemplateSize.Medium.

C#

```
public const ushort ParameterExtractionIdentificationTemplateSize = 10205;
```

Notes

Identification template size cannot be smaller than enrollment template size.

Parameter cannot be changed while surveillance is running (cannot be changed between Start [1896] and Stop [1897] calls).

7.2.8.1.2.2.24 NSurveillance.ParameterExtractionMaxRecordsPerTemplate Field

Sets the maximum number of records an extraction function can return in one NLTemplate [1730]. The value of this parameter can be in range [1..16]. The default value is 5.

C#

```
public const ushort ParameterExtractionMaxRecordsPerTemplate = 10203;
```

Notes

Parameter cannot be changed while surveillance is running (cannot be changed between Start [1896] and Stop [1897] calls).

7.2.8.1.2.2.25 NSurveillance.ParameterExtractionMouthOpenConfidenceThreshold Field

Face with mouth open absence/presence confidence threshold.

C#

```
public const ushort ParameterExtractionMouthOpenConfidenceThreshold = 10217;
```

7.2.8.1.2.2.26 NSurveillance.ParameterIdentificationMatchingThreshold Field

Identifier specifying identification matching threshold (biggest allowed FAR) parameter of type UShort. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%). The default value is 48.

C#

```
public const ushort ParameterIdentificationMatchingThreshold = 10302;
```

Notes

Parameter cannot be changed while surveillance is running (cannot be changed between Start [1896] and Stop [1897] calls).

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.8.1.2.2.27 NSurveillance.ParameterIdentificationMaxBestMatchCount Field

Maximum length of the list which is returned by each identification. The default value is 10.

C#

```
public const ushort ParameterIdentificationMaxBestMatchCount = 10303;
```

Notes

Parameter cannot be changed while surveillance is running (cannot be changed between Start [1896] and Stop [1897] calls).

7.2.8.1.2.2.28 NSurveillance.ParameterMaxThreadCount Field

Id of the parameter used to set maximum number of used threads. The default value is 5.

C#

```
public const ushort ParameterMaxThreadCount = 10001;
```

Remarks

Minimal thread count should be N*3 + 2 where N is running cameras count. If ParameterMaxThreadCount is set less than this value an exception should appear.

Notes

Parameter cannot be changed while surveillance is running (cannot be changed between Start [1896] and Stop [1897] calls).

7.2.8.1.2.3 NSurveillance Methods

7.2.8.1.2.3.1 NSurveillance.AddEvent Method

Adds event to class.

C#

```
public void AddEvent(NSurveillanceEventType eventType, EventHandler<NSurveillanceEventArgs>
handler);
```

Parameters

Parameters	Description
NSurveillanceEventType eventType	Type of event to add.
EventHandler<NSurveillanceEventArgs> handler	Events handler.

7.2.8.1.2.3.2 AddTemplate Method

7.2.8.1.2.3.2.1 NSurveillance.AddTemplate Method (NBuffer, object)

Adds a template to the internal database.

C#

```
public void AddTemplate(NBuffer template, object templateId);
```

Parameters

Parameters	Description
NBuffer template	Bytes array which contains template to add.
object templateId	Id of template to add.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.8.1.2.3.2.2 NSurveillance.AddTemplate Method (byte[], object)

Adds a template to the internal database.

C#

```
public void AddTemplate(byte[] template, object templateId);
```

Parameters

Parameters	Description
byte[] template	Bytes array which contains template to add.
object templateId	Id of template to add.

7.2.8.1.2.3.3 NSurveillance.GetTemplateIds Method

Returns array of template Ids that internal database contains.

C#

```
public object[] GetTemplateIds();
```

Returns

Array which contains Ids of all templates.

7.2.8.1.2.3.4 NSurveillance.RemoveAllTemplates Method

Clears internal template database.

C#

```
public void RemoveAllTemplates();
```

7.2.8.1.2.3.5 NSurveillance.RemoveEvent Method

Removes the event from Surveillance [1883] framework.

C#

```
public void RemoveEvent(NSurveillanceEventType eventType,
EventHandler<NSurveillanceEventArgs> handler);
```

Parameters

Parameters	Description
NSurveillanceEventType eventType	Event Type which will be removed.
EventHandler<NSurveillanceEventArgs> handler	EventHandler which has the event to be removed

7.2.8.1.2.3.6 NSurveillance.RemoveTemplate Method

Removes a template with specified Id from internal database.

C#

```
public void RemoveTemplate(object templateId);
```

Parameters

Parameters	Description
object templateId	Id of template to remove.

7.2.8.1.2.3.7 NSurveillance.Start Method

Starts surveillance.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public void Start();
```

7.2.8.1.2.3.8 NSurveillance.Stop Method

Stops surveillance.

C#

```
public void Stop();
```

7.2.8.1.2.4 NSurveillance Properties

7.2.8.1.2.4.1 NSurveillance.DetectionDetectMoreFaces Property

C#

```
public bool DetectionDetectMoreFaces;
```

See Also

Neurotec.Surveillance.NSurveillance.ParameterDetectionMaxRollAngleDeviation [1891]

7.2.8.1.2.4.2 NSurveillance.DetectionFaceConfidenceThreshold Property

Gets or sets the threshold which is considered when looking for faces in an image. For each face candidate confidence parameter is calculated. With higher threshold value faces are selected more strictly by the face detection routines. Must be in range [0..100].

C#

```
public double DetectionFaceConfidenceThreshold;
```

7.2.8.1.2.4.3 NSurveillance.DetectionFaceDetectionScaleCount Property

C#

```
public int DetectionFaceDetectionScaleCount;
```

See Also

Neurotec.Surveillance.NSurveillance.ParameterDetectionMaxRollAngleDeviation [1891]

7.2.8.1.2.4.4 NSurveillance.DetectionFaceRollAngleBase Property

C#

```
public short DetectionFaceRollAngleBase;
```

See Also

Neurotec.Surveillance.NSurveillance.ParameterDetectionMaxRollAngleDeviation [1891]

7.2.8.1.2.4.5 NSurveillance.DetectionMaxIod Property

Gets or sets maximum distance between eyes in face. Faces [45] which have greater distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000].

C#

```
public int DetectionMaxIod;
```

7.2.8.1.2.4.6 NSurveillance.DetectionMaxRollAngleDeviation Property

Gets or sets maximum roll angle deviation from frontal face in degrees. The value of this parameter can be in range [0..180]. The default value is 15.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public short DetectionMaxRollAngleDeviation;
```

See Also

Neurotec.Surveillance.NSurveillance.ParameterDetectionMaxRollAngleDeviation [1891]

7.2.8.1.2.4.7 NSurveillance.DetectionMaxYawAngleDeviation Property

C#

```
public short DetectionMaxYawAngleDeviation;
```

See Also

Neurotec.Surveillance.NSurveillance.ParameterDetectionMaxRollAngleDeviation [1891]

7.2.8.1.2.4.8 NSurveillance.DetectionMinIod Property

Gets or sets minimum distance between eyes in face. Faces [45] which have smaller distance between eyes than this parameter, won't be returned by the face detection routines. Must be in range [10..10000].

C#

```
public int DetectionMinIod;
```

7.2.8.1.2.4.9 NSurveillance.DeviceManager Property

Gets Device Manager.

C#

```
public NDeviceManager DeviceManager;
```

7.2.8.1.2.4.10 NSurveillance.ExtractionBlinkConfidenceThreshold Property

Gets or sets closed eyes absence/presence confidence threshold.

C#

```
public byte ExtractionBlinkConfidenceThreshold;
```

Property value

Blink confidence threshold.

7.2.8.1.2.4.11 NSurveillance.ExtractionDarkGlassesConfidenceThreshold Property

Gets or sets dark glasses confidence threshold.

C#

```
public byte ExtractionDarkGlassesConfidenceThreshold;
```

Property value

Dark glasses confidence threshold.

7.2.8.1.2.4.12 NSurveillance.ExtractionDetectBlink Property

Gets or sets identifier used to detect closed eyes.

C#

```
public bool ExtractionDetectBlink;
```

Property value

Blink value.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.8.1.2.4.13 NSurveillance.ExtractionDetectDarkGlasses Property

Gets or sets identifier used to detect faces with dark glasses.

C#

```
public bool ExtractionDetectDarkGlasses;
```

Property value

Bool value indicating dark glasses.

7.2.8.1.2.4.14 NSurveillance.ExtractionDetectExpression Property

Gets or sets identifier used to detect face expressions.

C#

```
public bool ExtractionDetectExpression;
```

Property value

Face expression identifier.

7.2.8.1.2.4.15 NSurveillance.ExtractionDetectGender Property

Gets or sets gender detection flag parameter.

C#

```
public bool ExtractionDetectGender;
```

See Also

Neurotec.Surveillance.NSurveillance.ParameterExtractionIdentificationTemplateSize [1894]

7.2.8.1.2.4.16 NSurveillance.ExtractionDetectGlasses Property

Gets or sets identifier used to detect faces with glasses.

C#

```
public bool ExtractionDetectGlasses;
```

Property value

true if glasses were detected on a face; otherwise false.

7.2.8.1.2.4.17 NSurveillance.ExtractionDetectMouthOpen Property

Gets or sets identifier used to detect faces with mouth open.

C#

```
public bool ExtractionDetectMouthOpen;
```

Property value

true if face is with mouth open; otherwise false.

7.2.8.1.2.4.18 NSurveillance.ExtractionEnrollmentTemplateSize Property

Gets or sets enrollment template size parameter.

C#

```
public NleTemplateSize ExtractionEnrollmentTemplateSize;
```

See Also

Neurotec.Surveillance.NSurveillance.ParameterExtractionEnrollmentTemplateSize [1893]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.8.1.2.4.19 NSurveillance.ExtractionExpressionConfidenceThreshold Property

Gets or sets face expressions absence/presence confidence threshold.

C#

```
public byte ExtractionExpressionConfidenceThreshold;
```

Property value

Face expression confidence threshold.

7.2.8.1.2.4.20 NSurveillance.ExtractionFaceQualityThreshold Property

Gets or sets the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.

C#

```
public byte ExtractionFaceQualityThreshold;
```

7.2.8.1.2.4.21 NSurveillance.ExtractionGenderConfidenceThreshold Property

Gets or sets gender detection confidence threshold parameter.

C#

```
public byte ExtractionGenderConfidenceThreshold;
```

See Also

Neurotec.Surveillance.NSurveillance.ParameterExtractionIdentificationTemplateSize [1894]

7.2.8.1.2.4.22 NSurveillance.ExtractionGlassesConfidenceThreshold Property

Gets or sets glasses on a face absence/presence confidence threshold.

C#

```
public byte ExtractionGlassesConfidenceThreshold;
```

Property value

Glasses confidence threshold.

7.2.8.1.2.4.23 NSurveillance.ExtractionIdentificationTemplateSize Property

Gets or sets identification template size parameter.

C#

```
public NleTemplateSize ExtractionIdentificationTemplateSize;
```

See Also

Neurotec.Surveillance.NSurveillance.ParameterExtractionIdentificationTemplateSize [1894]

7.2.8.1.2.4.24 NSurveillance.ExtractionMaxRecordsPerTemplate Property

Gets or sets the maximum number of records an extraction function can return in one NLTemplate [1730]. The value of this parameter can be in range [1..16]. The default value is 5.

C#

```
public int ExtractionMaxRecordsPerTemplate;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.8.1.2.4.25 NSurveillance.ExtractionMouthOpenConfidenceThreshold Property

Gets or sets face with mouth open absence/presence confidence threshold.

C#

```
public byte ExtractionMouthOpenConfidenceThreshold;
```

Property value

Mouth open confidence threshold.

7.2.8.1.2.4.26 NSurveillance.IdentificationMatchingThreshold Property

Gets or sets matching threshold (biggest allowed FAR) parameter of type UShort. Parameter value is equal to -12 * log10(FAR) and must be not less than zero (for example, 48 for FAR = 0.01%).

C#

```
public int IdentificationMatchingThreshold;
```

7.2.8.1.2.4.27 NSurveillance.IdentificationMaxBestMatchCount Property

Gets or sets maximum length of the list which is returned by each identification.

C#

```
public int IdentificationMaxBestMatchCount;
```

7.2.8.1.2.4.28 NSurveillance.IsRunning Property

Checks if Surveillance [1883] is running (performing operations).

C#

```
public bool IsRunning;
```

Property value

bool type value which indicates if Surveillance [1883] is running.

7.2.8.1.2.4.29 NSurveillance.MaxThreadCount Property

Gets or sets Id of the parameter used to set number of detection threads.

C#

```
public int MaxThreadCount;
```

7.2.8.1.2.4.30 NSurveillance.NativeType Property

Neurotechnology framework type of the object.

C#

```
public static new NType NativeType;
```

7.2.8.1.2.5 NSurveillance Events

7.2.8.1.2.5.1 NSurveillance.Any Event

This callback is called on every event.

C#

```
public event EventHandler<NSurveillanceEventArgs> Any;
```

7.2.8.1.2.5.2 NSurveillance.FacesDetected Event

This callback is called when faces are detected.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public event EventHandler<NSurveillanceEventArgs> FacesDetected;
```

7.2.8.1.2.5.3 NSurveillance.ImageCaptured Event

This event is fired when a new image is captured.

C#

```
public event EventHandler<NSurveillanceEventArgs> ImageCaptured;
```

7.2.8.1.2.5.4 NSurveillance.IsRunningChanged Event

This event is called when Surveillance [1883] "running" task changed.

C#

```
public event EventHandler<EventArgs> IsRunningChanged;
```

7.2.8.1.2.5.5 NSurveillance.SubjectAppeared Event

This event is fired when a new subject appears in the field of view of the camera.

C#

```
public event EventHandler<NSurveillanceEventArgs> SubjectAppeared;
```

7.2.8.1.2.5.6 NSurveillance.SubjectDisappeared Event

This event is fired when a subject being tracked leaves the field of view of the camera.

C#

```
public event EventHandler<NSurveillanceEventArgs> SubjectDisappeared;
```

7.2.8.1.2.5.7 NSurveillance.SubjectTrack Event

This event is fired when successfull identification is performed on a tracked subject.

C#

```
public event EventHandler<NSurveillanceEventArgs> SubjectTrack;
```

7.2.8.1.3 NSurveillanceEventArgs Class

A class that is passed to event handler and through which event data can be accessed.

C#

```
public class NSurveillanceEventArgs : EventArgs;
```

NSurveillanceEventArgs Properties

	Name	Description
	EventDetails [1902]	Gets event details object.
	EventType [1903]	Gets event type.

7.2.8.1.3.1 NSurveillanceEventArgs Properties

7.2.8.1.3.1.1 NSurveillanceEventArgs.EventDetails Property

Gets event details object.

C#

```
public NSurveillanceEventDetails EventDetails;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Property value

NSurveillanceEventDetails [1903] object which contains event arguments.

7.2.8.1.3.1.2 NSurveillanceEventArgs.EventType Property

Gets event type.

C#

```
public NSurveillanceEventType EventType;
```

Property value

NSurveillanceEventType [1908] object which contains event types.

7.2.8.1.4 NSurveillanceEventDetails Class

A class that provides access to event details data.

C#

```
public sealed class NSurveillanceEventDetails : NObject;
```

NSurveillanceEventDetails Classes

NSurveillanceEventDetails Class

	Name	Description
	BestMatchCollection [1906]	Contains best matching items.

NObject Delegates

NObject Class

Name	Description
EventInvoker [1568]	Creates a delegate to an event invoker function.
FromHandlePlain [1568]	Creates a delegate to FromHandlePlain function.
GetArrayPlain [1568]	Creates a delegate to a function, which gets a plain array.
GetArrayWithHandle [1569]	Creates a delegate to a function, which gets an array with handle.
GetObjectPlain [1569]	Creates a delegate to a function which returns a plain object.
GetObjectWithHandle [1569]	Creates a delegate to a function, which returns an object with handle.
GetOutArrayPlain [1569]	Creates a delegate to a function, which gets a plain out array.
GetOutArrayWithHandle [1570]	Creates a delegate to a function, which gets an out array with a handle.
GetStringPlainN [1570]	Creates a delegate to a function, which gets a plain string.
GetStringWithHandleN [1570]	Creates a delegate to a function, which gets a string with handle.
SetArrayPlain [1571]	Creates a delegate to a function, which sets values of a plain array.
SetArrayWithHandle [1571]	Creates a delegate to a function, which sets values of an array with handle.
SetObjectPlain [1571]	Creates a delegate to a function, which sets a value of a plain object.
SetObjectWithHandle [1571]	Creates a delegate to a function, which sets a value of an object with handle.
SetStringPlainN [1572]	Creates a delegate to a function, which sets a plain string.
SetStringWithHandleN [1572]	Creates a delegate to a function, which sets a string with handle.

NObject Events

NObject Class

	Name	Description
	PropertyChanged [1572]	Adds or removes a "Property Changed" event.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NObject Fields

NObject Class

	Name	Description
	FlagRefRet [1551]	This flag indicates that an object should be returned with incremented ref count.

NDisposable Methods

	Name	Description
V	Dispose [1536]	Performs tasks associated with freeing, releasing, or resetting unmanaged resources.

NObject Class

	Name	Description
	Clone [1551]	Creates a copy of specified object.
	CompareTo [1551]	Compares the current object with the specified version and returns an integer that indicates their relative position in the sort order.
S	CopyParameters [1552]	Copies parameters values from one NObject [1549] object to another.
S	CopyProperties [1552]	Copies proeprty values from one NObject [1549] object to another.
S	DisposeArray [1552]	Frees an array of objects.
	Equals [1552]	Compares the objects.
S	FromHandle [1552]	Create C# object from native API objects handle. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	FromObject [1554]	Creates a new NObject [1549] from the specified object.
S	GetArray [1554]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetHashCode [1555]	Gets hash code (numeric value) representing the object. Is intended for a hash function for a particular type. GetHashCode is suitable for use in hashing algorithms and data structures like a hash table.
	GetNativeType [1555]	Retrieves native type of object.
S	GetObject [1555]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	GetObjects [1556]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	GetParameter [1557]	Gets value of parameter by parameter id.
	GetProperty [1558]	Retrieves the specified property of the current object.
	GetSize [1558]	Returns the size of the current object.
S	GetSizeMany [1558]	Returns the size of objects in the array.
S	GetString [1559]	Helper function for getting data from the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	Invoke [1559]	Invoke Event.
S	Ref [1560]	Increments the reference count for an interface on a specified object. This function should be called for every new copy of a pointer to an interface of HNObject.
	Reset [1560]	Resets all NObject [1549] parameters to default values.
	ResetProperty [1560]	Resets the value of the specified property.
	Save [1560]	Saves the current object into a buffer.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

S	SaveMany [1562]	Saves an array of objects into a buffer.
S	SetArray [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	SetObject [1563]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
	SetParameter [1563]	Sets value of parameter by parameter id.
	SetProperty [1565]	Sets the value of the specified property.
S	SetString [1565]	Helper function for setting data of the native object. **This method supports the .NET Framework infrastructure and is not intended to be used directly from your code.**
S	ToHandle [1566]	Gets the representation of native API handle corresponding to the C# object.
S	ToObject [1566]	Creates a new object from the specified NObject [1549].
	ToString [1566]	Returns string representation of the object.
S	Unref [1567]	Decrements the reference count for an interface on a specified object. This function should be called on returned objects when they are no longer used to prevent memory leaks.
S	UnrefArray [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.
S	UnrefElements [1567]	Decrements the reference count for an interface on specified array of objects. This function should be called on returned objects array when they are no longer used to prevent memory leaks.

NSurveillanceEventDetails Class

	Name	Description
	GetFaces [1906]	Retrieves an array of structures containing information of a face location in an image.
	GetSubjectTemplate [1906]	Retrieves generalized Subject template.

NDisposable Properties

	Name	Description
V	IsDisposed [1536]	Gets whether allocated resources were released previously.

NObject Class

	Name	Description
	Flags [1567]	Gets flags from object.
	IsDisposed [1567]	Gets whether allocated resources were released previously.
S	NativeType [1568]	Neurotechnology framework type of the object.
	Owner [1568]	Gets owner of the object.

NSurveillanceEventDetails Class

	Name	Description
	BestMatches [1906]	Retrieves BestMatchesCollection.
	DetectionDetails [1906]	A property that returns detection details of a face being processed.
	ExtractionStatus [1907]	A property that returns extraction status of a face being processed
	FrameIndex [1907]	A property that returns the number of captured images prior to this event.
	GrayscaleImage [1907]	A property that returns grayscale version of originally captured image.
S	NativeType [1907]	Neurotechnology framework type of the object.
	OriginalImage [1907]	A property that returns originally captured image.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

	Owner [1907]	Gets owner of the object.
	SourceIndex [1907]	A property that returns an index of the camera the subject being tracked. The index is zero-based and represents unique id of camera.
	Template [1907]	A property that returns extracted serialized template.
	TimeStamp [1908]	A property that returns the date and time of the captured image.
	TraceIndex [1908]	A property that returns an index of the current subject being tracked. The index is zero-based and represents unique id of subject being tracked at the current moment. After subject disappears this index becomes invalid.

7.2.8.1.4.1 NSurveillanceEventDetails Classes

7.2.8.1.4.1.1 NSurveillanceEventDetails.BestMatchCollection Class

Contains best matching items.

C#

```
public sealed class BestMatchCollection : DisposableStructReadOnlyCollection<NsedMatch,
NsedMatchEx_>;
```

7.2.8.1.4.2 NSurveillanceEventDetails Methods

7.2.8.1.4.2.1 NSurveillanceEventDetails.GetFaces Method

Retrieves an array of structures containing information of a face location in an image.

C#

```
public NleFace[] GetFaces();
```

Returns

An array of NleFace [1701] structures.

7.2.8.1.4.2.2 NSurveillanceEventDetails.GetSubjectTemplate Method

Retrieves generalized Subject template.

C#

```
public NLTemplate GetSubjectTemplate();
```

Returns

Subject template is a **compressed** generalized template of all extracted faces during subject tracking whose Id matches that of the best matched Id. If there is no best match, then it includes all templates into generalization.

The template is updated every time a new face is determined to belong to the tracked subject.

Returned template must be cloned before using it.

7.2.8.1.4.3 NSurveillanceEventDetails Properties

7.2.8.1.4.3.1 NSurveillanceEventDetails.BestMatches Property

Retrieves BestMatchesCollection.

C#

```
public BestMatchCollection BestMatches;
```

7.2.8.1.4.3.2 NSurveillanceEventDetails.DetectionDetails Property

A property that returns detection details of a face being processed.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

C#

```
public NleDetectionDetails DetectionDetails;
```

7.2.8.1.4.3.3 NSurveillanceEventDetails.ExtractionStatus Property

A property that returns extraction status of a face being processed

C#

```
public NleExtractionStatus ExtractionStatus;
```

7.2.8.1.4.3.4 NSurveillanceEventDetails.FrameIndex Property

A property that returns the number of captured images prior to this event.

C#

```
public ulong FrameIndex;
```

7.2.8.1.4.3.5 NSurveillanceEventDetails.GrayscaleImage Property

A property that returns grayscale version of originally captured image.

C#

```
public NGrayscaleImage GrayscaleImage;
```

7.2.8.1.4.3.6 NSurveillanceEventDetails.NativeType Property

Neurotechnology framework type of the object.

C#

```
public static new NType NativeType;
```

7.2.8.1.4.3.7 NSurveillanceEventDetails.OriginalImage Property

A property that returns originally captured image.

C#

```
public NImage OriginalImage;
```

7.2.8.1.4.3.8 NSurveillanceEventDetails.Owner Property

Gets owner of the object.

C#

```
public new NSurveillance Owner;
```

Property value

Object (owner) of NObject type.

7.2.8.1.4.3.9 NSurveillanceEventDetails.SourceIndex Property

A property that returns an index of the camera the subject being tracked. The index is zero-based and represents unique id of camera.

C#

```
public int SourceIndex;
```

7.2.8.1.4.3.10 NSurveillanceEventDetails.Template Property

A property that returns extracted serialized template.

C#

```
public NLTemplate Template;
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.8.1.4.3.11 NSurveillanceEventDetails.TimeStamp Property

A property that returns the date and time of the captured image.

C#

```
public DateTime TimeStamp;
```

7.2.8.1.4.3.12 NSurveillanceEventDetails.TraceIndex Property

A property that returns an index of the current subject being tracked. The index is zero-based and represents unique id of subject being tracked at the current moment. After subject disappears this index becomes invalid.

C#

```
public int TraceIndex;
```

7.2.8.2 Structs, Records, Enums

7.2.8.2.1 Neurotec.Surveillance.NSurveillanceEventType Enumeration

Enumeration type for different Surveillance [1883] events. Different information is available inside EventDetails field of NSurveillanceEventArgs [1902]. Events are asynchronous, so no assumption on the order of events should be made.

C#

```
public enum NSurveillanceEventType {
  Any = -1,
  ImageCaptured = 1,
  FacesDetected = 2,
  SubjectAppeared = 3,
  SubjectTrack = 4,
  SubjectDisappeared = 5
}
```

Members

Members	Description
Any = -1	Indicates that any event can be captured by supplied callback function.
ImageCaptured = 1	Indicates that a new image was captured from capturing source.
FacesDetected = 2	Indicates that at least one face was successfully detected in the captured image.
SubjectAppeared = 3	Indicates that a new subject appeared in the field of view of the capturing device.
SubjectTrack = 4	Indicates that previously appeared subject is being tracked logging it's activity.
SubjectDisappeared = 5	Indicates that a previously tracked subject disappeared from the field of view of the capturing device.

Remarks

The table below explains what data is available during different events:

	ImageCaptured	FacesDetected	SubjectAppeared	SubjectTrack	SubjectDisappeared
CameraIndex	+	+	+	+	+
OriginalImage	+	+	+	+	
GrayscaleImage	+	+	+	+	
FrameIndex	+	+	+	+	

Copyright © 2009-2014 Neurotechnology. All rights reserved.

TimeStamp	+	+	+	+	+
GetFaces()		+			
DetectionDetails			+	+	
ExtractionStatus			+	+	
Template			+	+	
TraceIndex			+	+	+
GetBestMatches()			+	+	+
GetSubjectTemplate()			+	+	+

7.2.9 Functions

7.2.9.1 NLicensingServiceGetBinPath Function

Gets the binary path (path to `pg.exe/pgd`) to the license.

C++

```
NResult N_API NLicensingServiceGetBinPath(HNString * phValue);
```

Parameters

Parameters	Description
HNString * phValue	[out] Binary path to the license.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

.NET Reference [1468]

7.2.9.2 NLicensingServiceGetConfPath Function

Gets path to licensing service configuration file (`pgd.conf`).

C++

```
NResult N_API NLicensingServiceGetConfPath(HNString * phValue);
```

Parameters

Parameters	Description
HNString * phValue	[out] Path to the licensing service configuration file.

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

.NET Reference [1468]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.9.3 NLicensingServiceGetStatus Function

Gets the status of the licensing service.

C++

```
NResult N_API NLicensingServiceGetStatus(NLicensingServiceStatus * pValue);
```

Parameters

Parameters	Description
NLicensingServiceStatus * pValue	[out] Pointer to NLicensingServiceStatus [1911].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

.NET Reference [1468]

7.2.9.4 NLicensingServiceInstall Function

Installs licensing service.

C++

```
NResult N_API NLicensingServiceInstall(const NChar * szBinPath, const NChar * szConf);
```

Parameters

Parameters	Description
const NChar * szBinPath	[in] Path to the licensing service (`pg.exe/pgd`).
const NChar * szConf	[in] Path to licensing service configuration file (`pgd.conf`).

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

.NET Reference [1468]

7.2.9.5 NLicensingServiceIsTrial Function

Checks whether the licensing service is trial.

C++

```
NResult N_API NLicensingServiceIsTrial(NBool * pValue);
```

Parameters

Parameters	Description
NBool * pValue	[out] Pointer to NBool [1004] that receives NTrue [1064] value if licensing service is trial; otherwise, NFalse [1041].

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

.NET Reference [1468]

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.2.9.6 NLicensingServiceStart Function

Starts licensing service.

C++

```
NResult N_API NLicensingServiceStart();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

.NET Reference [1468]

7.2.9.7 NLicensingServiceStop Function

Stops licensing service.

C++

```
NResult N_API NLicensingServiceStop();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

.NET Reference [1468]

7.2.9.8 NLicensingServiceUninstall Function

Uninstalls licensing service.

C++

```
NResult N_API NLicensingServiceUninstall();
```

Returns

NResult [1007] which value can be checked using NFailed [434] and NSucceeded [434] macros.

Module

.NET Reference [1468]

7.2.10 Structs, Records, Enums

7.2.10.1 NLicensingServiceStatus Enumeration

Enumerates licensing service status values.

C++

```
typedef enum NLicensingServiceStatus_ {
  nlssNotInstalled = 0,
  nlssStopped = 0x00000001,
```

Copyright © 2009-2014 Neurotechnology. All rights reserved.

```
    nlssStartPending = 0x00000002,
    nlssStopPending = 0x00000003,
    nlssRunning = 0x00000004,
    nlssContinuePending = 0x00000005,
    nlssPausePending = 0x00000006,
    nlssPaused = 0x00000007
  } NLicensingServiceStatus;
```

Members

Members	Description
nlssNotInstalled = 0	Indicates that licensing service is nor installed.
nlssStopped = 0x00000001	Indicates that licensing service is stopped.
nlssStartPending = 0x00000002	Indicates that licensing service is pending to be started.
nlssStopPending = 0x00000003	Indicates that licensing service is pending to be stopped.
nlssRunning = 0x00000004	Indicates that licensing service is running.
nlssContinuePending = 0x00000005	Indicates that licensing service was continued after pause.
nlssPausePending = 0x00000006	Indicates that licensing service is pending to be paused.
nlssPaused = 0x00000007	Indicates that licensing service is paused.

Module

.NET Reference [1468]

7.2.11 Macros

7.2.11.1 NLicensingServiceInstall Macro

Picks a Unicode (W) or ANSCII (A) version of the NLicensingServiceInstall function.

C++

```
#define NLicensingServiceInstall N_FUNC_AW(NLicensingServiceInstall)
```

Module

.NET Reference [1468]

7.3 Change Log

In this section change log of the Neurotechnology API components is provided. These abbreviations are used:

- **FIX** - Fixed
- **CHN** - Changed
- **UPD** - Updated
- **ADD** - Added
- **REM** - Removed
- **MER** - Merged

Copyright © 2009-2014 Neurotechnology. All rights reserved.

7.3.1 C API Reference

7.3.1.1 NCore Library

Version 3.1.1.1

- **UPD**: N_E_COM [428], N_E_SYS [433] and N_E_WIN32 [433] now are inner errors if they are cause of another error.

Version 3.1.1.0

- **CHN**: Move NFileAccess [496] and NByteOrder [496] enum definitions from NTypes.h [1078] to NStream.h.

Version 3.1.0.4

- **FIX**: Minor fixes.

Version 3.1.0.3

- **FIX**: Minor fixes.

Version 3.1.0.2

- **FIX**: Minor fixes.

Version 3.1.0.1

- **FIX**: NTypeSupportsParameters function name.

Version 3.1.0.0

- **ADD**: HNObject, a handle to base object for other objects and operations (free and parameters) on it.
- **ADD**: HNType, a handle to type information for an object and operations on it.
- **ADD**: New, more specific error codes.
- **CHN**: N_CALLBACK [380] is now pointer-to-function.
- **REM**: Obsolete defines, functions and members.
- **UPD**: Refactoring and update of headers.

Version 3.0.0.3

- **FIX**: Crash when file open fails on *nix.

Version 3.0.0.2

- **UPD**: String parameter handling in parameter framework.

Version 3.0.0.1

- **FIX**: Minor fixes.

Version 3.0.0.0

- **UPD**: Unicode support made cross-platform.
- **ADD**: System error (errno) intergration to NErrors.
- **ADD**: SSSE4, SSE4.1, SSE4.2, SSE4A, LZCNT and POPCNT detection support to NProcessorInfo [637].
- **UPD**: NLibraryInfo structure.
- **ADD**: NCoreOnXxx functions for NCore [257] initialization/cleanup when it is a static library (does nothing when it is a

Copyright © 2009-2014 Neurotechnology. All rights reserved.

dynamic library).

Version 2.4.4.0

- **UPD**: Error framework made cross-platform.

Version 2.4.3.0

- **ADD**: Aligned memory management functions.

Version 2.4.2.1

- **FIX**: Minor fixes.

Version 2.4.2.0

- **ADD**: NPointF [1227], NPointD [1227], NSizeF [1227], NSizeD [1227], NRectF [1227] and NRectD [1227] types.
- **ADD**: NRange type.
- **UPD**: Internal optimizations.

Version 2.4.1.1

- **FIX**: Minor fixes.

Version 2.4.1.0

- **ADD**: NLibraryInfo module.
- **ADD**: NCoreGetInfo [353] function.

Version 2.4.0.0

- **ADD**: Integration with Win32 and COM errors on Windows.
- **ADD**: NStream [660] module.

Version 2.3.1.0

- **ADD**: NProcessorInfo [637] module for CPU identification on Windows.

Version 2.3.0.1

- **FIX**: Memory leak in parameters framework.

Version 2.3.0.0

- **ADD**: HNStream type.
- **ADD**: Stream integration with .NET.
- **UPD**: Exception integration with .NET.

Version 2.2.2.0

- **CHN**: NMemory [502] interface.

Version 2.2.1.0

- **ADD**: More robust error handling on Windows.

Version 2.2.0.0

- **ADD**: Unicode support.

Version 2.1.0.2

- **FIX**: Functions' calling convention on Windows.

Version 2.1.0.1

- **UPD**: Minor updates.

Version 2.1.0.0

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- **REM**: Registration error codes.
- **UPD**: Updated to use Microsoft Visual C++ Runtime Library 8.0.

Version 2.0.1.1

- **CHN**: Minor changes.

Version 2.0.1.0

- **ADD**: NParameters [575] module instead of NMetaTypes module for internal infrastructure support.
- **CHN**: Infrastructure optimization for 64-bit support.

Version 2.0.0.0

- **ADD**: A lot of stuff for internal infrastructure support.
- **CHN**: Some changes in internal infrastructure support.

Version 1.0.0.2

- **ADD**: NIndexPair [1006] structure.

Version 1.0.0.1

- **FIX**: Minor fixes in headers.

Version 1.0.0.0

- Initial release.

7.3.1.2 NLicensing Library

Version 3.1.0.0

- **ADD**: NLicenseIsComponentActivated [1192] function.
- **UPD**: Now uses NCore [257] library version 3.1.

Version 3.0.0.0

- New version of the NLicensing library released.

7.3.1.3 NSurveillance Library

Version 1.1.0.1

- **UPD**: Minor updates.

Version 1.1.0.0

- **CHN**: Upgraded face recognition algorithm.
- **CHN**: Extended API to support several instances of NSurveillance [1439] in one process.
- **ADD**: New algorithm parameteters NSP_DETECTION_MAX_ROLL_ANGLE_DEVIATION [1459], NSP_EXTRACTION_ENROLLMENT_TEMPLATE_SIZE [1461], and NSP_EXTRACTION_IDENTIFICATION_TEMPLATE_SIZE [1461].
- **UPD**: Now uses NExtractors library version 3.1.
- **UPD**: Now uses NMatchers library version 3.1.
- **UPD**: Now uses NTemplates library version 3.1.
- **UPD**: Now uses NImages library version 2.6.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- **UPD**: Now uses NCore [257] library version 3.1.

Version 1.0.0.0

- NSurveillance [1439] library released.

7.3.2 .NET API Reference

7.3.2.1 Neurotec Library

Version 3.1.1.3

- **UPD**: More symmetric exception integration.

Version 3.1.1.2

- **FIX**: NLibraryInfo portability issues.

Version 3.1.1.1

- **FIX**: Namespaces for collection classes.

Version 3.1.1.0

- **FIX**: Namespaces for some objects.

Version 3.1.0.4

- **FIX**: Marshalling of enumeration arrays in collection classes.

Version 3.1.0.3

- **FIX**: Hide some unused members of NCollection.

Version 3.1.0.2

- **FIX**: Minor fixes.

Version 3.1.0.1

- **UPD**: Updated to reflect changes in unmanaged code.

Version 3.1.0.0

- **ADD**: HNObject operations to NObject [536] class (parameters).
- **ADD**: HNType, a handle to type information for an object and operations on it.
- **ADD**: New, more specific error codes and exception classes.
- **ADD**: NReadOnlyCollection and NCollection classes and their subclasses as base classes for other NObject [536] internal collections.
- **REM**: Parameter-specific types (now NObject [536] has that functionality).
- **REM**: Obsolete members and types.

Version 3.0.0.1

- **FIX**: Stream wrapper crash sometimes.

Version 3.0.0.0

- **ADD**: System error (errno) intergration for Neurotec exceptions.
- **ADD**: SSSE4, SSE4.1, SSE4.2, SSE4A, LZCNT and POPCNT detection support to NProcessorInfo [637].

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- **UPD**: NLibraryInfo structure.

Version 2.4.5.1

- **FIX**: Minor fixes.

Version 2.4.5.0

- **ADD**: PointD, SizeD, RectangleD types.
- **ADD**: NRange type.

Version 2.4.4.0

- **ADD**: NMemoryStream [510] class.
- **ADD**: NBuffer.Wrap method.

Version 2.4.3.1

- **ADD**: Constants in NResult [1007] for defined Neurotec functions return codes.

Version 2.4.2.1

- **ADD**: NBuffer [313] class.
- **ADD**: Extended unmanaged memory management.

Version 2.4.1.1

- **FIX**: Cross-AppDomain usage issues.

Version 2.4.1.0

- **ADD**: NLibraryInfo class.
- **ADD**: NCore.GetInfo method.

Version 2.4.0.0

- **UPD**: Reflects changes in unmanaged code.

Version 2.3.1.0

- **ADD**: NProcessorInfo [637] class.

Version 2.3.0.0

- **ADD**: Stream integration with unmanaged code.
- **UPD**: Exception integration with unmanaged code.

Version 2.2.2.0

- **ADD**: More robust unmanaged error handling on Windows.

Version 2.2.1.0

- **ADD**: Classes for internal architecture support.

Version 2.2.0.0

- **UPD**: Updated to support changes in unmanaged code.

Version 2.1.0.0

- **REM**: LicenseManagerException class.
- **CHN**: Now uses Microsoft .NET Framework 2.0.

Version 2.0.1.2

- **FIX**: Minor fixes in parameters framework.

Version 2.0.1.1

Copyright © 2009-2014 Neurotechnology. All rights reserved.

- **FIX**: Minor fixes.
- **UPD**: Minor updates in structures.

Version 2.0.1.0

- **ADD**: NParameters [575] class for internal infrastructure support.
- **CHN**: Infrastructure optimization for 64-bit support.

Version 2.0.0.0

- **ADD**: A lot of stuff for internal infrastructure support.
- **CHN**: Some changes in internal infrastructure support.

Version 1.0.0.2

- **ADD**: NIndexPair [1006] structure.

Version 1.0.0.1

- **FIX**: All error codes are mapped to appropriate exceptions.

Version 1.0.0.0

- Initial release.

7.3.2.2 Neurotec.Biometrics.Gui

Version 3.1.0.0

- **CHN**: NView moved to Neurotec.Gui library.
- **UPD**: Now uses Neurotec.Images library version 2.6.
- **UPD**: Now uses Neurotec.Biometrics.Templates library version 3.1.
- **UPD**: Now uses Neurotec library version 3.1.

Version 3.0.0.0

- New version of the Neurotec.Biometrics.Gui [1782] library released.

7.3.2.3 Neurotec.Licensing

Version 3.1.0.0

- **ADD**: NLicense.IsComponentActivated method.
- **UPD**: Now uses Neurotec library version 3.1.

Version 3.0.0.0

- New version of the Neurotec.Licensing [1851] library released.

7.3.2.4 Neurotec.Surveillance Changes

Version 1.1.0.1

- **UPD**: Minor updates because of Neurotec changes.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Version 1.1.0.0

- **ADD**: New algorithm parameters DetectionMaxRollAngleDeviation, ExtractionEnrollmentTemplateSize and ExtractionIdentificationTemplateSize.
- **UPD**: Now uses Neurotec.Biometrics.Extractors library version 3.1.
- **UPD**: Now uses Neurotec.Biometrics.Matchers library version 3.1.
- **UPD**: Now uses Neurotec.Biometrics.Templates library version 3.1.
- **UPD**: Now uses Neurotec.Images library version 2.6.
- **UPD**: Now uses Neurotec library version 3.1.

Version 1.0.0.0

- New version of the Neurotec.Surveillance [1883] library released.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

8 Support

Neurotechnology provides customer support during the entire period, while the customer develops and uses his own system using our products. If you face problems using our products, have any questions or suggestions, you can reach us by email: support@neurotechnology.com.

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Index

.

.NET 56
.NET API Reference 1916
.NET Reference 1468

A

Accesing NDeviceManger 23
AccidentalWhorl enumeration member 1775
AChar enumeration member 1652
AddDevice 24
Adding custom device to NDeviceManager 21
Aligned enumeration member 1651
Amd enumeration member 1652
Amputation enumeration member 1775
Any enumeration member 1824, 1908
API Reference 60
Array enumeration member 1652
AutoFocusFailure enumeration member 1823

B

BadObject enumeration member 1769
BadObjectSequence enumeration member 1769
BadPosition enumeration member 1769
BadSize enumeration member 1769
BadSpeed enumeration member 1769
Beard enumeration member 1780
Bifurcation enumeration member 1774
BiometricDevice enumeration member 1824
Blink enumeration member 1780
Bmp Unit 1196
Bmp.h 1200
BmpLoadImageFromFile function 1197
BmpLoadImageFromHBitmap macro 1200
BmpLoadImageFromMemory function 1198
BmpLoadImageFromStream function 1198
BmpSaveImageToFile function 1199
BmpSaveImageToHBitmap macro 1200
BmpSaveImageToMemory function 1199
BmpSaveImageToStream function 1200
BodyOdor enumeration member 1771
Boolean enumeration member 1652
Both enumeration member 1772
BothEyePatch enumeration member 1780
Buffer enumeration member 1652
Byte enumeration member 1652

C

C API Reference 1913
C Reference 60
C++ (WxWidgets) 49
Camera enumeration member 1824
Camera Unit 1126
Camera.h 1137
CameraCopyParameters macro 1132
CameraGetCurrentFrame function 1127
CameraGetDevice function 1127
CameraGetId function 1128
CameraGetParameterA macro 1132
CameraGetParameterW macro 1132
CameraGetVideoFormat function 1128
CameraGetVideoFormats function 1129
CameraIsCapturing function 1129
CameraMan Unit 1138
CameraMan.h 1141
CameraManGetCamera function 1139
CameraManGetCameraById function 1139
CameraManGetCameraCount function 1140
CameraManGetCameraFromDevice function 1140
CameraManGetDeviceManager function 1140
CameraManInitialize function 1141
CameraManUninitialize function 1141
CAMERAP_AUTOMATIC_SETTINGS macro 1133
CAMERAP_EXPOSURE macro 1133
CAMERAP_EXPOSURE_MAX macro 1133
CAMERAP_EXPOSURE_MIN macro 1133
CAMERAP_GAIN macro 1134
CAMERAP_GAIN_MAX macro 1134
CAMERAP_GAIN_MIN macro 1134
CAMERAP_IP_CHANNEL_ID macro 1134

Copyright © 2009-2014 Neurotechnology. All rights reserved.

CAMERAP_IP_CHANNEL_NAME macro 1135
CAMERAP_IP_PASSWORD macro 1135
CAMERAP_IP_USERNAME macro 1135
CAMERAP_MIRROR_HORIZONTAL macro 1135
CAMERAP_MIRROR_VERTICAL macro 1135
CAMERAP_VIDEO_DROP_FRAMES macro 1135
CAMERAP_VIDEO_FILE_NAME macro 1136
CameraParams.h 1137
Cameras 16
CameraSetParameterA macro 1136
CameraSetParameterW macro 1136
CameraSetVideoFormat function 1130
CameraStartCapturing function 1130
CameraStopCapturing function 1130
CameraVideoFormat structure 1131
CameraVideoFormat_ structure 1131
CancelBiometricDevice 39
Canceled enumeration member 1769
CanUse 45
CaptureDevice enumeration member 1824
CaptureFScanner 41
CaptureIrisScanner 43
Centaur enumeration member 1652
CentralPocketLoop enumeration member 1775
Change Log 1912
Cisco Video Surveillance IP 4500 18
CleaningNeeded enumeration member 1769
Compact enumeration member 1779
Complex enumeration member 1652
Configuring on Linux and Mac OS X 21
Configuring on Microsoft Windows 20
ContinuePending enumeration member 1862
Custom enumeration member 1651
Cyrix enumeration member 1652

D

DarkGlasses enumeration member 1780
DateTime enumeration member 1652
Debug enumeration member 1651
Default enumeration member 1651
Device manager plugin lifecycle 22
DeviceAdded 44
DeviceLost 44
DeviceRemoved 45
Devices 55
Disabled enumeration member 1882
DistortingCondition enumeration member 1780
Dna enumeration member 1771
Double enumeration member 1652
DoubleLoop enumeration member 1775
Duplicate enumeration member 1882

E

Ear enumeration member 1771
EarShape enumeration member 1771
EightNeighbors enumeration member 1778
EightNeighborsWithIndexes enumeration member 1778
End enumeration member 1774
EndBiometricDeviceSequence 39
Enterprise Licensing 3
Error codes 17
ErrorSuppressedEventArgs class 1470
    about ErrorSuppressedEventArgs class 1470
    Error 1470
    ErrorSuppressedEventArgs 1470
Exe enumeration member 1651
EyesAway enumeration member 1780
EyesNotDetected enumeration member 1779

F

Face enumeration member 1771
Face Feature Points 46
FaceNotDetected enumeration member 1779
Faces 45
FacesDetected enumeration member 1908
FaceTooCloseToImageBorder enumeration member 1779
FacialFeatures enumeration member 1771
Features 2
Female enumeration member 1778
Finger enumeration member 1771
FingerGeometry enumeration member 1771
Fingerprint enumeration member 1771

Copyright © 2009-2014 Neurotechnology. All rights reserved.

FingerScanner enumeration member 1824
Foot enumeration member 1771
FootPrint enumeration member 1771
Format Specifications 685
FourNeighbors enumeration member 1778
FourNeighborsWithIndexes enumeration member 1778
Frowning enumeration member 1780
FScanner enumeration member 1824
Functionality expected by NDeviceManger 25
Functions for biometric devices 36
Functions for cameras 35
Functions for capture devices 32
Functions for fingerprint/palm scanners 40
Functions for iris scanners 42
Functions for microphones 36
FuseAlways enumeration member 1781

G

Gait enumeration member 1771
General functions 28
GeneralizationFailed enumeration member 1779
GetBiometricDeviceBiometricType 36
GetBiometricDeviceProductId 37
GetBiometricDeviceSpoofDetection 38
GetBiometricDeviceVendorId 37
GetCameraFrame 35
GetCaptureDeviceCurrentFormat 34
GetCaptureDeviceFormats 34
GetCaptureDeviceMediaType 33
GetDeviceDisplayNameA 29
GetDeviceDisplayNameW 29
GetDeviceIdA 28
GetDeviceIdW 28
GetDeviceMakeA 30
GetDeviceMakeW 30
GetDeviceModelA 30
GetDeviceModelW 31
GetDeviceParent 27
GetDeviceSerialNumberA 31
GetDeviceSerialNumberW 31
GetDeviceType 28
GetFScannerSupportedImpressionTypes 40
GetFScannerSupportedPositions 40
GetIrisScannerSupportedPositions 42
GetMicrophoneSoundSample 36
GetSupportedDeviceTypes 27
Glasses enumeration member 1780
Guid enumeration member 1652

H

HandGeometry enumeration member 1771
HasCurvature enumeration member 1774
HasG enumeration member 1774
HasQuality enumeration member 1774
Hat enumeration member 1780
HCamera 1131
High enumeration member 1781
HNImage 1276
HNImageFile 1298
HNImageFormat 1324
HNLExtractor 114
HNLMatcher 132
HNLRecord 140
HNlsMatcher 149
HNLTemplate 165
HNMatcher 183
HNMediaFormat 1336
HNMediaReader 1343
HNmediaSource 1350
HNPluginModule 634
HNTemplate 243
HNVideoWriter 1421
HNVideoWriterOptions 1421

I

IHead Unit 1201
IHead.h 1205
IHeadLoadImageFromFile function 1201
IHeadLoadImageFromMemory function 1202
IHeadLoadImageFromStream function 1203
IHeadSaveImageToFile function 1203
IHeadSaveImageToMemory function 1203

Copyright © 2009-2014 Neurotechnology. All rights reserved.

IHeadSaveImageToStream function 1204
Image 11
Image File 13
Image Format 12
Image Support 10
ImageCaptured enumeration member 1908
IncompatibleWithOtherPlugins enumeration member 1882
Int16 enumeration member 1652
Int32 enumeration member 1652
Int64 enumeration member 1652
Inteface reference 23
Intel enumeration member 1652
InterfaceTypeMismatch enumeration member 1882
InterfaceVersionMismatch enumeration member 1882
Introduction 1
InvalidInterface enumeration member 1882
InvalidModule enumeration member 1882
IP Cameras 17
Iris enumeration member 1771
IrisScanner enumeration member 1824
IsBiometricDeviceSpoofDetectionSupported 37
IsCaptureDeviceCapturing 33
IsDriverLoaded 24

J

Jpeg Unit 1205
Jpeg.h 1212
JPEG_2K_DEFAULT_RATIO macro 1221
JPEG_DEFAULT_QUALITY macro 1212
Jpeg2K Unit 1212
Jpeg2K.h 1221
Jpeg2KInfoGetProfile function 1213
Jpeg2KInfoGetRatio function 1214
Jpeg2KInfoSetProfile function 1214
Jpeg2KInfoSetRatio function 1214
Jpeg2KLoadImageFromFile function 1215
Jpeg2KLoadImageFromMemory function 1215
Jpeg2KLoadImageFromStream function 1216
jpeg2kpFaceLossless enumeration member 1220
jpeg2kpFaceLossy enumeration member 1220
jpeg2kpFingerprint1000Ppi enumeration member 1220
jpeg2kpNone enumeration member 1220
Jpeg2KProfile enumeration 1220
Jpeg2KProfile_ enumeration 1220
Jpeg2KSaveImageToFile function 1216
Jpeg2KSaveImageToFileWithProfile function 1217
Jpeg2KSaveImageToMemory function 1217
Jpeg2KSaveImageToMemoryWithProfile function 1218
Jpeg2KSaveImageToStream function 1219
Jpeg2KSaveImageToStreamWithProfile function 1219
JpegInfoGetQuality function 1206
JpegInfoIsLossless function 1206
JpegInfoSetLossless function 1206
JpegInfoSetQuality function 1207
JpegLoadImageFromFile function 1207
JpegLoadImageFromMemory function 1208
JpegLoadImageFromStream function 1208
JpegSaveImageToFile function 1209
JpegSaveImageToMemory function 1209
JpegSaveImageToStream function 1210

K

Keystroke enumeration member 1771
KeystrokeDynamics enumeration member 1771

L

Large enumeration member 1779
LatentImpression enumeration member 1773
LatentLift enumeration member 1773
LatentPalmImpression enumeration member 1773
LatentPalmLift enumeration member 1773
LatentPalmPhoto enumeration member 1773
LatentPalmTracing enumeration member 1773
LatentPhoto enumeration member 1773
LatentTracing enumeration member 1773
Left enumeration member 1770, 1772
LeftBackOfHand enumeration member 1770
LeftEyePatch enumeration member 1780
LeftFullPalm enumeration member 1776
LeftHypothenar enumeration member 1776
LeftIndex enumeration member 1776
LeftIndexFinger enumeration member 1776

Copyright © 2009-2014 Neurotechnology. All rights reserved.

LeftIndexMiddleFingers enumeration member 1776
LeftIndexMiddleRingFingers enumeration member 1776
LeftInterdigital enumeration member 1776
LeftLittle enumeration member 1776
LeftLittleFinger enumeration member 1770, 1776
LeftLowerPalm enumeration member 1776
LeftMiddle enumeration member 1776
LeftMiddleFinger enumeration member 1770, 1776
LeftMiddleRingFingers enumeration member 1776
LeftMiddleRingLittleFingers enumeration member 1776
LeftOther enumeration member 1776
LeftPalm enumeration member 1770
LeftPointerFinger enumeration member 1770
LeftRing enumeration member 1776
LeftRingFinger enumeration member 1770, 1776
LeftRingLittleFingers enumeration member 1776
LeftSlantLoop enumeration member 1775
LeftThenar enumeration member 1776
LeftThumb enumeration member 1770, 1776
LeftUpperPalm enumeration member 1776
LeftWrist enumeration member 1770
LeftWritersPalm enumeration member 1776
LensClosed enumeration member 1823
Lib enumeration member 1651
Licensing 3
LipMovement enumeration member 1771
LivenessCheckFailed enumeration member 1779
LiveScanContactless enumeration member 1773
LiveScanNonOpticalContactlessPlain enumeration member 1773
LiveScanNonOpticalContactlessRolled enumeration member 1773
LiveScanNonOpticalContactPlain enumeration member 1773
LiveScanNonOpticalContactRolled enumeration member 1773
LiveScanOpticalContactlessPlain enumeration member 1773
LiveScanOpticalContactlessRolled enumeration member 1773
LiveScanOpticalContactPlain enumeration member 1773
LiveScanOpticalContactRolled enumeration member 1773
LiveScanPalm enumeration member 1773
LiveScanPlain enumeration member 1773
LiveScanRolled enumeration member 1773
LiveScanVerticalSwipe enumeration member 1773
LoadError enumeration member 1882
LoseDevice 24
LosslessJpegSaveImageToFile function 1210
LosslessJpegSaveImageToMemory function 1211
LosslessJpegSaveImageToStream function 1211
Low enumeration member 1781
Low-Level Image Input-Output 13

M

Male enumeration member 1778
Matching Threshold and FAR/FRR 13
Medium enumeration member 1779, 1781
Microphone enumeration member 1824
MirrorUp enumeration member 1823
Mobotix DualNight M12 19
Moustache enumeration member 1780
MouthOpen enumeration member 1780
MultipleBiometrics enumeration member 1771

N

N_64 macro 379
N_ACHAR_ENCODING macro 397
N_ALIGN macro 379
N_ANSI_C macro 379
N_API macro 379
N_API_PTR_RET macro 379
N_AW macro 1011
N_BIG_ENDIAN macro 380
N_BIOMETRIC_QUALITY_FAILED macro 97
N_BIOMETRIC_QUALITY_MAX macro 97
N_BIOMETRIC_QUALITY_MIN macro 97
N_BIOMETRIC_QUALITY_UNKNOWN macro 98
N_BOOLEAN_FALSE_STRING macro 1011
N_BOOLEAN_FALSE_STRINGA macro 1011
N_BOOLEAN_FALSE_STRINGW macro 1012
N_BOOLEAN_TRUE_STRING macro 1012
N_BOOLEAN_TRUE_STRINGA macro 1012
N_BOOLEAN_TRUE_STRINGW macro 1012
N_BYTE_MAX macro 1012
N_BYTE_MIN macro 1013
N_CALL_CONV macro 380

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_CALLBACK macro 380
N_CALLBACK_AW macro 1013
N_CHAR_ENCODING macro 397
N_CPP macro 380
N_DATE_TIME_MAX macro 377
N_DATE_TIME_MIN macro 377
N_DEBUG macro 380
N_DECLARE_HANDLE macro 1011
N_DECLARE_HANDLE_TYPE macro 1013
N_DECLARE_MODULE macro 526
N_DECLARE_OBJECT_TYPE macro 1013
N_DECLARE_STATIC_OBJECT_TYPE macro 1013
N_DECLARE_TYPE macro 1014
N_DECLATE_PRIMITIVE macro 1014
N_DEPRECATED macro 380
N_DOUBLE_EPSILON macro 1014
N_DOUBLE_MAX macro 1014
N_DOUBLE_MIN macro 1014
N_E_ABANDONED_MUTEX macro 427
N_E_ARGUMENT macro 427
N_E_ARGUMENT_NULL macro 427
N_E_ARGUMENT_OUT_OF_RANGE macro 428
N_E_ARITHMETIC macro 428
N_E_BAD_IMAGE_FORMAT macro 428
N_E_CLR macro 428
N_E_COM macro 428
N_E_CORE macro 428
N_E_CPP macro 429
N_E_DIRECTORY_NOT_FOUND macro 430
N_E_DLL_NOT_FOUND macro 430
N_E_DRIVE_NOT_FOUND macro 430
N_E_END_OF_STREAM macro 429
N_E_ENTRY_POINT_NOT_FOUND macro 431
N_E_EXTERNAL macro 429
N_E_FAILED macro 429
N_E_FILE_FORMAT macro 432
N_E_FILE_LOAD macro 432
N_E_FILE_NOT_FOUND macro 432
N_E_FORMAT macro 429
N_E_INDEX_OUT_OF_RANGE macro 430
N_E_INVALID_CAST macro 433
N_E_INVALID_ENUM_ARGUMENT macro 433
N_E_INVALID_OPERATION macro 430
N_E_IO macro 430
N_E_JVM macro 434
N_E_MAC macro 434
N_E_NOT_ACTIVATED macro 431
N_E_NOT_IMPLEMENTED macro 431
N_E_NOT_SUPPORTED macro 431
N_E_NULL_REFERENCE macro 431
N_E_OUT_OF_MEMORY macro 432
N_E_OVERFLOW macro 432
N_E_PARAMETER macro 432
N_E_PARAMETER_READ_ONLY macro 433
N_E_PATH_TOO_LONG macro 434
N_E_SECURITY macro 435
N_E_SOCKET macro 435
N_E_SYS macro 433
N_E_TIMEOUT macro 435
N_E_WIN32 macro 433
N_EMPTY_STRING macro 1014
N_EMPTY_STRINGA macro 1015
N_EMPTY_STRINGW macro 1015
N_EVENT_RAISE_1 macro 452
N_EVENT_RAISE_2 macro 452
N_EVENT_RAISE_3 macro 453
N_EVENT_RAISE_4 macro 453
N_EVENT_RAISE_5 macro 454
N_EVENT_RAISE_6 macro 454
N_EVENT_RAISE_RAW macro 455
N_EVENT_SIZE macro 455
N_FAST_FLOAT macro 381
N_FIELD_AW macro 1015
N_FLOAT_EPSILON macro 1015
N_FLOAT_INFINITY_STRING macro 1015
N_FLOAT_INFINITY_STRINGA macro 1015
N_FLOAT_INFINITY_STRINGW macro 1016
N_FLOAT_MAX macro 1016
N_FLOAT_MIN macro 1016
N_FLOAT_NAN_STRING macro 1016
N_FLOAT_NAN_STRINGA macro 1016
N_FLOAT_NAN_STRINGW macro 1017

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_FUNC_AW macro 1017
N_GCC macro 381
N_INFINITE macro 1124
N_INT_MAX macro 1017
N_INT_MIN macro 1017
N_INT16_MAX macro 1017
N_INT16_MIN macro 1017
N_INT32_MAX macro 1018
N_INT32_MIN macro 1018
N_INT64_MAX macro 1018
N_INT64_MIN macro 1018
N_INT8_MAX macro 1018
N_INT8_MIN macro 1018
N_JOIN_SYMBOLS macro 1019
N_JOIN_SYMBOLS_ macro 1019
N_JOIN_SYMBOLS2 macro 1019
N_JOIN_SYMBOLS2_ macro 1019
N_LI_ACTIVATED_MAX_LENGTH macro 499
N_LI_COMPANY_MAX_LENGTH macro 500
N_LI_COPYRIGHT_MAX_LENGTH macro 500
N_LI_PRODUCT_MAX_LENGTH macro 500
N_LI_TITLE_MAX_LENGTH macro 500
N_LIB macro 381
N_LIKELY macro 381
N_LINUX macro 381
N_LONG_MAX macro 1019
N_LONG_MIN macro 1020
N_MAC macro 381
N_MACRO_AW macro 1020
N_MAX_FILE_NAME macro 590
N_MODULE_OF macro 526
N_MONITOR_SIZE macro 534
N_MSVC macro 382
N_NEW_LINE macro 729
N_NO_ANSI_FUNC macro 382
N_NO_INLINE macro 382
N_NO_INT_64 macro 382
N_NO_RETURN macro 382
N_NO_THROW macro 383
N_NO_UNICODE macro 383
N_OBJECT_REF_RET macro 561
N_OK macro 433
N_PACKED macro 383
N_PATH_ALT_DIRECTORY_SEPARATOR macro 590
N_PATH_DIRECTORY_SEPARATOR macro 590
N_PATH_IS_CASE_SENSITIVE macro 590
N_PATH_PATH_SEPARATOR macro 590
N_PATH_VOLUME_SEPARATOR macro 591
N_PC_TYPE_ID macro 576
N_PHRASE_ID_UNKNOWN macro 98
N_PTR_SIZE macro 1020
N_RETURN_ADDRESS macro 383
N_SBYTE_MAX macro 1020
N_SBYTE_MIN macro 1020
N_SHORT_MAX macro 1020
N_SHORT_MIN macro 1021
N_SINGLE_EPSILON macro 1021
N_SINGLE_MAX macro 1021
N_SINGLE_MIN macro 1021
N_SIZE_TYPE_MAX macro 1021
N_SIZE_TYPE_MIN macro 1021
N_SLOW_INTERLOCKED macro 492
N_SSIZE_TYPE_MAX macro 1022
N_SSIZE_TYPE_MIN macro 1022
N_STATIC_ASSERT macro 383
N_STRING_BUILDER_SIZE macro 804
N_STRING_SIZE macro 729
N_STRINGIZE macro 1022
N_STRINGIZE_ macro 1022
N_STRINGIZE_A macro 1022
N_STRINGIZE_W macro 1023
N_STRINGIZEA macro 1023
N_STRINGIZEW macro 1023
N_STRUCT_AW macro 1023
N_SYMBOL_AW macro 1023
N_T macro 1023
N_T_A macro 1024
N_T_W macro 1024
N_TA macro 1024
N_TIME_SPAN_MAX macro 866
N_TIME_SPAN_MIN macro 866
N_TIME_SPAN_TICKS_PER_DAY macro 867

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_TIME_SPAN_TICKS_PER_HOUR macro 867
N_TIME_SPAN_TICKS_PER_MILLISECOND macro 867
N_TIME_SPAN_TICKS_PER_MINUTE macro 867
N_TIME_SPAN_TICKS_PER_SECOND macro 867
N_TRACE macro 383
N_TW macro 1024
N_TYPE_AW macro 1024
N_TYPE_BOOL macro 576
N_TYPE_BYTE macro 576
N_TYPE_DOUBLE macro 576
N_TYPE_FLOAT macro 577
N_TYPE_INT macro 577
N_TYPE_LONG macro 577
N_TYPE_OF macro 1024
N_TYPE_SBYTE macro 577
N_TYPE_SHORT macro 577
N_TYPE_STRING macro 578
N_TYPE_UINT macro 578
N_TYPE_ULONG macro 578
N_TYPE_USHORT macro 578
N_UINT_MAX macro 1025
N_UINT_MIN macro 1025
N_UINT16_MAX macro 1025
N_UINT16_MIN macro 1025
N_UINT32_MAX macro 1025
N_UINT32_MIN macro 1026
N_UINT64_MAX macro 1026
N_UINT64_MIN macro 1026
N_UINT8_MAX macro 1026
N_UINT8_MIN macro 1026
N_ULONG_MAX macro 1026
N_ULONG_MIN macro 1027
N_UNICODE macro 384
N_UNLIKELY macro 384
N_UNREFERENCED_PARAMETER macro 1027
N_UNUSED_VARIABLE macro 1027
N_USHORT_MAX macro 1027
N_USHORT_MIN macro 1027
N_VAR_AW macro 1027
N_VIDEO_FORMAT_AI44 macro 1402
N_VIDEO_FORMAT_ARGB32 macro 1402
N_VIDEO_FORMAT_AVC1 macro 1402
N_VIDEO_FORMAT_AYUV macro 1402
N_VIDEO_FORMAT_DV25 macro 1403
N_VIDEO_FORMAT_DV50 macro 1403
N_VIDEO_FORMAT_DVC macro 1403
N_VIDEO_FORMAT_DVH1 macro 1403
N_VIDEO_FORMAT_DVHD macro 1403
N_VIDEO_FORMAT_DVSD macro 1404
N_VIDEO_FORMAT_DVSL macro 1404
N_VIDEO_FORMAT_DX50 macro 1404
N_VIDEO_FORMAT_H264 macro 1404
N_VIDEO_FORMAT_I420 macro 1404
N_VIDEO_FORMAT_IYUV macro 1404
N_VIDEO_FORMAT_L16 macro 1405
N_VIDEO_FORMAT_L8 macro 1405
N_VIDEO_FORMAT_M4S2 macro 1405
N_VIDEO_FORMAT_MJPG macro 1405
N_VIDEO_FORMAT_MP43 macro 1405
N_VIDEO_FORMAT_MP4S macro 1405
N_VIDEO_FORMAT_MP4V macro 1406
N_VIDEO_FORMAT_MPG1 macro 1406
N_VIDEO_FORMAT_MSS1 macro 1406
N_VIDEO_FORMAT_MSS2 macro 1406
N_VIDEO_FORMAT_NV11 macro 1406
N_VIDEO_FORMAT_NV12 macro 1407
N_VIDEO_FORMAT_P010 macro 1407
N_VIDEO_FORMAT_P016 macro 1407
N_VIDEO_FORMAT_P210 macro 1407
N_VIDEO_FORMAT_P216 macro 1407
N_VIDEO_FORMAT_RGB24 macro 1407
N_VIDEO_FORMAT_RGB32 macro 1408
N_VIDEO_FORMAT_RGB555 macro 1408
N_VIDEO_FORMAT_RGB565 macro 1408
N_VIDEO_FORMAT_RGB8 macro 1408
N_VIDEO_FORMAT_UNKNOWN macro 1408
N_VIDEO_FORMAT_UYVY macro 1408
N_VIDEO_FORMAT_V210 macro 1409
N_VIDEO_FORMAT_V216 macro 1409
N_VIDEO_FORMAT_V410 macro 1409
N_VIDEO_FORMAT_WMV1 macro 1409
N_VIDEO_FORMAT_WMV2 macro 1409

Copyright © 2009-2014 Neurotechnology. All rights reserved.

N_VIDEO_FORMAT_WMV3 macro 1410
N_VIDEO_FORMAT_WVC1 macro 1410
N_VIDEO_FORMAT_Y210 macro 1410
N_VIDEO_FORMAT_Y216 macro 1410
N_VIDEO_FORMAT_Y410 macro 1410
N_VIDEO_FORMAT_Y416 macro 1410
N_VIDEO_FORMAT_Y41P macro 1411
N_VIDEO_FORMAT_Y41T macro 1411
N_VIDEO_FORMAT_Y42T macro 1411
N_VIDEO_FORMAT_YUY2 macro 1411
N_VIDEO_FORMAT_YV12 macro 1411
N_VIDEO_FORMAT_YVU9 macro 1411
N_VIDEO_FORMAT_YVYU macro 1412
N_WCHAR_ENCODING macro 397
N_WCHAR_SIZE macro 1028
N_WINDOWS macro 384
naArray enumeration member 1002
naByRef enumeration member 1002
naCache enumeration member 1002
NAChar type 1004
NACharParse macro 1028
NACharParseChars macro 1028
NACharParseN function 920
NACharParseStrOrChars function 921
NACharParseStrOrChars macro 1029
NACharParseVN function 921
NACharParseVN macro 1029
NACharToChars function 921
NACharToChars macro 1029
NACharToString function 922
NACharToString macro 1029
NACharTryParse macro 1029
NACharTryParseChars macro 1030
NACharTryParseN function 922
NACharTryParseStrOrChars function 923
NACharTryParseStrOrChars macro 1030
NACharTryParseVN function 923
NACharTryParseVN macro 1030
naDefaultValue enumeration member 1002
naDeprecated enumeration member 1002
NAlignedFree function 503
NAlignedOffsetAlloc function 503
NAlignedOffsetCAlloc function 503
NAlignedOffsetReAlloc function 504
NAlloc function 504
naLocal enumeration member 1002
naMaxValue enumeration member 1002
Naming requirements 22
naMinValue enumeration member 1002
naNone enumeration member 1002
naNoRead enumeration member 1002
naNoWrite enumeration member 1002
naNullable enumeration member 1002
naOptional enumeration member 1002
naPromoteValue enumeration member 1002
naPublic enumeration member 1002
naRetValue enumeration member 1002
NArray class 1471
- about NArray class 1471
- FromArray 1475, 1476, 1477, 1478, 1479, 1480, 1481, 1482, 1483, 1484, 1485
- FromPointerArray 1485
- FromResultArray 1486
- GetEnumerator 1486
- GetValueAs 1486, 1487, 1488
- GetValueAsBoolean 1488, 1489
- GetValueAsByte 1489, 1490
- GetValueAsChar 1490
- GetValueAsDouble 1491
- GetValueAsInt16 1492
- GetValueAsInt32 1492, 1493
- GetValueAsInt64 1493, 1494
- GetValueAsIntPtr 1494, 1495
- GetValueAsObject 1495, 1496
- GetValueAsPointer 1496
- GetValueAsResult 1497
- GetValueAsSByte 1498
- GetValueAsSingle 1498, 1499
- GetValueAsString 1499, 1500
- GetValueAsUInt16 1500, 1501
- GetValueAsUInt32 1501, 1502
- GetValueAsUInt64 1502
- GetValueAsUIntPtr 1503

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Length 1519
NativeType 1519
ToArray 1504, 1505
ToBooleanArray 1506
ToByteArray 1506, 1507
ToCharArray 1507, 1508
ToDoubleArray 1508
ToInt16Array 1509
ToInt32Array 1509, 1510
ToInt64Array 1510
ToIntPtrArray 1511
ToObjectArray 1512
ToPointerArray 1512, 1513
ToResultArray 1513, 1514
ToSByteArray 1514
ToSingleArray 1515
ToStringArray 1515, 1516
ToUInt16Array 1516
ToUInt32Array 1517
ToUInt64Array 1518
ToUIntPtrArray 1518, 1519
NArray Unit 259
NArray.h 289
NArrayCopyTo function 261
NArrayCopyToBooleanArray function 262
NArrayCopyToByteArray function 262
NArrayCopyToCharArray function 262
NArrayCopyToCharArray macro 288
NArrayCopyToDoubleArray function 263
NArrayCopyToInt16Array function 263
NArrayCopyToInt32Array function 264
NArrayCopyToInt64Array function 264
NArrayCopyToObjectArray function 264
NArrayCopyToObjectArrayP function 265
NArrayCopyToP function 265
NArrayCopyToPointerArray function 266
NArrayCopyToResultArray function 266
NArrayCopyToSByteArray function 267
NArrayCopyToSingleArray function 267
NArrayCopyToSizeTypeArray function 267
NArrayCopyToSSizeTypeArray function 268
NArrayCopyToStringArray function 268
NArrayCopyToUInt16Array function 269
NArrayCopyToUInt32Array function 269
NArrayCopyToUInt64Array function 269
NArrayCreate function 270
NArrayCreateCustom function 270
NArrayCreateFromBooleanArray function 271
NArrayCreateFromByteArray function 271
NArrayCreateFromCharArray function 272
NArrayCreateFromCharArray macro 289
NArrayCreateFromDoubleArray function 272
NArrayCreateFromInt16Array function 273
NArrayCreateFromInt32Array function 273
NArrayCreateFromInt64Array function 273
NArrayCreateFromObjectArray function 274
NArrayCreateFromObjectArrayP function 274
NArrayCreateFromPointerArray function 275
NArrayCreateFromResultArray function 275
NArrayCreateFromSByteArray function 275
NArrayCreateFromSingleArray function 276
NArrayCreateFromSizeTypeArray function 276
NArrayCreateFromSSizeTypeArray function 277
NArrayCreateFromStringArray function 277
NArrayCreateFromStringArray macro 289
NArrayCreateFromUInt16Array function 277
NArrayCreateFromUInt32Array function 278
NArrayCreateFromUInt64Array function 278
NArrayCreateP function 279
NArrayGetLength function 279
NArrayGetValue function 279
NArrayGetValueAs function 280
NArrayGetValueAsBoolean function 280
NArrayGetValueAsByte function 281
NArrayGetValueAsChar function 281
NArrayGetValueAsChar macro 289
NArrayGetValueAsDouble function 281
NArrayGetValueAsInt16 function 282
NArrayGetValueAsInt32 function 282
NArrayGetValueAsInt64 function 283
NArrayGetValueAsObject function 283
NArrayGetValueAsObjectP function 283

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NArrayGetValueAsP function 284
NArrayGetValueAsPointer function 284
NArrayGetValueAsResult function 285
NArrayGetValueAsSByte function 285
NArrayGetValueAsSingle function 285
NArrayGetValueAsSizeType function 286
NArrayGetValueAsSSizeType function 286
NArrayGetValueAsString function 287
NArrayGetValueAsUInt16 function 287
NArrayGetValueAsUInt32 function 287
NArrayGetValueAsUInt64 function 288
naSet enumeration member 1002
naSignNeutral enumeration member 1002
naSingleValue enumeration member 1002
naStatic enumeration member 1002
naStdValues enumeration member 1002
naStdValuesExclusive enumeration member 1002
NationalSemiconductor enumeration member 1652
NAttributes enumeration 1002
NAttributes_ enumeration 1002
NBinaryReader Unit 291
NBinaryReaderCreate function 293
NBinaryReaderCreateWithEncoding function 293
NBinaryReaderGetBaseStream function 293
NBinaryReaderGetByteOrder function 294
NBinaryReaderGetNull function 294
NBinaryReaderGetPosition function 295
NBinaryReaderPeekChar function 295
NBinaryReaderPeekChar macro 303
NBinaryReaderRead function 295
NBinaryReaderRead macro 303
NBinaryReaderReadBoolean function 296
NBinaryReaderReadByte function 296
NBinaryReaderReadBytesDst function 296
NBinaryReaderReadBytesDstN function 297
NBinaryReaderReadBytesN function 297
NBinaryReaderReadChar function 298
NBinaryReaderReadChar macro 303
NBinaryReaderReadChars function 298
NBinaryReaderReadChars macro 303
NBinaryReaderReadCharsDst function 298
NBinaryReaderReadCharsDst macro 303
NBinaryReaderReadDouble function 299
NBinaryReaderReadInt16 function 299
NBinaryReaderReadInt32 function 300
NBinaryReaderReadInt64 function 300
NBinaryReaderReadSByte function 300
NBinaryReaderReadSingle function 301
NBinaryReaderReadUInt16 function 301
NBinaryReaderReadUInt32 function 301
NBinaryReaderReadUInt64 function 302
NBinaryReaderSeek function 302
NBinaryReaderSetPosition function 302
NBinaryWriter Unit 303
NBinaryWriterCreate function 305
NBinaryWriterCreateWithEncoding function 305
NBinaryWriterFlush function 305
NBinaryWriterGetBaseStream function 306
NBinaryWriterGetByteOrder function 306
NBinaryWriterGetNull function 306
NBinaryWriterGetPosition function 307
NBinaryWriterSeek function 307
NBinaryWriterSetPosition function 307
NBinaryWriterWriteBoolean function 308
NBinaryWriterWriteByte function 308
NBinaryWriterWriteBytes function 308
NBinaryWriterWriteBytesN function 309
NBinaryWriterWriteChar function 309
NBinaryWriterWriteChar macro 313
NBinaryWriterWriteChars function 309
NBinaryWriterWriteChars macro 313
NBinaryWriterWriteDouble function 310
NBinaryWriterWriteInt16 function 310
NBinaryWriterWriteInt32 function 310
NBinaryWriterWriteInt64 function 311
NBinaryWriterWriteSByte function 311
NBinaryWriterWriteSingle function 311
NBinaryWriterWriteUInt16 function 312
NBinaryWriterWriteUInt32 function 312
NBinaryWriterWriteUInt64 function 312
NBiometricAngleFromRadians function 65
NBiometricAngleToRadians function 65

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBiometricAngleToString function 65
NBiometricAngleToString macro 98
NBiometricAttributes class 1655
about NBiometricAttributes class 1655
BiometricType 1658
NativeType 1658
Quality 1658
Status 1658
NBiometricDevice class 1792
about NBiometricDevice class 1792
BiometricType 1796
Cancel 1796
EndSequence 1796
IsSpoofDetectionSupported 1796
NativeType 1797
ProductId 1797
SpoofDetection 1797
StartSequence 1796
VendorId 1797
NBiometricDevice Unit 1142
NBiometricDeviceCancel function 1142
NBiometricDeviceEndSequence function 1142
NBiometricDeviceGetBiometricType function 1143
NBiometricDeviceGetProductId function 1143
NBiometricDeviceGetSpoofDetection function 1144
NBiometricDeviceGetVendorId function 1144
NBiometricDeviceIsSpoofDetectionSupported function 1144
NBiometricDeviceSetSpoofDetection function 1145
NBiometricDeviceStartSequence function 1145
NBiometricQualityIsValid function 66
NBiometricQualityToString function 66
NBiometricQualityToString macro 98
NBiometrics class 1658
about NBiometrics class 1658
DllName 1659
NativeModule 1659
NBiometrics Library 60
NBiometricStatus enumeration 86
NBiometricStatus_ enumeration 86
NBiometricStatusIsFinal function 66
NBiometricStatusIsValid function 67
NBiometricSubtype enumeration 88
NBiometricSubtype_ enumeration 88
NBiometricSubtypeIsValid function 67
NBiometricType enumeration 89
NBiometricType_ enumeration 89
NBiometricTypeIsValid function 67
NBiometricTypes class 1659
about NBiometricTypes class 1659
AngleFromRadians 1662
AngleToRadians 1662
AngleToString 1662, 1663
GetPositionAvailableParts 1663
IsBiometricStatusFinal 1664
IsBiometricStatusValid 1664
IsBiometricSubtypeValid 1664
IsBiometricTypeValid 1664
IsImpressionTypeCompatibleWith 1665
IsImpressionTypeContact 1665
IsImpressionTypeContactless 1665
IsImpressionTypeFinger 1665
IsImpressionTypeGeneric 1666
IsImpressionTypeLatent 1666
IsImpressionTypeLiveScan 1666
IsImpressionTypeNonliveScan 1666
IsImpressionTypeNonOptical 1667
IsImpressionTypeOneOf 1667
IsImpressionTypeOptical 1667
IsImpressionTypePalm 1667
IsImpressionTypePlain 1668
IsImpressionTypeRolled 1668
IsImpressionTypeSwipe 1668
IsImpressionTypeValid 1668
IsImpressionTypeValidFinger 1669
IsImpressionTypeValidPalm 1669
IsPositionCompatibleWith 1669, 1670
IsPositionFinger 1670
IsPositionFourFingers 1670
IsPositionKnown 1670, 1671
IsPositionLeft 1671
IsPositionLeftAndRight 1671
IsPositionOneOf 1672

Copyright © 2009-2014 Neurotechnology. All rights reserved.

IsPositionPalm 1672
IsPositionRight 1673
IsPositionRightAndLeft 1673
IsPositionSingleEye 1673
IsPositionSingleFinger 1674
IsPositionTheEye 1674
IsPositionTheFinger 1674
IsPositionThePalm 1674
IsPositionThreeFingers 1675
IsPositionTwoEyes 1675
IsPositionTwoFingers 1675
IsPositionValid 1675, 1676
IsPositionValidFinger 1676
IsPositionValidPalm 1676
IsPositionValidTheEye 1676
IsPositionValidTheFinger 1677
IsQualityValid 1677
NativeType 1678
NBiometricStatusNativeType 1678
NBiometricSubtypeNativeType 1678
NBiometricTypeNativeType 1678
NEPositionNativeType 1678
NFImpressionTypeNativeType 1679
NFMinutiaFormatNativeType 1679
NFMinutiaTypeNativeType 1679
NFPatternClassNativeType 1679
NFPositionNativeType 1679
NFRidgeCountsTypeNativeType 1679
NGenderNativeType 1679
NLExpressionNativeType 1679
NLPropertiesNativeType 1679
PhraseIdUnknown 1661
QualityFailed 1661
QualityMax 1661
QualityMin 1662
QualityToString 1677
QualityUnknown 1662
ToImpressionTypeFinger 1678
NBiometricTypes Unit 61
NBiometricTypes.h 100
nboBigEndian enumeration member 496
nboLittleEndian enumeration member 496
NBool type 1004
NBoolean type 1004
NBooleanParse macro 1030
NBooleanParseChars macro 1031
NBooleanParseN function 924
NBooleanParseStrOrChars function 924
NBooleanParseStrOrChars macro 1031
NBooleanParseVN function 924
NBooleanParseVN macro 1031
NBooleanToChars function 925
NBooleanToChars macro 1032
NBooleanToString function 925
NBooleanToString macro 1032
NBooleanTryParse macro 1032
NBooleanTryParseChars macro 1032
NBooleanTryParseN function 926
NBooleanTryParseStrOrChars function 926
NBooleanTryParseStrOrChars macro 1033
NBooleanTryParseVN function 927
NBooleanTryParseVN macro 1033
nbsBadObject enumeration member 86
nbsBadObjectSequence enumeration member 86
nbsBadPosition enumeration member 86
nbsBadSize enumeration member 86
nbsBadSpeed enumeration member 86
nbsCanceled enumeration member 86
nbsCleaningNeeded enumeration member 86
nbsNone enumeration member 86
nbsObjectMissing enumeration member 86
nbsObjectNotFound enumeration member 86
nbsObjectsNotRemoved enumeration member 86
nbsOk enumeration member 86
nbsSpoofDetected enumeration member 86
nbsTimeout enumeration member 86
nbstLeft enumeration member 88
nbstLeftBackOfHand enumeration member 88
nbstLeftLittleFinger enumeration member 88
nbstLeftMiddleFinger enumeration member 88
nbstLeftPalm enumeration member 88
nbstLeftPointerFinger enumeration member 88

Copyright © 2009-2014 Neurotechnology. All rights reserved.

nbstLeftRingFinger enumeration member 88
nbstLeftThumb enumeration member 88
nbstLeftWrist enumeration member 88
nbstNone enumeration member 88
nbsTooClose enumeration member 86
nbsTooEast enumeration member 86
nbsTooFar enumeration member 86
nbsTooFast enumeration member 86
nbsTooFewObjects enumeration member 86
nbsTooHard enumeration member 86
nbsTooLong enumeration member 86
nbsTooManyObjects enumeration member 86
nbsTooNarrow enumeration member 86
nbsTooNorth enumeration member 86
nbsTooShort enumeration member 86
nbsTooSkewed enumeration member 86
nbsTooSlow enumeration member 86
nbsTooSoft enumeration member 86
nbsTooSouth enumeration member 86
nbsTooWest enumeration member 86
nbsTooWide enumeration member 86
nbstRight enumeration member 88
nbstRightBackOfHand enumeration member 88
nbstRightLittleFinger enumeration member 88
nbstRightMiddleFinger enumeration member 88
nbstRightPalm enumeration member 88
nbstRightPointerFinger enumeration member 88
nbstRightRingFinger enumeration member 88
nbstRightThumb enumeration member 88
nbstRightWrist enumeration member 88
nbsWrongDirection enumeration member 86
nbtBodyOdor enumeration member 89
nbtDna enumeration member 89
nbtEar enumeration member 89
nbtEarShape enumeration member 89
nbtFace enumeration member 89
nbtFacialFeatures enumeration member 89
nbtFinger enumeration member 89
nbtFingerGeometry enumeration member 89
nbtFingerprint enumeration member 89
nbtFoot enumeration member 89
nbtFootPrint enumeration member 89
nbtGait enumeration member 89
nbtHandGeometry enumeration member 89
nbtIris enumeration member 89
nbtKeystroke enumeration member 89
nbtKeystrokeDynamics enumeration member 89
nbtLipMovement enumeration member 89
nbtMultipleBiometrics enumeration member 89
nbtNone enumeration member 89
nbtPalm enumeration member 89
nbtPalmGeometry enumeration member 89
nbtPalmPrint enumeration member 89
nbtRetina enumeration member 89
nbtScent enumeration member 89
nbtSignatureDynamics enumeration member 89
nbtSignatureOrSign enumeration member 89
nbtThermalFace enumeration member 89
nbtThermalFaceImage enumeration member 89
nbtThermalHand enumeration member 89
nbtThermalHandImage enumeration member 89
nbtVein enumeration member 89
nbtVeinPattern enumeration member 89
nbtVoice enumeration member 89
NBuffer class 1825
- about NBuffer class 1825
- Copy 1832
- CopyFrom 1833
- CopyTo 1834, 1835, 1836
- Empty 1837
- LongSize 1837
- NativeSize 1837
- NativeType 1837
- NBuffer 1828, 1829, 1830, 1831
- Ptr 1837
- Size 1837
- ToArray 1836
- WriteTo 1836

NBuffer Unit 313
NBufferCopy function 314
NBufferCopyFromPtr function 314
NBufferCopyTo function 315

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NBufferCopyToPtr function 315
NBufferCopyToPtrRange function 315
NBufferCreate function 316
NBufferCreateFromBuffer function 316
NBufferCreateFromConstPtr macro 320
NBufferCreateFromPtr function 317
NBufferCreateFromPtrEx function 317
NBufferCreateFromPtrWithFree function 317
NBufferCreateFromPtrWithFreeProc function 318
NBufferedStream Unit 320
NBufferedStreamCreate function 320
NBufferedStreamCreateWithBufferSize function 321
NBufferGetEmpty function 318
NBufferGetPtr function 319
NBufferGetSize function 319
NBufferToPtr function 319
NByte type 1005
NByteOrder enumeration 496
NByteOrder_ enumeration 496
NByteOrderGetSystem function 493
NByteOrderIsReverse function 494
NByteOrderIsValid function 494
NCallback Unit 321
NCallback.h 332
NCallbackClone function 322
NCallbackCreate macro 325
NCallbackCreateCustom macro 326
NCallbackCreateCustomRaw function 322
NCallbackCreateRaw function 323
NCallbackEquals function 323
NCallbackFree function 324
NCallbackGetHashCode function 324
NCallbackGetParam function 324
NCallbackGetProc macro 327
NCallbackGetProcRaw function 325
NCallbackInvoke0 macro 327
NCallbackInvoke1 macro 327
NCallbackInvoke2 macro 328
NCallbackInvoke3 macro 328
NCallbackInvoke4 macro 328
NCallbackInvoke5 macro 329
NCallbackInvoke6 macro 329
NCallbackInvoke7 macro 330
NCallbackInvoke8 macro 330
NCallbackInvoke9 macro 331
NCallbackInvokeRaw macro 331
NCallbackSet function 325
NCAlloc function 505
NCallStackEntry_ structure 427
NCallStackEntryCopy function 403
NCallStackEntryCreate function 403
NCallStackEntryCreate macro 435
NCallStackEntryDispose function 404
NCallStackEntrySet function 404
NCamera class 1797
about NCamera class 1797
CaptureStill 1801
Focus 1801
FocusRegion 1802
GetCurrentStillFormat 1801
GetFrame 1801
GetStillFormats 1802
IsFocusRegionSupported 1802
IsFocusSupported 1802
IsStillCaptureSupported 1803
NativeType 1803
NCameraStatusNativeType 1803
ResetFocus 1802
SetCurrentStillFormat 1802
NCamera Unit 1145
NCameraAddIsCapturingChangedCallback macro 1153
NCameraAddStillCaptured function 1146
NCameraAddStillCapturedCallback function 1147
NCameraCaptureStill function 1147
NCameraFocus function 1148
NCameraGetCurrentStillFormat function 1148
NCameraGetFocusRegion function 1148
NCameraGetFrame function 1149
NCameraGetStillFormats function 1149
NCameraIsCapturing macro 1153
NCameraIsFocusRegionSupported function 1149
NCameraIsFocusSupported function 1150

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCameraIsStillCaptureSupported function 1150
NCameraRemoveIsCapturingChangedCallback macro 1154
NCameraRemoveStillCaptured function 1151
NCameraRemoveStillCapturedCallback function 1151
NCameraResetFocus function 1151
NCameraSetCurrentStillFormat function 1152
NCameraSetFocusRegion function 1152
NCameraStartCapturing macro 1154
NCameraStatus enumeration 1152
NCameraStatus_ enumeration 1152
NCameraStillCapturedEventArgs class 1803
about NCameraStillCapturedEventArgs class 1803
Id 1804
NCameraStillCapturedEventArgs 1803
Properties 1804
Stream 1804
NCameraStopCapturing macro 1154
NCaptureDevice class 1804
about NCaptureDevice class 1804
GetCurrentFormat 1807
GetFormats 1807
IsCapturing 1808
IsCapturingChanged 1808
MediaType 1808
NativeType 1808
SetCurrentFormat 1807
StartCapturing 1808
StopCapturing 1808
NCaptureDevice Unit 1155
NCaptureDeviceAddIsCapturingChanged function 1155
NCaptureDeviceAddIsCapturingChangedCallback function 1155
NCaptureDeviceGetCurrentFormat function 1156
NCaptureDeviceGetFormats function 1156
NCaptureDeviceGetMediaType function 1157
NCaptureDeviceIsCapturing function 1157
NCaptureDeviceRemoveIsCapturingChanged function 1157
NCaptureDeviceRemoveIsCapturingChangedCallback function 1158
NCaptureDeviceSetCurrentFormat function 1158
NCaptureDeviceStartCapturing function 1158
NCaptureDeviceStopCapturing function 1159
NChar type 1005
NCharFromBinDigit function 927
NCharFromBinDigit macro 1033
NCharFromDigit function 927
NCharFromDigit macro 1033
NCharFromHexDigit function 928
NCharFromHexDigit macro 1033
NCharFromOctDigit function 928
NCharFromOctDigit macro 1034
NCharIsAscii function 928
NCharIsAscii macro 1034
NCharIsBinDigit function 929
NCharIsBinDigit macro 1034
NCharIsDigit function 929
NCharIsDigit macro 1034
NCharIsHexDigit function 929
NCharIsHexDigit macro 1034
NCharIsLetter function 930
NCharIsLetter macro 1034
NCharIsLower function 930
NCharIsLower macro 1035
NCharIsOctDigit function 930
NCharIsOctDigit macro 1035
NCharIsUpper function 930
NCharIsUpper macro 1035
NCharIsWhiteSpace function 931
NCharIsWhiteSpace macro 1035
NCharParse macro 1035
NCharParseChars macro 1036
NCharParseN function 931
NCharParseStrOrChars function 931
NCharParseVN function 932
NCharToBinDigit function 932
NCharToBinDigit macro 1036
NCharToChars function 933
NCharToDigit function 933
NCharToDigit macro 1036
NCharToHexDigit function 933
NCharToHexDigit macro 1036
NCharToLower function 934
NCharToLower macro 1037

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NCharToOctDigit function 934
NCharToOctDigit macro 1037
NCharToString function 934
NCharToUpper function 935
NCharToUpper macro 1037
NCharTryParse macro 1037
NCharTryParseChars macro 1038
NCharTryParseN function 935
NCharTryParseStrOrChars function 935
NCharTryParseVN function 936
NClear macro 509
NCompare function 505
NComplex structure 1519
about NComplex structure 1519
Imaginary 1521
ImaginaryOne 1520
NativeType 1521
NComplex 1520
One 1520
Real 1521
ToString 1521
Zero 1520
NComplex type 1005
NComplex_ structure 1001
NComplexToString function 936
NConsole Unit 333
NConsoleGetError function 335
NConsoleGetIn function 335
NConsoleGetOut function 335
NConsoleReadChar function 336
NConsoleReadChar macro 350
NConsoleReadLine function 336
NConsoleSetError function 336
NConsoleSetIn function 336
NConsoleSetOut function 337
NConsoleWrite function 337
NConsoleWrite macro 350
NConsoleWriteBoolean function 337
NConsoleWriteBooleanLine function 338
NConsoleWriteChar function 338
NConsoleWriteChar macro 350
NConsoleWriteCharLine function 338
NConsoleWriteCharLine macro 350
NConsoleWriteChars function 339
NConsoleWriteChars macro 350
NConsoleWriteCharsLine function 339
NConsoleWriteCharsLine macro 350
NConsoleWriteDouble function 339
NConsoleWriteDoubleLine function 340
NConsoleWriteEmptyLine function 340
NConsoleWriteFormat function 340
NConsoleWriteFormat macro 351
NConsoleWriteFormatLine function 341
NConsoleWriteFormatLine macro 351
NConsoleWriteFormatLineVA function 341
NConsoleWriteFormatLineVA macro 351
NConsoleWriteFormatVA function 341
NConsoleWriteFormatVA macro 351
NConsoleWriteInt32 function 342
NConsoleWriteInt32Line function 342
NConsoleWriteInt64 function 342
NConsoleWriteInt64Line function 343
NConsoleWriteLine function 343
NConsoleWriteLine macro 351
NConsoleWriteObject function 343
NConsoleWriteObjectLine function 343
NConsoleWritePointer function 344
NConsoleWritePointerLine function 344
NConsoleWriteResult function 344
NConsoleWriteResultLine function 345
NConsoleWriteSingle function 345
NConsoleWriteSingleLine function 345
NConsoleWriteSizeType function 346
NConsoleWriteSizeTypeLine function 346
NConsoleWriteSSizeType function 346
NConsoleWriteSSizeTypeLine function 347
NConsoleWriteUInt32 function 347
NConsoleWriteUInt32Line function 347
NConsoleWriteUInt64 function 347
NConsoleWriteUInt64Line function 348
NConsoleWriteValue function 348
NConsoleWriteValueLine function 348

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NConsoleWriteValueLineP function 349
NConsoleWriteValueP function 349
NCopy function 506
NCore class 1521
about NCore class 1521
AlignedAlloc 1522, 1523, 1524
AlignedCAlloc 1524, 1525, 1526
AlignedFree 1526
AlignedReAlloc 1526, 1527, 1528
Alloc 1528, 1529
CAlloc 1529
Clear 1530
Compare 1530, 1531
Copy 1531, 1532
DllName 1522
ErrorSuppressed 1536
Fill 1532, 1533
Free 1533
Move 1534
NativeModule 1536
ReAlloc 1535
Shutdown 1535
ShutdownThread 1535
NCore Library 257, 1913
NCore Unit 352
NCore.h 356
NCoreAddErrorSuppressed function 353
NCoreAddErrorSuppressedCallback function 352
NCoreGetCallStack function 353
NCoreGetInfo function 353
NCoreModuleOf 355
NCoreOnExit macro 356
NCoreOnExitEx function 354
NCoreOnStart function 354
NCoreOnThreadExit function 354
NCoreOnThreadExitEx macro 356
NCoreOnThreadStart function 354
NCoreRemoveErrorSuppressed function 355
NCoreRemoveErrorSuppressedCallback function 355
ncsAutoFocusFailure enumeration member 1152
ncsLensClosed enumeration member 1152
ncsMirrorUp enumeration member 1152
ncsNoLens enumeration member 1152
ncsNone enumeration member 1152
ncsOk enumeration member 1152
ncsSensorCleaning enumeration member 1152
ncsSilentOperation enumeration member 1152
ncsStroboCharge enumeration member 1152
NCustomParameterDescriptor Unit 357
NCustomParameterDescriptor.h 358
NCustomParameterDescriptorCreate function 357
NCustomParameterDescriptorGetData function 358
NCustomPropertyDescriptor Unit 358
NCustomPropertyDescriptor.h 359
NCustomPropertyDescriptorGetData function 359
NCustomStream Unit 359
NCustomStreamCreate function 360
NDateTime Unit 361
NDateTime_ type 377
NDateTimeAddDays function 362
NDateTimeAddHours function 363
NDateTimeAddMilliseconds function 363
NDateTimeAddMinutes function 363
NDateTimeAddMonths function 364
NDateTimeAddSeconds function 364
NDateTimeAddTicks function 364
NDateTimeAddTimeSpan function 365
NDateTimeAddYears function 365
NDateTimeCompare function 366
NDateTimeCreate function 366
NDateTimeCreateDate function 366
NDateTimeCreateFromTicks function 367
NDateTimeDaysInMonth function 367
NDateTimeDecode function 367
NDateTimeGetDate function 368
NDateTimeGetDay function 368
NDateTimeGetDayOfWeek function 369
NDateTimeGetDayOfYear function 369
NDateTimeGetHour function 369
NDateTimeGetMillisecond function 370
NDateTimeGetMinute function 370
NDateTimeGetMonth function 370

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDateTimeGetNow function 371
NDateTimeGetSecond function 371
NDateTimeGetTicks function 371
NDateTimeGetTimeOfDay function 372
NDateTimeGetToday function 372
NDateTimeGetUtcNow function 372
NDateTimeGetYear function 373
NDateTimeIsDateTimeValid function 373
NDateTimeIsDateValid function 373
NDateTimeIsLeapYear function 374
NDateTimeIsValid function 374
NDateTimeSubtract function 374
NDateTimeSubtractTimeSpan function 375
NDateTimeToLocalTime function 375
NDateTimeToString function 375
NDateTimeToString macro 377
NDateTimeToUniversalTime function 376
NDayOfWeek enumeration 376
NDayOfWeek_ enumeration 376
NDefs Unit 378
NDescriptor Unit 384
NDescriptor.h 385
NDescriptorGetAttributes function 384
NDescriptorGetName function 385
NDevice class 1809
about NDevice class 1809
Children 1812
DeviceType 1812
DisplayName 1812
Id 1812
IsAvailable 1812
IsDisconnectable 1812
IsPrivate 1813
Make 1813
Model 1813
NativeType 1813
NDeviceTypeNativeType 1813
Owner 1813
Parent 1813
Plugin 1814
SerialNumber 1814
NDevice Unit 1159
NDevice.ChildCollection class 1811
about NDevice.ChildCollection class 1811
NDevice.h 1167
NDeviceGetChild function 1160
NDeviceGetChildCount function 1160
NDeviceGetDeviceType function 1161
NDeviceGetDisplayName function 1161
NDeviceGetDisplayName macro 1166
NDeviceGetId function 1162
NDeviceGetId macro 1166
NDeviceGetMake function 1162
NDeviceGetMake macro 1166
NDeviceGetModel function 1162
NDeviceGetModel macro 1167
NDeviceGetParent function 1163
NDeviceGetPlugin function 1163
NDeviceGetSerialNumber function 1163
NDeviceGetSerialNumber macro 1167
NDeviceIsAvailable function 1164
NDeviceIsDisconnectable function 1164
NDeviceIsPrivate function 1165
NDeviceManager class 1814
about NDeviceManager class 1814
AutoUpdate 1820
ConnectToDevice 1819
DeviceAdded 1821
DeviceRemoved 1821
Devices 1821
DevicesChanged 1821
DevicesChanging 1822
DevicesRefreshed 1822
DeviceTypes 1821
DisconnectFromDevice 1819
GetConnectToDeviceParameters 1820
IsConnectToDeviceSupported 1820
NativeType 1821
NDeviceManager 1817, 1818
PluginManager 1821
Refresh 1820
NDeviceManager Unit 1168

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NDeviceManager.DeviceCollection class 1818
about NDeviceManager.DeviceCollection class 1818
IndexOf 1818
this 1819
NDeviceManager.h 1177
NDeviceManagerAddDeviceAddedCallback function 1169
NDeviceManagerAddDeviceRemovedCallback function 1169
NDeviceManagerAddDevicesChangedCallback function 1170
NDeviceManagerAddDevicesChangingCallback function 1170
NDeviceManagerAddDevicesRefreshedCallback function 1170
NDeviceManagerCreate function 1171
NDeviceManagerDeviceEventArgs class 1822
about NDeviceManagerDeviceEventArgs class 1822
Device 1822
NDeviceManagerDeviceEventArgs 1822
NDeviceManagerGetAutoUpdate function 1172
NDeviceManagerGetDevice function 1172
NDeviceManagerGetDeviceById function 1172
NDeviceManagerGetDeviceById macro 1176
NDeviceManagerGetDeviceCount function 1173
NDeviceManagerGetDeviceTypes function 1173
NDeviceManagerGetInfoA macro 1177
NDeviceManagerGetInfoW macro 1177
NDeviceManagerGetPluginManager function 1174
NDeviceManagerRefresh function 1174
NDeviceManagerRemoveDeviceAddedCallback function 1174
NDeviceManagerRemoveDeviceRemovedCallback function 1175
NDeviceManagerRemoveDevicesChangedCallback function 1175
NDeviceManagerRemoveDevicesChangingCallback function 1176
NDeviceManagerRemoveDevicesRefreshedCallback function 1176
NDevices class 1822
about NDevices class 1822
DllName 1823
NativeModule 1823
NDevices Library 1124
NDevices Unit 1178
NDevicesGetInfo function 1178
NDeviceType enumeration 1165
NDeviceType_ enumeration 1165
NDirectory Unit 385
NDirectoryExists function 386
NDirectoryExists macro 386
NDisposable class 1536
about NDisposable class 1536
Dispose 1536
IsDisposed 1536
NdmMedia plugin 19
NdmSample - Adding support of custom device to NDeviceManager 55
NDouble type 1005
NDoubleGetNaN function 937
NDoubleGetNegativeInfinity function 937
NDoubleGetPositiveInfinity function 937
NDoubleIsInfinity function 937
NDoubleIsNaN function 938
NDoubleIsNegativeInfinity function 938
NDoubleIsPositiveInfinity function 938
NDoubleParse macro 1038
NDoubleParseChars macro 1038
NDoubleParseN function 939
NDoubleParseStrOrChars function 939
NDoubleParseStrOrChars macro 1039
NDoubleParseVN function 939
NDoubleParseVN macro 1039
NDoubleToChars function 940
NDoubleToChars macro 1039
NDoubleToString function 940
NDoubleToString macro 1039
NDoubleTryParse macro 1039
NDoubleTryParseChars macro 1040
NDoubleTryParseN function 941
NDoubleTryParseStrOrChars function 941
NDoubleTryParseStrOrChars macro 1040
NDoubleTryParseVN function 942
NDoubleTryParseVN macro 1040
ndowFriday enumeration member 376
ndowMonday enumeration member 376
ndowSaturday enumeration member 376
ndowSunday enumeration member 376
ndowThursday enumeration member 376

Copyright © 2009-2014 Neurotechnology. All rights reserved.

ndowTuesday enumeration member 376
ndowWednesday enumeration member 376
ndtAny enumeration member 1165
ndtBiometricDevice enumeration member 1165
ndtCamera enumeration member 1165
ndtCaptureDevice enumeration member 1165
ndtFingerScanner enumeration member 1165
ndtFScanner enumeration member 1165
ndtIrisScanner enumeration member 1165
ndtMicrophone enumeration member 1165
ndtNone enumeration member 1165
ndtPalmScanner enumeration member 1165
NE_IS_DIRECTORY_ACCESS macro 435
NE_MERGE_CALL_STACK macro 435
NE_NO_CALL_STACK macro 436
NE_PRESERVE_INNER_ERROR macro 436
NE_SKIP_ONE_FRAME macro 436
neAscii enumeration member 396
neDefault enumeration member 396
NEncoding Unit 389
NEncoding_ enumeration 396
NEncodingConvert function 391
NEncodingDetect function 391
NEncodingGetBytes function 392
NEncodingGetBytes macro 397
NEncodingGetBytesForChars function 392
NEncodingGetBytesForChars macro 397
NEncodingGetChars function 393
NEncodingGetChars macro 398
NEncodingGetMaxByteCount function 393
NEncodingGetMaxByteCount macro 398
NEncodingGetMaxCharCount function 393
NEncodingGetMaxCharCount macro 398
NEncodingGetMinBytesPerChar function 394
NEncodingGetPreamble function 394
NEncodingGetPreambleLength function 395
NEncodingGetStringN function 395
NEncodingIsSingleByte function 395
NEncodingIsValid function 396
NEnum Unit 386
NEnumParseN function 387
NEnumParseP function 387
NEnumToStringN function 387
NEnumToStringP function 388
NEnumTryParseN function 388
NEnumTryParseP function 389
nepBoth enumeration member 91
nepLeft enumeration member 91
NEPosition enumeration 91
NEPosition_ enumeration 91
NEPositionGetAvailableParts function 67
NEPositionIsKnown function 68
NEPositionIsLeft function 68
NEPositionIsOneOf function 69
NEPositionIsRight function 69
NEPositionIsRightAndLeft function 69
NEPositionIsSingleEye function 70
NEPositionIsTheEye function 70
NEPositionIsTwoEyes function 70
NEPositionIsValid function 70
NEPositionIsValidTheEye function 71
nepRight enumeration member 91
nepUnknown enumeration member 91
NError Unit 398
NError.h 443
NErrorAppend function 404
NErrorAppend macro 436
NErrorGetCallStackCount function 405
NErrorGetCallStackEntry function 405
NErrorGetCallStackFileEx function 405
NErrorGetCallStackFileEx macro 436
NErrorGetCallStackFunctionEx function 406
NErrorGetCallStackFunctionEx macro 436
NErrorGetCallStackLength function 406
NErrorGetCallStackLine function 406
NErrorGetCallStackLineEx function 407
NErrorGetCallStackN function 407
NErrorGetClrError macro 437
NErrorGetCode function 407
NErrorGetCodeEx function 408
NErrorGetComError macro 437
NErrorGetDefaultMessageEx function 408

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NErrorGetDefaultMessageEx macro 437
NErrorGetDefaultMessageN function 408
NErrorGetExternalCallStack macro 437
NErrorGetExternalCallStackEx function 409
NErrorGetExternalCallStackEx macro 438
NErrorGetExternalCallStackN function 409
NErrorGetExternalError function 410
NErrorGetExternalErrorEx function 410
NErrorGetInnerError function 410
NErrorGetInnerErrorEx function 411
NErrorGetLast function 411
NErrorGetLastEx function 411
NErrorGetLastSysError function 412
NErrorGetLastWin32Error function 412
NErrorGetMacErrorMessage macro 438
NErrorGetMacErrorMessageEx function 412
NErrorGetMacErrorMessageEx macro 438
NErrorGetMacErrorMessageN function 412
NErrorGetMachErrorMessageN function 413
NErrorGetMessageEx function 413
NErrorGetMessageEx macro 438
NErrorGetMessageN function 413
NErrorGetParam macro 439
NErrorGetParamEx function 414
NErrorGetParamEx macro 439
NErrorGetParamN function 414
NErrorGetSysError macro 439
NErrorGetSysErrorMessageEx function 414
NErrorGetSysErrorMessageEx macro 439
NErrorGetSysErrorMessageN function 415
NErrorGetWin32Error macro 440
NErrorGetWin32ErrorMessageN function 415
NErrorReport function 416
NErrorReportEx function 416
NErrorSetCom function 416
NErrorSetCom macro 440
NErrorSetComMPN function 417
NErrorSetComMPN macro 440
NErrorSetComN function 417
NErrorSetComPN function 417
NErrorSetComPN macro 440
NErrorSetLast function 418
NErrorSetLast macro 440
NErrorSetLastEx function 418
NErrorSetLastMPN function 419
NErrorSetLastMPN macro 441
NErrorSetLastN function 419
NErrorSetLastPN function 419
NErrorSetLastPN macro 441
NErrorSetMac function 420
NErrorSetMac macro 441
NErrorSetMach function 420
NErrorSetMach macro 441
NErrorSetMachMPN function 421
NErrorSetMachMPN macro 441
NErrorSetMachN function 421
NErrorSetMachPN function 421
NErrorSetMachPN macro 442
NErrorSetMacMPN function 422
NErrorSetMacMPN macro 442
NErrorSetMacN function 422
NErrorSetMacPN function 423
NErrorSetMacPN macro 442
NErrorSetSys function 423
NErrorSetSys macro 442
NErrorSetSysMPN function 423
NErrorSetSysMPN macro 442
NErrorSetSysN function 424
NErrorSetSysPN function 424
NErrorSetSysPN macro 442
NErrorSetWin32 function 425
NErrorSetWin32 macro 443
NErrorSetWin32MPN function 425
NErrorSetWin32MPN macro 443
NErrorSetWin32N function 425
NErrorSetWin32PN function 426
NErrorSetWin32PN macro 443
NErrorSuppress function 426
Neurotec Library 1916
Neurotec namespace 1469
Neurotec.Biometrics namespace 1654
Neurotec.Biometrics.Gui 1918

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Neurotec.Biometrics.Gui namespace 1782
Neurotec.Biometrics.NBiometricStatus enumeration 1769
Neurotec.Biometrics.NBiometricSubtype enumeration 1770
Neurotec.Biometrics.NBiometricType enumeration 1771
Neurotec.Biometrics.NEPosition enumeration 1772
Neurotec.Biometrics.NFImpressionType enumeration 1773
Neurotec.Biometrics.NFMinutiaFormat enumeration 1774
Neurotec.Biometrics.NFMinutiaType enumeration 1774
Neurotec.Biometrics.NFPatternClass enumeration 1775
Neurotec.Biometrics.NFPosition enumeration 1776
Neurotec.Biometrics.NFRidgeCountsType enumeration 1778
Neurotec.Biometrics.NGender enumeration 1778
Neurotec.Biometrics.NleExtractionStatus enumeration 1779
Neurotec.Biometrics.NleTemplateSize enumeration 1779
Neurotec.Biometrics.NLExpression enumeration 1780
Neurotec.Biometrics.NLProperties enumeration 1780
Neurotec.Biometrics.NMatchingSpeed enumeration 1781
Neurotec.Biometrics.NMFusionType enumeration 1781
Neurotec.Devices namespace 1792
Neurotec.Devices.NCameraStatus enumeration 1823
Neurotec.Devices.NDeviceType enumeration 1824
Neurotec.IO namespace 1825
Neurotec.Licensing 1918
Neurotec.Licensing namespace 1851
Neurotec.Licensing.NLicensingServiceStatus enumeration 1862
Neurotec.NLibraryGetInfo type 1654
Neurotec.NMemoryType enumeration 1651
Neurotec.NModuleOptions enumeration 1651
Neurotec.NProcessorVendor enumeration 1652
Neurotec.NTypeCode enumeration 1652
Neurotec.Plugins namespace 1863
Neurotec.Plugins.NPluginState enumeration 1882
Neurotec.Surveillance Changes 1918
Neurotec.Surveillance namespace 1883
Neurotec.Surveillance.NSurveillanceEventType enumeration 1908
neUtf16BE enumeration member 396
neUtf16LE enumeration member 396
neUtf32BE enumeration member 396
neUtf32LE enumeration member 396
neUtf7 enumeration member 396
neUtf8 enumeration member 396
Neutral enumeration member 1780
NEvent Unit 448
NEvent.h 457
NEventAdd macro 456
NEventAddN function 449
NEventAddRaw function 449
NEventDispose function 449
NEventGetCallbacks function 450
NEventInit function 450
NEventIsEmpty function 450
NEventReleaseCallbacks function 451
NEventRemove macro 456
NEventRemoveN function 451
NEventRemoveRaw function 452
NEView class 1782
- about NEView class 1782
- Image 1783
- InnerBoundaryColor 1783
- InnerBoundaryWidth 1783
- NEView 1783
- OuterBoundaryColor 1783
- OuterBoundaryWidth 1783
- SegmentationDetails 1784

NexGen enumeration member 1652
NExpandableObject class 1537
- about NExpandableObject class 1537
- NativeType 1539
- Properties 1539

NExpandableObject Unit 457
NExpandableObject.h 458
NExpandableObjectGetProperties function 458
NExtractors Unit 103
NExtractors.h 103
NFailed macro 434
NFalse macro 1041
nfaRead enumeration member 496
nfaReadWrite enumeration member 496
nfatDirectory enumeration member 477
nfatNone enumeration member 477
nfaWrite enumeration member 496

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFCore structure 1680
about NFCore structure 1680
Angle 1681
NativeType 1681
NFCore 1680, 1681
RawAngle 1682
ToString 1681
X 1682
Y 1682
NFCoreToString function 71
NFCoreToString macro 98
NFDelta structure 1682
about NFDelta structure 1682
Angle1 1684
Angle2 1685
Angle3 1685
NativeType 1685
NFDelta 1683
RawAngle1 1685
RawAngle2 1685
RawAngle3 1685
ToString 1684
X 1686
Y 1686
NFDeltaToString function 71
NFDeltaToString macro 99
NFDoubleCore structure 1686
about NFDoubleCore structure 1686
NativeType 1687
NFDoubleCore 1687
ToString 1687
X 1687
Y 1688
NFDoubleCoreToString function 72
NFDoubleCoreToString macro 99
NFile Unit 458
NFileAccess enumeration 496
NFileAccess_ enumeration 496
NFileAccessIsValid function 494
NFileAppendAllLines function 461
NFileAppendAllLinesPN function 461
NFileAppendAllLinesWithEncoding function 461
NFileAppendAllLinesWithEncodingPN function 462
NFileAppendAllText function 462
NFileAppendAllText macro 477
NFileAppendAllTextPN function 462
NFileAppendAllTextPN macro 477
NFileAppendAllTextWithEncoding function 463
NFileAppendAllTextWithEncoding macro 478
NFileAppendAllTextWithEncodingPN function 463
NFileAppendAllTextWithEncodingPN macro 478
NFileAppendText function 464
NFileAppendText macro 478
NFileAttributes enumeration 477
NFileAttributes_ enumeration 477
NFileCreate function 464
NFileCreate macro 478
NFileCreateText function 464
NFileCreateText macro 478
NFileCreateWithBufferSize function 465
NFileCreateWithBufferSize macro 478
NFileDelete function 465
NFileDelete macro 479
NFileExists function 465
NFileExists macro 479
NFileGetAttributes function 466
NFileGetAttributes macro 479
NFileMode enumeration 496
NFileMode_ enumeration 496
NFileModeIsValid function 495
NFileOpen function 466
NFileOpen macro 479
NFileOpenRead function 466
NFileOpenRead macro 479
NFileOpenText function 467
NFileOpenText macro 479
NFileOpenWithAccess function 467
NFileOpenWithAccess macro 480
NFileOpenWithAccessAndShare function 468
NFileOpenWithAccessAndShare macro 480
NFileOpenWrite function 468
NFileOpenWrite macro 480

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFileReadAllBytesCN function 468
NFileReadAllBytesCN macro 480
NFileReadAllBytesN function 469
NFileReadAllLinesCN function 469
NFileReadAllLinesCN macro 480
NFileReadAllLinesN function 469
NFileReadAllLinesWithEncodingCN function 470
NFileReadAllLinesWithEncodingCN macro 481
NFileReadAllLinesWithEncodingN function 470
NFileReadAllTextCN function 471
NFileReadAllTextCN macro 481
NFileReadAllTextN function 471
NFileReadAllTextWithEncodingCN function 471
NFileReadAllTextWithEncodingCN macro 481
NFileReadAllTextWithEncodingN function 472
NFileShare enumeration 497
NFileShare_ enumeration 497
NFileShareIsValid function 495
NFileStream Unit 483
NFileStreamCreate function 484
NFileStreamCreate macro 487
NFileStreamCreateFromOSHandle function 484
NFileStreamCreateFromOSHandleWithBufferSize function 484
NFileStreamCreateWithAccess function 485
NFileStreamCreateWithAccess macro 487
NFileStreamCreateWithAccessAndShare function 485
NFileStreamCreateWithAccessAndShare macro 487
NFileStreamCreateWithAccessShareAndBufferSize function 486
NFileStreamCreateWithAccessShareAndBufferSize macro 487
NFileStreamGetOSHandle function 486
NFileWriteAllBytes function 472
NFileWriteAllBytes macro 481
NFileWriteAllBytesCN function 473
NFileWriteAllBytesCN macro 481
NFileWriteAllBytesPN function 473
NFileWriteAllLines function 473
NFileWriteAllLines macro 481
NFileWriteAllLinesPN function 474
NFileWriteAllLinesPN macro 482
NFileWriteAllLinesWithEncoding function 474
NFileWriteAllLinesWithEncoding macro 482
NFileWriteAllLinesWithEncodingPN function 475
NFileWriteAllLinesWithEncodingPN macro 482
NFileWriteAllText function 475
NFileWriteAllText macro 482
NFileWriteAllTextPN function 476
NFileWriteAllTextPN macro 482
NFileWriteAllTextWithEncoding function 476
NFileWriteAllTextWithEncoding macro 482
NFileWriteAllTextWithEncodingPN function 476
NFileWriteAllTextWithEncodingPN macro 483
NFill function 506
NFImpressionType enumeration 91
NFImpressionType_ enumeration 91
NFImpressionTypeIsCompatibleWith function 72
NFImpressionTypeIsContact function 73
NFImpressionTypeIsContactless function 73
NFImpressionTypeIsFinger function 73
NFImpressionTypeIsGeneric function 74
NFImpressionTypeIsLatent function 74
NFImpressionTypeIsLiveScan function 74
NFImpressionTypeIsNonliveScan function 74
NFImpressionTypeIsNonOptical function 75
NFImpressionTypeIsOneOf function 75
NFImpressionTypeIsOptical function 76
NFImpressionTypeIsPalm function 76
NFImpressionTypeIsPlain function 76
NFImpressionTypeIsRolled function 76
NFImpressionTypeIsSwipe function 77
NFImpressionTypeIsValid function 77
NFImpressionTypeIsValidFinger function 77
NFImpressionTypeIsValidPalm function 78
NFImpressionTypeToFinger function 78
nfitLatentImpression enumeration member 91
nfitLatentLift enumeration member 91
nfitLatentPalmImpression enumeration member 91
nfitLatentPalmLift enumeration member 91
nfitLatentPalmPhoto enumeration member 91
nfitLatentPalmTracing enumeration member 91
nfitLatentPhoto enumeration member 91

Copyright © 2009-2014 Neurotechnology. All rights reserved.

nfitLatentTracing enumeration member 91
nfitLiveScanContactless enumeration member 91
nfitLiveScanNonOpticalContactlessPlain enumeration member 91
nfitLiveScanNonOpticalContactlessRolled enumeration member 91
nfitLiveScanNonOpticalContactPlain enumeration member 91
nfitLiveScanNonOpticalContactRolled enumeration member 91
nfitLiveScanOpticalContactlessPlain enumeration member 91
nfitLiveScanOpticalContactlessRolled enumeration member 91
nfitLiveScanOpticalContactPlain enumeration member 91
nfitLiveScanOpticalContactRolled enumeration member 91
nfitLiveScanPalm enumeration member 91
nfitLiveScanPlain enumeration member 91
nfitLiveScanRolled enumeration member 91
nfitLiveScanVerticalSwipe enumeration member 91
nfitNonliveScanPalm enumeration member 91
nfitNonliveScanPlain enumeration member 91
nfitNonliveScanRolled enumeration member 91
nfitOther enumeration member 91
nfitSwipe enumeration member 91
nfitUnknown enumeration member 91
NFloat type 1005
nfmAppend enumeration member 496
nfmCreate enumeration member 496
nfmCreateNew enumeration member 496
NFMinutia structure 1688
- about NFMinutia structure 1688
- Angle 1690
- Curvature 1690
- G 1691
- NativeType 1691
- NFMinutia 1688, 1689
- Quality 1691
- RawAngle 1691
- ToString 1690
- Type 1691
- X 1692
- Y 1692

NFMinutiaNeighbor structure 1692
- about NFMinutiaNeighbor structure 1692
- Empty 1693
- Index 1693
- NativeType 1694
- NFMinutiaNeighbor 1693
- RidgeCount 1694
- ToString 1693

NFMinutiaNeighborToString function 78
NFMinutiaNeighborToString macro 99
NFMinutiaToString function 79
NFMinutiaToString macro 99
nfmOpen enumeration member 496
nfmOpenOrCreate enumeration member 496
nfmTruncate enumeration member 496
nfpLeftFullPalm enumeration member 92
nfpLeftHypothenar enumeration member 92
nfpLeftIndex enumeration member 92
nfpLeftIndexFinger enumeration member 92
nfpLeftIndexMiddleFingers enumeration member 92
nfpLeftIndexMiddleRingFingers enumeration member 92
nfpLeftInterdigital enumeration member 92
nfpLeftLittle enumeration member 92
nfpLeftLittleFinger enumeration member 92
nfpLeftLowerPalm enumeration member 92
nfpLeftMiddle enumeration member 92
nfpLeftMiddleFinger enumeration member 92
nfpLeftMiddleRingFingers enumeration member 92
nfpLeftMiddleRingLittleFingers enumeration member 92
nfpLeftOther enumeration member 92
nfpLeftRing enumeration member 92
nfpLeftRingFinger enumeration member 92
nfpLeftRingLittleFingers enumeration member 92
nfpLeftThenar enumeration member 92
nfpLeftThumb enumeration member 92
nfpLeftUpperPalm enumeration member 92
nfpLeftWritersPalm enumeration member 92
NFPosition enumeration 92
NFPosition_ enumeration 92
NFPositionGetAvailableParts function 79
NFPositionIsCompatibleWith function 80
NFPositionIsCompatibleWithImpressionType function 80
NFPositionIsFinger function 80

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NFPositionIsFourFingers function 81
NFPositionIsKnown function 81
NFPositionIsLeft function 81
NFPositionIsLeftAndRight function 81
NFPositionIsOneOf function 82
NFPositionIsPalm function 82
NFPositionIsRight function 83
NFPositionIsSingleFinger function 83
NFPositionIsTheFinger function 83
NFPositionIsThePalm function 83
NFPositionIsThreeFingers function 84
NFPositionIsTwoFingers function 84
NFPositionIsValid function 84
NFPositionIsValidFinger function 85
NFPositionIsValidPalm function 85
NFPositionIsValidTheFinger function 85
nfpPlainLeftFourFingers enumeration member 92
nfpPlainLeftThumb enumeration member 92
nfpPlainRightFourFingers enumeration member 92
nfpPlainRightThumb enumeration member 92
nfpPlainThumbs enumeration member 92
nfpRightFullPalm enumeration member 92
nfpRightHypothenar enumeration member 92
nfpRightIndex enumeration member 92
nfpRightIndexFinger enumeration member 92
nfpRightIndexLeftIndexFingers enumeration member 92
nfpRightIndexMiddleFingers enumeration member 92
nfpRightIndexMiddleRingFingers enumeration member 92
nfpRightInterdigital enumeration member 92
nfpRightLittle enumeration member 92
nfpRightLittleFinger enumeration member 92
nfpRightLowerPalm enumeration member 92
nfpRightMiddle enumeration member 92
nfpRightMiddleFinger enumeration member 92
nfpRightMiddleRingFingers enumeration member 92
nfpRightMiddleRingLittleFingers enumeration member 92
nfpRightOther enumeration member 92
nfpRightRing enumeration member 92
nfpRightRingFinger enumeration member 92
nfpRightRingLittleFingers enumeration member 92
nfpRightThenar enumeration member 92
nfpRightThumb enumeration member 92
nfpRightUpperPalm enumeration member 92
nfpRightWritersPalm enumeration member 92
nfpUnknown enumeration member 92
nfpUnknownFourFingers enumeration member 92
nfpUnknownPalm enumeration member 92
nfpUnknownThreeFingers enumeration member 92
nfpUnknownTwoFingers enumeration member 92
NFree function 506
NFreeEx function 507
nfsNone enumeration member 497
nfsRead enumeration member 497
nfsReadWrite enumeration member 497
nfsWrite enumeration member 497
NFView class 1784
about NFView class 1784
GetDoubleCoreAtScreenPoint 1785
HoveredDoubleCoreIndexChanged 1786
OnHoveredDoubleCoreIndexChanged 1785
OnSelectedDoubleCoreIndexChanged 1785
SelectedCoreIndex 1785
SelectedDeltaIndex 1786
SelectedDoubleCoreIndex 1786
SelectedDoubleCoreIndexChanged 1786
SelectedMinutiaIndex 1786
Template 1786
NFView.PointerTool class 1785
about NFView.PointerTool class 1785
NGender enumeration 95
NGender_ enumeration 95
NGeometry Unit 1222
NGeometry.h 1228
ngFemale enumeration member 95
ngMale enumeration member 95
NGrayscaleImage Unit 1228
NGrayscaleImage.h 1232
NGrayscaleImageGetMaxValue function 1229
NGrayscaleImageGetMinValue function 1229
NGrayscaleImageGetPixel function 1230
NGrayscaleImageSetMaxValue function 1230
NGrayscaleImageSetMinValue function 1231

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NGrayscaleImageSetPixel function 1231
NGuidParse function 942
NGuidParse macro 1041
NGuidParseVN function 943
NGuidParseVN macro 1041
NGuidToString function 943
NGuidToString macro 1041
NGuidTryParse function 943
NGuidTryParse macro 1041
NGuidTryParseVN function 944
NGuidTryParseVN macro 1041
ngUnknown enumeration member 95
ngUnspecified enumeration member 95
NHandle type 1006
NI_ALL_DST macro 1284
NI_ALL_DST_AND_SRC macro 1284
NI_ALL_SRC macro 1284
NI_DST_ALPHA_CHANNEL_FIRST macro 1284
NI_DST_BOTTOM_TO_TOP macro 1284
NI_DST_SWAP_CHANNELS macro 1285
NI_READ_UNKNOWN_IMAGE macro 1285
NI_SRC_ALPHA_CHANNEL_FIRST macro 1285
NI_SRC_BOTTOM_TO_TOP macro 1285
NI_SRC_SWAP_CHANNELS macro 1285
NI_WRITE_UNKNOWN_IMAGE macro 1285
NImage Unit 1232
NImage.h 1291
NImageClone macro 1286
NImageCopy function 1235
NImageCopyData function 1236
NImageCopyDataPlanes function 1237
NImageCopyDataPlanesEx function 1238
NImageCopyFromData function 1239
NImageCopyFromDataPart function 1240
NImageCopyFromDataPlanesPart function 1241
NImageCopyFromDataPlanesPartEx function 1242
NImageCopyTo function 1236
NImageCopyToData function 1243
NImageCopyToDataPart function 1244
NImageCopyToDataPlanesPart function 1245
NImageCopyToDataPlanesPartEx function 1246
NImageCreate function 1247
NImageCreateEx function 1248
NImageCreateFromData function 1249
NImageCreateFromDataEx function 1251
NImageCreateFromDataPart function 1251
NImageCreateFromDataPartEx function 1252
NImageCreateFromDataPlanesPart function 1254
NImageCreateFromFileA macro 1286
NImageCreateFromFileEx function 1242
NImageCreateFromFileEx macro 1286
NImageCreateFromFileExN function 1256
NImageCreateFromFileW macro 1287
NImageCreateFromImage macro 1287
NImageCreateFromImageEx function 1257
NImageCreateFromImageEx2 function 1258
NImageCreateFromImagePart macro 1287
NImageCreateFromImagePartEx function 1258
NImageCreateFromImagePartEx2 function 1260
NImageCreateFromMemory function 1253
NImageCreateFromStream function 1255
NImageCreateWrapper function 1261
NImageCreateWrapperEx function 1262
NImageCreateWrapperForImagePart macro 1288
NImageCreateWrapperForImagePartEx2 function 1260
NImageCreateWrapperForPart function 1264
NImageCreateWrapperForPartEx function 1266
NImageCreateWrapperForPlanesPart function 1267
NImageCrop function 1248
NImageExportPalette function 1268
NImageFile Unit 1294
NImageFile.h 1298
NImageFileClose function 1295
NImageFileCreate function 1295
NImageFileFree macro 1298
NImageFileGetFormat function 1296
NImageFileGetFormatEx function 1296
NImageFileIsOpened function 1297
NImageFileReadImage function 1297
NImageFlipDiagonally function 1250
NImageFlipHorizontally function 1254
NImageFlipVertically function 1256

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NImageFormat Unit 1299
NImageFormat.h 1329
NImageFormatCanRead function 1301
NImageFormatCanWrite function 1302
NImageFormatCanWriteMultiple function 1302
NImageFormatCreateInfo function 1303
NImageFormatGetBmp function 1303
NImageFormatGetBmpEx function 1304
NImageFormatGetDefaultFileExtensionA macro 1325
NImageFormatGetDefaultFileExtensionEx function 1303
NImageFormatGetDefaultFileExtensionEx macro 1325
NImageFormatGetDefaultFileExtensionW macro 1325
NImageFormatGetDefaultInternetMediaType function 1305
NImageFormatGetDefaultInternetMediaType macro 1326
NImageFormatGetFileFilterA macro 1326
NImageFormatGetFileFilterEx function 1305
NImageFormatGetFileFilterEx macro 1326
NImageFormatGetFileFilterW macro 1326
NImageFormatGetFormat function 1304
NImageFormatGetFormatCount function 1306
NImageFormatGetFormatEx function 1306
NImageFormatGetIHead function 1306
NImageFormatGetIHeadEx function 1307
NImageFormatGetInternetMediaType function 1308
NImageFormatGetInternetMediaType macro 1327
NImageFormatGetJpeg function 1307
NImageFormatGetJpeg2K function 1308
NImageFormatGetJpeg2KEx function 1310
NImageFormatGetJpegEx function 1311
NImageFormatGetNameA macro 1327
NImageFormatGetNameEx function 1309
NImageFormatGetNameEx macro 1327
NImageFormatGetNameW macro 1327
NImageFormatGetPng function 1309
NImageFormatGetPngEx function 1312
NImageFormatGetTiff function 1311
NImageFormatGetTiffEx function 1313
NImageFormatGetWsq function 1311
NImageFormatGetWsqEx function 1315
NImageFormatLoadImageFromFile function 1312
NImageFormatLoadImageFromMemory function 1314
NImageFormatLoadImageFromStream function 1315
NImageFormatOpenFile function 1310
NImageFormatOpenFileFromMemory function 1317
NImageFormatOpenFileFromStream function 1317
NImageFormatOpenReaderFromFile function 1318
NImageFormatOpenReaderFromFile macro 1328
NImageFormatOpenReaderFromMemory function 1319
NImageFormatOpenReaderFromStream function 1318
NImageFormatOpenWriterToFile function 1321
NImageFormatOpenWriterToFile macro 1328
NImageFormatOpenWriterToMemory function 1323
NImageFormatOpenWriterToStream function 1319
NImageFormatSaveImagesToFile function 1313
NImageFormatSaveImagesToMemory function 1314
NImageFormatSaveImagesToStream function 1316
NImageFormatSaveImageToFile function 1319
NImageFormatSaveImageToMemory function 1320
NImageFormatSaveImageToStream function 1320
NImageFormatSelect function 1321
NImageFormatSelectByInternetMediaType function 1322
NImageFormatSelectByInternetMediaType macro 1328
NImageFormatSelectEx function 1322
NImageFormatSelectEx macro 1328
NImageFormatSelectReaderFromFile function 1324
NImageFormatSelectReaderFromFile macro 1328
NImageFormatSelectReaderFromMemory function 1324
NImageFormatSelectReaderFromStream function 1323
NImageFree macro 1289
NImageGetHeight function 1257
NImageGetHorzResolution function 1259
NImageGetImageSize function 1271
NImageGetIndex function 1272
NImageGetInfo function 1273
NImageGetMaxValue function 1273
NImageGetMinValue function 1274
NImageGetPalette function 1275
NImageGetPaletteEntry function 1275
NImageGetPaletteLength function 1276
NImageGetPalettePtr function 1276
NImageGetPaletteSize function 1276
NImageGetPixel function 1277

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NImageGetPixelFormat function 1277
NImageGetPixels macro 1289
NImageGetPixelsN function 1278
NImageGetPixelsPtr function 1278
NImageGetPlane function 1278
NImageGetPlaneCount function 1279
NImageGetPlanePtr function 1279
NImageGetPlanePtrs function 1279
NImageGetPlanes function 1280
NImageGetPlaneSize function 1280
NImageGetResolutionIsAspectRatio function 1269
NImageGetStride function 1263
NImageGetVertResolution function 1265
NImageGetWidth function 1267
NImageRotateFlip function 1268
NImageRotateFlipType enumeration 1283
NImageRotateFlipType_ enumeration 1283
NImages Unit 1331
NImages.h 1333
NImageSaveToFileA macro 1289
NImageSaveToFileEx function 1269
NImageSaveToFileEx macro 1290
NImageSaveToFileW macro 1290
NImageSaveToMemory function 1270
NImageSaveToStream function 1271
NImageSetHorzResolution function 1272
NImageSetIndex function 1281
NImageSetMaxValue function 1281
NImageSetMinValue function 1281
NImageSetPalette function 1282
NImageSetPaletteEntry function 1282
NImageSetPaletteN function 1282
NImageSetPixel function 1283
NImageSetResolutionIsAspectRatio function 1272
NImageSetVertResolution function 1273
NImagesGetGrayscaleColorWrapperEx function 1331
NImagesGetInfoA macro 1332
NImagesGetInfoW macro 1333
NImagesModuleOf macro 1333
NImagesRecolorImage function 1332
NImageToGrayscale function 1274
NImageToMonochrome function 1274
NImageToRgb function 1275
nimBicubic enumeration member 1223
nimBilinear enumeration member 1223
nimNearestNeighbor enumeration member 1223
NIndexPair structure 1539
about NIndexPair structure 1539
Index1 1541
Index2 1541
NativeType 1541
NIndexPair 1540
ToString 1540
NIndexPair type 1006
NIndexPair_ structure 1001
NIndexPairToString function 944
NIndexPairToString macro 1042
NInfo class 1541
about NInfo class 1541
NInfoObject class 1541
about NInfoObject class 1541
NInt type 1006
NInt16 type 1006
NInt16Parse macro 1042
NInt16ParseChars macro 1042
NInt16ParseN function 945
NInt16ParseStrOrChars function 945
NInt16ParseStrOrChars macro 1043
NInt16ParseVN function 946
NInt16ParseVN macro 1043
NInt16ToChars function 946
NInt16ToChars macro 1043
NInt16ToString function 946
NInt16ToString macro 1043
NInt16TryParse macro 1043
NInt16TryParseChars macro 1044
NInt16TryParseN function 947
NInt16TryParseStrOrChars function 947
NInt16TryParseStrOrChars macro 1044
NInt16TryParseVN function 948
NInt16TryParseVN macro 1044
NInt32 type 1006

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NInt32Parse macro 1044
NInt32ParseChars macro 1045
NInt32ParseN function 948
NInt32ParseStrOrChars function 949
NInt32ParseStrOrChars macro 1045
NInt32ParseVN function 949
NInt32ParseVN macro 1045
NInt32ToChars function 949
NInt32ToChars macro 1046
NInt32ToString function 950
NInt32ToString macro 1046
NInt32TryParse macro 1046
NInt32TryParseChars macro 1046
NInt32TryParseN function 950
NInt32TryParseStrOrChars function 951
NInt32TryParseStrOrChars macro 1047
NInt32TryParseVN function 951
NInt32TryParseVN macro 1047
NInt64 type 1006
NInt64Parse macro 1047
NInt64ParseChars macro 1047
NInt64ParseN function 952
NInt64ParseStrOrChars function 952
NInt64ParseStrOrChars macro 1048
NInt64ParseVN function 952
NInt64ParseVN macro 1048
NInt64ToChars function 953
NInt64ToChars macro 1048
NInt64ToString function 953
NInt64ToString macro 1048
NInt64TryParse macro 1048
NInt64TryParseChars macro 1049
NInt64TryParseN function 954
NInt64TryParseStrOrChars function 954
NInt64TryParseStrOrChars macro 1049
NInt64TryParseVN function 955
NInt64TryParseVN macro 1049
NInt8 type 1007
NInt8Parse macro 1050
NInt8ParseChars macro 1050
NInt8ParseN function 955
NInt8ParseStrOrChars function 956
NInt8ParseStrOrChars macro 1050
NInt8ParseVN function 956
NInt8ParseVN macro 1051
NInt8ToChars function 956
NInt8ToChars macro 1051
NInt8ToString function 957
NInt8ToString macro 1051
NInt8TryParse macro 1051
NInt8TryParseChars macro 1051
NInt8TryParseN function 957
NInt8TryParseStrOrChars function 958
NInt8TryParseStrOrChars macro 1052
NInt8TryParseVN function 958
NInt8TryParseVN macro 1052
NInterlocked Unit 488
NInterlockedAdd function 488
NInterlockedAnd function 489
NInterlockedCompareExchange function 489
NInterlockedCompareExchangePointer function 489
NInterlockedDecrement function 490
NInterlockedExchange function 490
NInterlockedExchangeAdd function 490
NInterlockedExchangePointer function 491
NInterlockedIncrement function 491
NInterlockedOr function 491
NInterlockedXor function 492
NInterpolationMode enumeration 1223
NInterpolationMode_ enumeration 1223
NIOTypes Unit 493
NIOTypes.h 498
nirftFlipNone enumeration member 1283
nirftFlipX enumeration member 1283
nirftFlipXY enumeration member 1283
nirftFlipY enumeration member 1283
nirftNone enumeration member 1283
nirftRotate180 enumeration member 1283
nirftRotate270 enumeration member 1283
nirftRotate90 enumeration member 1283
nirftRotateNone enumeration member 1283
NKeyValuePair_ structure 1119

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NKeyValuePairCopy function 1096
NKeyValuePairCreate function 1096
NKeyValuePairDispose function 1096
NKeyValuePairSet function 1097
NLAttributes class 1694
about NLAttributes class 1694
NativeType 1697
NLAttributes 1697
NLAttributes Unit 103
NLAttributes.h 104
NLAttributesCreate function 104
NLE_FEATURE_POINT_COUNT macro 120
NleCompressEx function 107
NleCreate function 112
NleDetectFace function 108
NleDetectFaces function 108
NleDetectFacialFeatures function 109
NleDetectionDetails class 1697
about NleDetectionDetails class 1697
BlinkConfidence 1698
Clear 1698
DarkGlassesConfidence 1698
Expression 1698
ExpressionConfidence 1698
Face 1698
FaceAvailable 1698
Gender 1699
GenderConfidence 1699
GlassesConfidence 1699
LeftEyeCenter 1699
MouthCenter 1699
MouthOpenConfidence 1699
NleDetectionDetails 1698
NoseTip 1699
Points 1700
Properties 1700
RightEyeCenter 1700
NleDetectionDetails type 118
NleDetectionDetails_ structure 115
NleDetectionDetailsClear function 109
nleesEyesNotDetected enumeration member 117
nleesFaceNotDetected enumeration member 117
nleesFaceTooCloseToImageBorder enumeration member 117
nleesGeneralizationFailed enumeration member 117
nleesLivenessCheckFailed enumeration member 117
nleesNone enumeration member 117
nleesQualityCheckExposureFailed enumeration member 117
nleesQualityCheckGrayscaleDensityFailed enumeration member 117
nleesQualityCheckSharpnessFailed enumeration member 117
nleesTemplateCreated enumeration member 117
NleExtract function 110
NleExtractEnd macro 121
NleExtractEndEx function 111
NleExtractionStatus enumeration 117
NleExtractionStatus_ enumeration 117
NleExtractNextEx function 111
NleExtractNextEx2 function 112
NleExtractStart function 113
NleExtractStartEx function 113
NleExtractUsingDetails function 114
NleEyes structure 117, 1700
about NleEyes structure 1700
First 1700
FirstConfidence 1700
Second 1701
SecondConfidence 1701
NleEyes_ structure 117
nleEyesAway enumeration member 95
NleFace structure 118, 1701
about NleFace structure 1701
Confidence 1701
Rectangle 1701
Rotation 1702
NleFace_ structure 118
nleFrowning enumeration member 95
NleGeneralizeEx function 110
nleNeutral enumeration member 95
NLEP_BLINK_CONFIDENCE_THRESHOLD macro 121
NLEP_DARK_GLASSES_CONFIDENCE_THRESHOLD macro 121
NLEP_DETECT_ALL_FEATURE_POINTS macro 121
NLEP_DETECT_BASE_FEATURE_POINTS macro 121

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLEP_DETECT_BLINK macro 122
NLEP_DETECT_DARK_GLASSES macro 122
NLEP_DETECT_EXPRESSION macro 122
NLEP_DETECT_GENDER macro 122
NLEP_DETECT_GLASSES macro 122
NLEP_DETECT_MORE_FACES macro 123
NLEP_DETECT_MOUTH_OPEN macro 123
NLEP_EXPRESSION_CONFIDENCE_THRESHOLD macro 123
NLEP_FACE_CONFIDENCE_THRESHOLD macro 122
NLEP_FACE_DETECTION_SCALE_COUNT macro 123
NLEP_FACE_QUALITY_THRESHOLD macro 122
NLEP_FACE_ROLL_ANGLE_BASE macro 123
NLEP_FAVOR_LARGEST_FACE macro 123
NLEP_GENDER_CONFIDENCE_THRESHOLD macro 124
NLEP_GLASSES_CONFIDENCE_THRESHOLD macro 124
NLEP_LIVENESS_THRESHOLD macro 123
NLEP_MAX_IOD macro 124
NLEP_MAX_RECORDS_PER_TEMPLATE macro 124
NLEP_MAX_ROLL_ANGLE_DEVIATION macro 124
NLEP_MAX_STREAM_DURATION_IN_FRAMES macro 124
NLEP_MAX_YAW_ANGLE_DEVIATION macro 125
NLEP_MIN_IOD macro 124
NLEP_MOUTH_OPEN_CONFIDENCE_THRESHOLD macro 125
NLEP_TEMPLATE_SIZE macro 125
NLEP_USE_LIVENESS_CHECK macro 125
nleRaisedBrows enumeration member 95
NleRotation structure 115, 1702
- about NleRotation structure 1702
- Pitch 1702
- Roll 1702
- Yaw 1702

NleRotation_ structure 115
nleSmile enumeration member 95
nleSmileOpenedJaw enumeration member 95
nleSquinting enumeration member 95
NleTemplateSize enumeration 115
NleTemplateSize_ enumeration 115
nletsCompact enumeration member 115
nletsLarge enumeration member 115
nletsMedium enumeration member 115
nletsSmall enumeration member 115
nleUnknown enumeration member 95
nleUnspecified enumeration member 95
NLExpression enumeration 95
NLExpression_ enumeration 95
NLExtractor class 1702
- about NLExtractor class 1702
- BlinkConfidenceThreshold 1715
- Compress 1712
- DarkGlassesConfidenceThreshold 1715
- DetectAllFeaturePoints 1716
- DetectBaseFeaturePoints 1716
- DetectBlink 1716
- DetectDarkGlasses 1716
- DetectExpression 1716
- DetectFace 1712
- DetectFaces 1713
- DetectFacialFeatures 1713
- DetectGender 1716
- DetectGlasses 1717
- DetectMoreFaces 1717
- DetectMouthOpen 1717
- ExpressionConfidenceThreshold 1717
- Extract 1714
- ExtractEnd 1714
- ExtractNext 1714
- ExtractStart 1714
- ExtractUsingDetails 1715
- FaceConfidenceThreshold 1717
- FaceDetectionScaleCount 1718
- FaceQualityThreshold 1718
- FaceRollAngleBase 1718
- FavorLargestFace 1718
- GenderConfidenceThreshold 1718
- Generalize 1715
- GlassesConfidenceThreshold 1718
- LivenessThreshold 1719
- MaxIod 1719
- MaxRecordsPerTemplate 1719
- MaxRollAngleDeviation 1719
- MaxStreamDurationInFrames 1720

Copyright © 2009-2014 Neurotechnology. All rights reserved.

MaxYawAngleDeviation 1720
MinIod 1720
MouthOpenConfidenceThreshold 1720
NativeType 1720
NleExtractionStatusNativeType 1720
NleTemplateSizeNativeType 1721
NLExtractor 1708
ParameterBlinkConfidenceThreshold 1708
ParameterDarkGlassesConfidenceThreshold 1708
ParameterDetectAllFeaturePoints 1709
ParameterDetectBaseFeaturePoints 1709
ParameterDetectBlink 1709
ParameterDetectDarkGlasses 1709
ParameterDetectExpression 1709
ParameterDetectGender 1709
ParameterDetectGlasses 1709
ParameterDetectMoreFaces 1710
ParameterDetectMouthOpen 1710
ParameterExpressionConfidenceThreshold 1710
ParameterFaceConfidenceThreshold 1710
ParameterFaceDetectionScaleCount 1710
ParameterFaceQualityThreshold 1710
ParameterFaceRollAngleBase 1710
ParameterFavorLargestFace 1710
ParameterGenderConfidenceThreshold 1711
ParameterGlassesConfidenceThreshold 1711
ParameterLivenessThreshold 1711
ParameterMaxIod 1711
ParameterMaxRecordsPerTemplate 1711
ParameterMaxRollAngleDeviation 1711
ParameterMaxStreamDurationInFrames 1711
ParameterMaxYawAngleDeviation 1712
ParameterMinIod 1712
ParameterMouthOpenConfidenceThreshold 1712
ParameterTemplateSize 1712
ParameterUseLivenessCheck 1712
TemplateSize 1721
UseLivenessCheck 1721
NLExtractor Unit 104
Macros 119
NLExtractor.h 125
NLExtractorParams.h 127
NLFeaturePoint structure 1721
about NLFeaturePoint structure 1721
Code 1723
Confidence 1723
NativeType 1723
NLFeaturePoint 1722
ToString 1722
X 1723
Y 1723
NLFeaturePoint type 97
NLFeaturePoint_ structure 86
NLFeaturePointToString function 86
NLFeaturePointToString macro 99
NLibraryInfo class 1541
about NLibraryInfo class 1541
Activated 1542
Company 1543
Copyright 1543
GetString 1542
Product 1543
Retrieve 1542
Title 1543
Version 1543
NLibraryInfo Unit 498
NLibraryInfo.h 500
NLibraryInfo_ structure 499
NLicense class 1851
about NLicense class 1851
ActivateOnline 1852
DeactivateOnline 1852
DeactivateOnlineWithId 1853
GenerateDeactivationId 1853
GenerateDeactivationIdForLicense 1854
GenerateId 1855
GetInfo 1855
GetInfoForComponent 1856
IsComponentActivated 1856
NativeType 1858
Obtain 1856, 1857
ObtainComponents 1857

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Release 1858
ReleaseComponents 1858
NLicense Unit 1179
NLicense.h 1193
NLicenseActivateOnline function 1180
NLicenseActivateOnline macro 1191
NLicenseDeactivateOnline function 1181
NLicenseDeactivateOnline macro 1191
NLicenseDeactivateOnlineWithId function 1181
NLicenseDeactivateOnlineWithId macro 1191
NLicenseGenerateDeactivationId function 1183
NLicenseGenerateDeactivationId macro 1192
NLicenseGenerateDeactivationIdForLicense function 1183
NLicenseGenerateDeactivationIdForLicense macro 1192
NLicenseGenerateId function 1184
NLicenseGenerateId macro 1192
NLicenseGetInfo function 1181
NLicenseGetInfoForComponent function 1182
NLicenseGetInfoForComponent macro 1192
NLicenseInfo structure 1858
about NLicenseInfo structure 1858
DistributorId 1859
IsObtained 1859
NativeType 1859
NLicenseInfo 1859
SerialNumber 1859
NLicenseInfo type 1191
NLicenseInfo_ structure 1190
NLicenseIsComponentActivated function 1184
NLicenseIsComponentActivated macro 1192
NLicenseObtain function 1185
NLicenseObtainComponents function 1187
NLicenseObtainComponents macro 1193
NLicenseRelease function 1189
NLicenseReleaseComponents function 1190
NLicenseReleaseComponents macro 1193
NLicensing class 1860
about NLicensing class 1860
DllName 1860
NativeModule 1860
NLicensing Library 1179, 1915
NLicensing Unit 1194
NLicensing.h 1195
NLicensingGetInfo function 1195
NLicensingModuleOf 1195
NLicensingService class 1860
about NLicensingService class 1860
GetBinPath 1861
GetConfPath 1861
GetStatus 1861
Install 1861
IsTrial 1862
NativeType 1862
NLicensingServiceStatusNativeType 1862
Start 1862
Stop 1862
Uninstall 1862
NLicensingService Unit 1195
NLicensingServiceGetBinPath function 1909
NLicensingServiceGetConfPath function 1909
NLicensingServiceGetStatus function 1910
NLicensingServiceInstall function 1910
NLicensingServiceInstall macro 1912
NLicensingServiceIsTrial function 1910
NLicensingServiceStart function 1911
NLicensingServiceStatus enumeration 1911
NLicensingServiceStatus_ enumeration 1911
NLicensingServiceStop function 1911
NLicensingServiceUninstall function 1911
NLMatcher Unit 128
Macros 133
NLMatcher.h 133
NLMatcherParams.h 134
NLMatchingDetails class 1723
about NLMatchingDetails class 1723
NativeType 1723
NLMatchingDetails Unit 134
NLMatchingDetails.h 134
NlmCreate function 129
NlmIdentifyEnd function 130
NlmIdentifyNext function 130
NlmIdentifyStart function 131

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLMP_MATCHING_SPEED macro 133
NLMP_MATCHING_THRESHOLD macro 133
nlmsHigh enumeration member 132
nlmsLow enumeration member 132
NlmSpeed enumeration 132
NlmSpeed_ enumeration 132
NlmVerify function 131
NLong type 1007
nlpBeard enumeration member 96
nlpBlink enumeration member 96
nlpBothEyePatch enumeration member 96
nlpDarkGlasses enumeration member 96
nlpDistortingCondition enumeration member 96
nlpGlasses enumeration member 96
nlpHat enumeration member 96
nlpLeftEyePatch enumeration member 96
nlpMoustache enumeration member 96
nlpMouthOpen enumeration member 96
nlpNoEar enumeration member 96
nlpNotSpecified enumeration member 96
nlpRightEyePatch enumeration member 96
NLProperties enumeration 96
NLProperties_ enumeration 96
nlpScarf enumeration member 96
nlpSpecified enumeration member 96
nlpTeethVisible enumeration member 96
NLRecord class 1724
about NLRecord class 1724
CbeffProductType 1729
Check 1728
GetQuality 1728
GetSize 1729
NativeType 1729
NLRecord 1726, 1727
Quality 1730
NLRecord Unit 134
NLRecord.h 142
NLRecordCheck function 135
NLRecordClone macro 140
NLRecordCreate function 136
NLRecordCreateFromMemory function 136
NLRecordGetCbeffProductType function 137
NLRecordGetCbeffProductTypeMem function 137
NLRecordGetQuality function 138
NLRecordGetQualityMem function 138
NLRecordGetSize macro 141
NLRecordGetSizeMem function 139
NLRecordSaveToMemory macro 141
NLRecordSetCbeffProductType function 139
NLRecordSetQuality function 140
NLSM_PART_NLM macro 151
NLSM_PART_NONE macro 151
NlsMatcher Unit 143
Macros 150
NlsMatcher.h 151
NlsMatcherParams.h 152
NlsmCreate function 144
NlsmFaceMatchDetails structure 149
NlsmFaceMatchDetails_ structure 149
NlsmIdentifyEnd function 144
NlsmIdentifyNext function 145
NlsmIdentifyStart function 145
NlsmMatchDetails structure 150
NlsmMatchDetails.h 152
NlsmMatchDetails_ structure 150
NlsmMatchDetailsDeserialize function 146
NlsmMatchDetailsDeserializeFromStream function 147
NlsmMatchDetailsFree function 147
NlsmMatchDetailsSerialize function 147
NlsmMatchDetailsSerializeToStream function 148
NLSMP_MATCHING_THRESHOLD macro 151
NlsmVerify function 148
nlssContinuePending enumeration member 1911
nlssNotInstalled enumeration member 1911
nlssPaused enumeration member 1911
nlssPausePending enumeration member 1911
nlssRunning enumeration member 1911
nlssStartPending enumeration member 1911
nlssStopped enumeration member 1911
nlssStopPending enumeration member 1911
NLT_MAX_RECORD_COUNT macro 166
NLT_PROCESS_FIRST_RECORD_ONLY macro 166

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NLTemplate class 1730
about NLTemplate class 1730
CalculateSize 1734
Check 1735
FlagProcessFirstRecordOnly 1734
GetRecordCount 1735
GetSize 1736
MaxRecordCount 1734
NativeType 1736
NLTemplate 1733, 1734
Records 1736
NLTemplate Unit 153
NLTemplate.h 170
NLTemplate.RecordCollection class 1734
about NLTemplate.RecordCollection class 1734
NLTemplateAddRecord function 154
NLTemplateAddRecordCopy macro 166
NLTemplateAddRecordCopyEx function 155
NLTemplateAddRecordEx function 156
NLTemplateAddRecordFromMemory macro 167
NLTemplateAddRecordFromMemoryEx function 154
NLTemplateCalculateSize function 156
NLTemplateCheck function 157
NLTemplateClearRecords function 157
NLTemplateClone macro 168
NLTemplateCreate macro 168
NLTemplateCreateEx function 159
NLTemplateCreateFromMemory function 158
NLTemplateGetRecord function 159
NLTemplateGetRecordCapacity function 160
NLTemplateGetRecordCount function 160
NLTemplateGetRecordCountMem function 161
NLTemplateGetRecordEx function 162
NLTemplateGetSize macro 168
NLTemplateGetSizeMem function 163
NLTemplateInsertRecord function 164
NLTemplatePack function 161
NLTemplateRemoveRecord function 163
NLTemplateSaveToMemory macro 169
NLTemplateSetRecord function 165
NLTemplateSetRecordCapacity function 164
NLTemplateUnpack function 165
NLView class 1787
about NLView class 1787
BaseFeaturePointSize 1787
DetectionDetails 1788
DrawConfidenceForEyes 1788
DrawFaceConfidence 1788
FaceIds 1788
FaceRectangleColor 1788
FaceRectangleWidth 1789
FeaturePointSize 1789
Image 1789
NLView 1787
ShowBaseFeaturePoints 1789
ShowExpression 1789
ShowExpressionConfidence 1790
ShowEyes 1790
ShowEyesConfidence 1790
ShowFaceConfidence 1790
ShowGender 1790
ShowGenderConfidence 1790
ShowMouth 1791
ShowMouthConfidence 1791
ShowNose 1791
ShowNoseConfidence 1791
ShowProperties 1791
ShowPropertiesConfidence 1792
NM_FINGERS_MODE_ATMEL_FINGERCHIP macro 188
NM_FINGERS_MODE_AUTHENTEC_AES2501B macro 189
NM_FINGERS_MODE_AUTHENTEC_AES4000 macro 189
NM_FINGERS_MODE_AUTHENTEC_AFS2 macro 189
NM_FINGERS_MODE_BIOMETRIKA_FX2000 macro 189
NM_FINGERS_MODE_BIOMETRIKA_FX3000 macro 189
NM_FINGERS_MODE_BMF_BLP100 macro 189
NM_FINGERS_MODE_CROSSMATCH_VERIFIER300 macro 189
NM_FINGERS_MODE_DIGENT_IZZIX macro 189
NM_FINGERS_MODE_DIGITALPERSONA_UAREU macro 189
NM_FINGERS_MODE_ETHENTICA macro 190
NM_FINGERS_MODE_FUJITSU_MBF200 macro 190
NM_FINGERS_MODE_FUTRONIC_FS80 macro 190

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NM_FINGERS_MODE_GENERAL macro 190
NM_FINGERS_MODE_IDENTICATORTECHNOLOGY_DF90 macro 190
NM_FINGERS_MODE_IDENTIX_DFR2090 macro 190
NM_FINGERS_MODE_IDENTIX_TOUCHVIEW macro 190
NM_FINGERS_MODE_KEYTRONIC_SECUREDESKTOP macro 190
NM_FINGERS_MODE_LIGHTUNING_LTTC500 macro 190
NM_FINGERS_MODE_NITGEN_FINGKEY_HAMSTER macro 191
NM_FINGERS_MODE_PRECISEBIOMETRICS_100CS macro 191
NM_FINGERS_MODE_SECUGEN_HAMSTER macro 191
NM_FINGERS_MODE_STARTEK_FM200 macro 191
NM_FINGERS_MODE_TACOMA_CMOS macro 191
NM_FINGERS_MODE_TESTECH_BIOI macro 191
NM_FINGERS_MODE_UPEK_TOUCHCHIP macro 191
NM_PART_FACES macro 191
NM_PART_FINGERS macro 192
NM_PART_IRISES macro 192
NM_PART_NONE macro 192
NM_PART_PALMS macro 192
NMatcher class 1737
- about NMatcher class 1737
- FacesMatchingSpeed 1754
- FacesMatchingThresholdNew 1754
- FingersMatchingSpeed 1754
- FingersMaximalRotation 1754
- FingersMinMatchedCount 1754
- FingersMinMatchedCountThreshold 1755
- FingersMode 1755
- FingersModeAtmelFingerChip 1743
- FingersModeAuthentecAes2501B 1743
- FingersModeAuthentecAes4000 1743
- FingersModeAuthentecAFS2 1743
- FingersModeBiometrikaFX2000 1743
- FingersModeBiometrikaFX3000 1743
- FingersModeBmfBlp100 1743
- FingersModeCrossMatchVerifier300 1743
- FingersModeDigentIzzix 1743
- FingersModeDigitalPersonaUareU 1744
- FingersModeEthentica 1744
- FingersModeFujitsuMbf200 1744
- FingersModeFutronicFS80 1744
- FingersModeGeneral 1744
- FingersModeIdenticatorTechnologyDF90 1744
- FingersModeIdentixDfr2090 1744
- FingersModeIdentixTouchView 1744
- FingersModeKeytronicSecureDesktop 1744
- FingersModeLighTuningLttC500 1745
- FingersModeNitgenFingkeyHamster 1745
- FingersModePreciseBiometrics100CS 1745
- FingersModeSecugenHamster 1745
- FingersModeStartekFM200 1745
- FingersModeTacomaCmos 1745
- FingersModeTestechBioI 1745
- FingersModeUpekTouchChip 1745
- FusionType 1755
- IdentifyEnd 1749
- IdentifyNext 1750, 1751
- IdentifyStart 1751, 1752
- IrisesMatchingSpeed 1755
- IrisesMatchingThreshold 1755
- IrisesMaximalRotation 1755
- IrisesMinMatchedCount 1756
- IrisesMinMatchedCountThreshold 1756
- MatchingThreshold 1756
- NativeType 1756
- NMatcher 1742
- NMatchingSpeedNativeType 1756
- NMFusionTypeNativeType 1756
- PalmsMatchingSpeed 1756
- PalmsMaximalRotation 1756
- PalmsMinMatchedCount 1757
- PalmsMinMatchedCountThreshold 1757
- ParameterFacesMatcherBase 1745
- ParameterFacesMatchingSpeed 1746
- ParameterFacesMatchingThresholdNew 1746
- ParameterFingersFillFullMatchingDetails 1746
- ParameterFingersMatcherBase 1746
- ParameterFingersMatchingSpeed 1746
- ParameterFingersMaximalRotation 1746
- ParameterFingersMinMatchedCount 1746

Copyright © 2009-2014 Neurotechnology. All rights reserved.

ParameterFingersMinMatchedCountThreshold 1746
ParameterFingersMode 1747
ParameterFusionType 1747
ParameterIrisesMatcherBase 1747
ParameterIrisesMatchingSpeed 1747
ParameterIrisesMatchingThreshold 1747
ParameterIrisesMaximalRotation 1747
ParameterIrisesMinMatchedCount 1747
ParameterIrisesMinMatchedCountThreshold 1747
ParameterMatchingThreshold 1748
ParameterPalmsFillFullMatchingDetails 1748
ParameterPalmsMatcherBase 1748
ParameterPalmsMatchingSpeed 1748
ParameterPalmsMaximalRotation 1748
ParameterPalmsMinMatchedCount 1748
ParameterPalmsMinMatchedCountThreshold 1748
ParameterVoicesMatcherBase 1749
ParameterVoicesUniquePhrasesOnly 1749
PartFaces 1749
PartFingers 1749
PartIrises 1749
PartNone 1749
PartPalms 1749
PrepareTemplate 1752
Verify 1752, 1753
VoicesUniquePhrasesOnly 1757
NMatcher Unit 171
Macros 185
NMatcher.h 196
NMatcherParams.h 197
NMatchers Unit 200
NMatchers.h 200
NMatchingDetails class 1757
about NMatchingDetails class 1757
Faces 1760
FacesMatchedIndex 1760
FacesScore 1760
Fingers 1760
FingersScore 1760
Irises 1761
IrisesScore 1761
NativeType 1761
NMatchingDetails 1758, 1759
Palms 1761
PalmsScore 1761
Voices 1761
VoicesScore 1762
NMatchingDetails Unit 200
NMatchingDetails.FaceCollection class 1759
about NMatchingDetails.FaceCollection class 1759
NMatchingDetails.FingerCollection class 1759
about NMatchingDetails.FingerCollection class 1759
NMatchingDetails.h 218
NMatchingDetails.IrisCollection class 1759
about NMatchingDetails.IrisCollection class 1759
NMatchingDetails.PalmCollection class 1760
about NMatchingDetails.PalmCollection class 1760
NMatchingDetails.VoiceCollection class 1760
about NMatchingDetails.VoiceCollection class 1760
NMatchingDetailsCreateFromMemory function 201
NMatchingDetailsCreateFromMemoryN function 202
NMatchingDetailsCreateFromStream function 202
NMatchingDetailsGetFace function 203
NMatchingDetailsGetFaceCount function 203
NMatchingDetailsGetFaceEx function 204
NMatchingDetailsGetFacesMatchedIndex function 204
NMatchingDetailsGetFacesScore function 205
NMatchingDetailsGetFinger function 206
NMatchingDetailsGetFingerCount function 206
NMatchingDetailsGetFingerEx function 207
NMatchingDetailsGetFingersScore function 207
NMatchingDetailsGetId function 208
NMatchingDetailsGetIris function 209
NMatchingDetailsGetIrisCount function 209
NMatchingDetailsGetIrisesScore function 210
NMatchingDetailsGetIrisEx function 211
NMatchingDetailsGetPalm function 211
NMatchingDetailsGetPalmCount function 212
NMatchingDetailsGetPalmEx function 212
NMatchingDetailsGetPalmsScore function 213
NMatchingDetailsGetVoice function 214
NMatchingDetailsGetVoiceCount function 215

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMatchingDetailsGetVoiceEx function 215
NMatchingDetailsGetVoicesScore function 215
NMatchingDetailsSaveToMemory function 217
NMatchingDetailsSaveToStream macro 217
NMatchingSpeed enumeration 184
NMatchingSpeed_ enumeration 184
NMCreate function 174
NMedia Library 1196
NMediaFormat Unit 1334
NMediaFormat.h 1336
NMediaFormatGetMediaSubtype function 1334
NMediaFormatGetMediaType function 1334
NMediaFormatIsCompatibleWith function 1335
NMediaFormatSetMediaSubtype function 1335
NMediaReader Unit 1336
NMediaReader.h 1344
NMediaReaderCreate function 1337
NMediaReaderCreateFromFile function 1338
NMediaReaderCreateFromFile macro 1344
NMediaReaderCreateFromUrl function 1338
NMediaReaderCreateFromUrl macro 1344
NMediaReaderGetLength function 1339
NMediaReaderGetPosition function 1340
NMediaReaderGetSource function 1340
NMediaReaderGetState function 1340
NMediaReaderIsLive function 1341
NMediaReaderPause function 1342
NMediaReaderReadAudioSample function 1341
NMediaReaderReadVideoSample function 1342
NMediaReaderSetPosition function 1342
NMediaReaderStart function 1343
NMediaReaderStop function 1343
NMediaSource Unit 1345
NMediaSource.h 1351
NMediaSourceCreateFromFile function 1346
NMediaSourceCreateFromFile macro 1350
NMediaSourceCreateFromUrl function 1346
NMediaSourceCreateFromUrl macro 1350
NMediaSourceEnumDevices function 1347
NMediaSourceGetCurrentFormat function 1347
NMediaSourceGetDisplayName function 1348
NMediaSourceGetDisplayName macro 1350
NMediaSourceGetFormats function 1348
NMediaSourceGetId function 1349
NMediaSourceGetId macro 1350
NMediaSourceGetMediaType function 1349
NMediaSourceSetCurrentFormat function 1349
NMediaState enumeration 1354
NMediaState_ enumeration 1354
NMediaType enumeration 1354
NMediaType_ enumeration 1354
NMediaTypeIsValid function 1353
NMediaTypeIsValidSingle function 1353
NMediaTypes Unit 1352
NMediaTypes.h 1355
NMemberDescriptor Unit 501
NMemberDescriptor.h 501
NMemberDescriptorGetComponentType function 501
NMemory Unit 502
NMemory.h 509
NMemoryStream class 1837
- about NMemoryStream class 1837
- Capacity 1843
- GetBuffer 1843
- LongCapacity 1843
- NativeCapacity 1843
- NativeType 1843
- NMemoryStream 1841, 1842
- PositionPtr 1844
- ToArray 1843
- WriteTo 1843

NMemoryStream Unit 510
NMemoryStreamCreate function 511
NMemoryStreamCreateFromBuffer function 511
NMemoryStreamCreateFromBufferForRead macro 515
NMemoryStreamCreateFromBufferN function 512
NMemoryStreamCreateWithCapacity function 512
NMemoryStreamDetachBuffer function 512
NMemoryStreamGetBuffer function 513
NMemoryStreamGetCapacity function 513
NMemoryStreamGetPositionPtr function 513
NMemoryStreamSetCapacity function 514

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NMemoryStreamSetPositionPtr function 514
NMemoryStreamToPtr function 514
NMemoryStreamWriteTo function 515
NMemoryType enumeration 509
NMemoryType_ enumeration 509
NMemoryTypeIsValid function 507
NMemoryTypeIsValidForFree function 507
nmftFuseAlways enumeration member 184
nmftSelectByFaceThenFuse enumeration member 184
nmftSelectByIrisThenFuse enumeration member 184
NMFusionType enumeration 184
NMFusionType_ enumeration 184
NMIdentifyEnd function 177
NMIdentifyNext function 177
NMIdentifyNextEx function 175
NMIdentifyStart function 178
NMIdentifyStartEx function 176
NMMatchDetails structure 185
NMMatchDetails.h 199
NMMatchDetails_ structure 185
NMMatchDetailsDeserialize function 180
NMMatchDetailsDeserializeFromStream function 180
NMMatchDetailsFree function 181
NMMatchDetailsSerialize function 182
NMMatchDetailsSerializeToStream function 183
nmmtAligned enumeration member 509
nmmtCustom enumeration member 509
nmmtDefault enumeration member 509
nmmtNone enumeration member 509
nmmtStandard enumeration member 509
nmmtWin32Global enumeration member 509
nmmtWin32Local enumeration member 509
nmoDebug enumeration member 526
NModule class 1543
about NModule class 1543
Activated 1547
CheckInit 1546
Company 1547
Copyright 1547
GetDependences 1546
GetLoadedModules 1547
Name 1547
NativeType 1547
NModule 1546
NModuleOptionsNativeType 1548
Options 1548
Product 1548
Title 1548
Version 1548
VersionBuild 1548
VersionMajor 1548
VersionMinor 1548
VersionRevision 1548
NModule Unit 516
NModuleCheckInit function 517
NModuleCreate function 517
NModuleGetActivated function 518
NModuleGetActivated macro 527
NModuleGetCompany function 518
NModuleGetCompany macro 527
NModuleGetCopyright function 519
NModuleGetCopyright macro 527
NModuleGetDependences function 519
NModuleGetName function 519
NModuleGetName macro 527
NModuleGetOptions function 520
NModuleGetProduct function 520
NModuleGetProduct macro 527
NModuleGetTitle function 521
NModuleGetTitle macro 527
NModuleGetVersionBuild function 521
NModuleGetVersionMajor function 521
NModuleGetVersionMinor function 522
NModuleGetVersionRevision function 522
NModuleInternal Unit 529
NModuleOptions enumeration 526
NModuleOptions_ enumeration 526
NModuleRegister function 529
NModuleSetActivatedProc function 529
NModuleSetCompany macro 528
NModuleSetCompanyN function 522
NModuleSetCopyright macro 528

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NModuleSetCopyrightN function 523
NModuleSetDependences function 530
NModuleSetInit function 530
NModuleSetLazyInit function 530
NModuleSetName macro 528
NModuleSetNameN function 523
NModuleSetOptions function 523
NModuleSetProduct macro 528
NModuleSetProductN function 524
NModuleSetThreadInit function 531
NModuleSetThreadUninit function 531
NModuleSetTitle macro 528
NModuleSetTitleN function 524
NModuleSetUninit function 531
NModuleSetVersionBuild function 524
NModuleSetVersionMajor function 525
NModuleSetVersionMinor function 525
NModuleSetVersionRevision function 525
NModuleUnregister function 532
nmoExe enumeration member 526
nmoLib enumeration member 526
NMonitor Unit 532
NMonitorDispose function 532
NMonitorEnter function 533
NMonitorExit function 533
NMonitorInit function 533
NMonitorTryEnter function 534
nmoNoAnsiFunc enumeration member 526
NMonochromeImage Unit 1355
NMonochromeImage.h 1358
NMonochromeImageGetMaxValue function 1356
NMonochromeImageGetMinValue function 1356
NMonochromeImageGetPixel function 1356
NMonochromeImageSetMaxValue function 1357
NMonochromeImageSetMinValue function 1357
NMonochromeImageSetPixel function 1358
nmoNone enumeration member 526
nmoNoUnicode enumeration member 526
nmoProtected enumeration member 526
nmoUnicode enumeration member 526
NMove function 508
NMP_FACES_MATCHER_BASE macro 192
NMP_FACES_MATCHING_SPEED macro 192
NMP_FACES_MATCHING_THRESHOLD macro 192
NMP_FACES_MATCHING_THRESHOLD_NEW macro 193
NMP_FINGERS_FILL_FULL_MATCHING_DETAILS macro 193
NMP_FINGERS_MATCHER_BASE macro 193
NMP_FINGERS_MATCHING_SPEED macro 193
NMP_FINGERS_MAXIMAL_ROTATION macro 193
NMP_FINGERS_MIN_MATCHED_COUNT macro 193
NMP_FINGERS_MIN_MATCHED_COUNT_THRESHOLD macro 193
NMP_FINGERS_MODE macro 194
NMP_FUSION_TYPE macro 194
NMP_IRISES_MATCHER_BASE macro 194
NMP_IRISES_MATCHING_SPEED macro 194
NMP_IRISES_MATCHING_THRESHOLD macro 194
NMP_IRISES_MAXIMAL_ROTATION macro 194
NMP_IRISES_MIN_MATCHED_COUNT macro 194
NMP_IRISES_MIN_MATCHED_COUNT_THRESHOLD macro 195
NMP_MATCHING_THRESHOLD macro 195
NMP_PALMS_FILL_FULL_MATCHING_DETAILS macro 195
NMP_PALMS_MATCHER_BASE macro 195
NMP_PALMS_MATCHING_SPEED macro 195
NMP_PALMS_MAXIMAL_ROTATION macro 195
NMP_PALMS_MIN_MATCHED_COUNT macro 196
NMP_PALMS_MIN_MATCHED_COUNT_THRESHOLD macro 196
NMP_VOICES_MATCHER_BASE macro 196
NMP_VOICES_UNIQUE_PHRASES_ONLY macro 196
NMPrepareTemplate function 179
NMS_ALLOW_DUPLICATE_DEVICES macro 1351
NMS_DO_NOT_USE_DIRECT_SHOW macro 1351
NMS_DO_NOT_USE_WINDOWS_MEDIA_FOUNDATION macro 1351
NMS_PREFER_DIRECT_SHOW macro 1351
nmsHigh enumeration member 184
nmsLow enumeration member 184
nmsMedium enumeration member 184
nmsPaused enumeration member 1354
nmsRunning enumeration member 1354
nmsStopped enumeration member 1354

Copyright © 2009-2014 Neurotechnology. All rights reserved.

nmtAudio enumeration member 1354
nmtNone enumeration member 1354
nmtVideo enumeration member 1354
NMutex Unit 534
NMutexCreate function 535
NMutexCreateFromOSHandle function 535
NMutexGetOSHandle function 535
NMutexRelease function 536
NMVerify function 181
NMVerifyEx function 179
NNameStringPair type 1007
NNameStringPair_ structure 1001
NNameStringPairCopy function 959
NNameStringPairCreate function 959
NNameStringPairDispose function 959
NNameStringPairSet function 960
NNameValuePair_ structure 1119
NNameValuePairCopy function 1097
NNameValuePairCreate function 1097
NNameValuePairCreate macro 1120
NNameValuePairDispose function 1098
NNameValuePairSet function 1098
NNoDeprecate Unit 536
NoAnsiFunc enumeration member 1651
NObject class 1549
about NObject class 1549
Clone 1551
CompareTo 1551
CopyParameters 1552
CopyProperties 1552
DisposeArray 1552
Equals 1552
EventInvoker 1568
FlagRefRet 1551
Flags 1567
FromHandle 1552, 1553
FromHandlePlain 1568
FromObject 1554
GetArray 1554
GetArrayPlain 1568
GetArrayWithHandle 1569
GetHashCode 1555
GetNativeType 1555
GetObject 1555
GetObjectPlain 1569
GetObjects 1556
GetObjectWithHandle 1569
GetOutArrayPlain 1569
GetOutArrayWithHandle 1570
GetParameter 1557
GetProperty 1558
GetSize 1558
GetSizeMany 1558, 1559
GetString 1559
GetStringPlainN 1570
GetStringWithHandleN 1570
Invoke 1559
IsDisposed 1567
NativeType 1568
Owner 1568
PropertyChanged 1572
Ref 1560
Reset 1560
ResetProperty 1560
Save 1560, 1561
SaveMany 1562
SetArray 1563
SetArrayPlain 1571
SetArrayWithHandle 1571
SetObject 1563
SetObjectPlain 1571
SetObjectWithHandle 1571
SetParameter 1563, 1564
SetProperty 1565
SetString 1565
SetStringPlainN 1572
SetStringWithHandleN 1572
ToHandle 1566
ToObject 1566
ToString 1566
Unref 1567
UnrefArray 1567

Copyright © 2009-2014 Neurotechnology. All rights reserved.

UnrefElements 1567
NObject Unit 536
NObject.h 566
NObjectAddDisposed function 539
NObjectAddDisposedCallback function 539
NObjectAddOwnerChanged function 539
NObjectAddOwnerChangedCallback function 540
NObjectAddPropertyChanged function 540
NObjectAddPropertyChangedCallback function 540
NObjectClone function 541
NObjectCompareTo function 541
NObjectCopyParameters macro 561
NObjectCopyProperties function 542
NObjectEquals function 542
NObjectFree function 542
NObjectFreeArray function 543
NObjectFreeElements function 543
NObjectGet function 543
NObjectGetArray function 543
NObjectGetConcurrent function 544
NObjectGetFlags function 544
NObjectGetHashCode function 545
NObjectGetOwner function 545
NObjectGetOwnerEx function 545
NObjectGetParameter macro 562
NObjectGetParameterEx macro 562
NObjectGetParameterWithPart macro 563
NObjectGetParameterWithPartEx function 546
NObjectGetParameterWithPartExN function 546
NObjectGetPropertyN function 547
NObjectGetPropertyNN function 547
NObjectGetPropertyP function 548
NObjectGetPropertyP macro 563
NObjectGetSize function 548
NObjectGetSizeMany function 549
NObjectGetType function 549
NObjectRef function 549
NObjectRemoveDisposed function 550
NObjectRemoveDisposedCallback function 550
NObjectRemoveOwnerChanged function 550
NObjectRemoveOwnerChangedCallback function 551
NObjectRemovePropertyChanged function 551
NObjectRemovePropertyChangedCallback function 551
NObjectReset function 552
NObjectResetProperty function 552
NObjectResetProperty macro 564
NObjectSaveManyToMemoryDst function 552
NObjectSaveManyToMemoryDstN function 553
NObjectSaveManyToMemoryN function 553
NObjectSaveManyToStream function 554
NObjectSaveToMemoryDst function 554
NObjectSaveToMemoryDstN function 554
NObjectSaveToMemoryN function 555
NObjectSaveToStream function 555
NObjectSet function 556
NObjectSetConcurrent function 556
NObjectSetFlags function 556
NObjectSetParameter macro 564
NObjectSetParameterEx macro 564
NObjectSetParameterWithPart macro 565
NObjectSetParameterWithPartEx function 557
NObjectSetParameterWithPartExN function 557
NObjectSetPropertyN function 558
NObjectSetPropertyNN function 558
NObjectSetPropertyP function 559
NObjectSetPropertyP macro 566
NObjectToString function 559
NObjectToString macro 566
NObjectUnref function 560
NObjectUnrefArray function 560
NObjectUnrefElements function 561
NoEar enumeration member 1780
NoLens enumeration member 1823
None enumeration member 1651, 1652, 1769, 1770, 1771, 1774, 1778, 1779, 1823, 1824, 1882
NonliveScanPalm enumeration member 1773
NonliveScanPlain enumeration member 1773
NonliveScanRolled enumeration member 1773
NOSFamily enumeration 1003
NOSFamily_ enumeration 1003
NOSFamilyGetCurrent function 960
nosfAndroid enumeration member 1003
nosfEmbeddedLinux enumeration member 1003

Copyright © 2009-2014 Neurotechnology. All rights reserved.

nosfIOS enumeration member 1003
nosfLinux enumeration member 1003
nosfMacOSX enumeration member 1003
nosfNone enumeration member 1003
nosfUnix enumeration member 1003
nosfWindows enumeration member 1003
nosfWindowsCE enumeration member 1003
nosfWindowsPhone enumeration member 1003
NotInstalled enumeration member 1862
NotRecognized enumeration member 1882
NotSpecified enumeration member 1780
NoUnicode enumeration member 1651
NParameterBag Unit 569
NParameterBag.h 575
NParameterBagApplyPropertyBag function 569
NParameterBagContains function 570
NParameterBagContains macro 574
NParameterBagCreate function 570
NParameterBagGet function 571
NParameterBagGet macro 574
NParameterBagGetAt function 571
NParameterBagGetCount function 571
NParameterBagGetKey function 572
NParameterBagGetValue function 572
NParameterBagSet function 572
NParameterBagSet macro 574
NParameterBagSetValue function 573
NParameterBagToPropertyBag function 573
NParameterBagTryGet function 573
NParameterBagTryGet macro 574
NParameterDescriptor Unit 579
NParameterDescriptor.h 582
NParameterDescriptorGetDefaultValue function 580
NParameterDescriptorGetFormat function 580
NParameterDescriptorGetMaxValue function 580
NParameterDescriptorGetMinValue function 581
NParameterDescriptorGetParameterType function 581
NParameterDescriptorGetStdValue function 581
NParameterDescriptorGetStdValueCount function 582
NParameterMakeId macro 578
NParameters Unit 575
NParameters.h 579
NPath Unit 583
NPath.h 593
NPathChangeExtension function 584
NPathChangeExtension macro 591
NPathChangeExtensionPN function 585
NPathChangeExtensionPN macro 591
NPathCheck function 585
NPathCheck macro 591
NPathCheckN function 585
NPathCombine function 586
NPathCombine macro 591
NPathCombineP1N function 586
NPathCombineP1N macro 591
NPathCombineP2N function 586
NPathCombineP2N macro 592
NPathGetDirectoryName function 587
NPathGetDirectoryName macro 592
NPathGetExtension function 587
NPathGetExtension macro 592
NPathGetFileName function 588
NPathGetFileName macro 592
NPathGetFileNameWithoutExtension function 588
NPathGetFileNameWithoutExtension macro 592
NPathGetTempFileName function 588
NPathGetTempPath function 589
NPathHasExtension function 589
NPathHasExtension macro 593
NPathIsRooted function 589
NPathIsRooted macro 593
NPF_GRAYSCALE_16S macro 1370
NPF_GRAYSCALE_16U macro 1370
NPF_GRAYSCALE_1U macro 1370
NPF_GRAYSCALE_2U macro 1370
NPF_GRAYSCALE_32F macro 1371
NPF_GRAYSCALE_32S macro 1371
NPF_GRAYSCALE_32U macro 1371
NPF_GRAYSCALE_4U macro 1371
NPF_GRAYSCALE_64F macro 1371
NPF_GRAYSCALE_64S macro 1372
NPF_GRAYSCALE_64U macro 1372

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPF_GRAYSCALE_8S macro 1372
NPF_GRAYSCALE_8U macro 1372
NPF_GRAYSCALE_A_16S macro 1372
NPF_GRAYSCALE_A_16U macro 1373
NPF_GRAYSCALE_A_32F macro 1373
NPF_GRAYSCALE_A_32S macro 1373
NPF_GRAYSCALE_A_32U macro 1373
NPF_GRAYSCALE_A_64F macro 1373
NPF_GRAYSCALE_A_64S macro 1374
NPF_GRAYSCALE_A_64U macro 1374
NPF_GRAYSCALE_A_8S macro 1374
NPF_GRAYSCALE_A_8U macro 1374
NPF_GRAYSCALE_PA_16S macro 1374
NPF_GRAYSCALE_PA_16U macro 1375
NPF_GRAYSCALE_PA_32F macro 1375
NPF_GRAYSCALE_PA_32S macro 1375
NPF_GRAYSCALE_PA_32U macro 1375
NPF_GRAYSCALE_PA_64F macro 1375
NPF_GRAYSCALE_PA_64S macro 1376
NPF_GRAYSCALE_PA_64U macro 1376
NPF_GRAYSCALE_PA_8S macro 1376
NPF_GRAYSCALE_PA_8U macro 1376
NPF_RGB_16S macro 1376
NPF_RGB_16U macro 1377
NPF_RGB_32F macro 1377
NPF_RGB_32S macro 1377
NPF_RGB_32U macro 1377
NPF_RGB_64F macro 1377
NPF_RGB_64S macro 1378
NPF_RGB_64U macro 1378
NPF_RGB_8S macro 1378
NPF_RGB_8U macro 1378
NPF_RGB_8U_INDEXED_1 macro 1378
NPF_RGB_8U_INDEXED_2 macro 1379
NPF_RGB_8U_INDEXED_4 macro 1379
NPF_RGB_8U_INDEXED_8 macro 1379
NPF_RGB_A_16S macro 1379
NPF_RGB_A_16U macro 1379
NPF_RGB_A_32F macro 1380
NPF_RGB_A_32S macro 1380
NPF_RGB_A_32U macro 1380
NPF_RGB_A_64F macro 1380
NPF_RGB_A_64S macro 1380
NPF_RGB_A_64U macro 1381
NPF_RGB_A_8S macro 1381
NPF_RGB_A_8U macro 1381
NPF_RGB_A_8U_INDEXED_1 macro 1381
NPF_RGB_A_8U_INDEXED_2 macro 1381
NPF_RGB_A_8U_INDEXED_4 macro 1382
NPF_RGB_A_8U_INDEXED_8 macro 1382
NPF_RGB_PA_16S macro 1382
NPF_RGB_PA_16U macro 1382
NPF_RGB_PA_32F macro 1382
NPF_RGB_PA_32S macro 1383
NPF_RGB_PA_32U macro 1383
NPF_RGB_PA_64F macro 1383
NPF_RGB_PA_64S macro 1383
NPF_RGB_PA_64U macro 1383
NPF_RGB_PA_8S macro 1384
NPF_RGB_PA_8U macro 1384
NPF_UNDEFINED macro 1384
npfGrayscale macro 1384
npfMonochrome macro 1384
npfRgb macro 1385
NPixelFormat type 1369
NPixelFormat Unit 1359
NPixelFormat.h 1385
NPixelFormat_ type 1370
NPixelFormatCalcRowSize function 1363
NPixelFormatCreate function 1364
NPixelFormatGetBitsPerPixel function 1364
NPixelFormatGetBytesPerPixel function 1365
NPixelFormatGetIndexed function 1365
NPixelFormatGetNonIndexed function 1365
NPixelFormatGetNonSeparated function 1366
NPixelFormatGetPaletteSize function 1366
NPixelFormatGetPixelType function 1366
NPixelFormatGetRowSize function 1367
NPixelFormatGetSeparated function 1367
NPixelFormatGetWithExtraChannel function 1367
NPixelFormatIsValid function 1368
NPixelFormatToString function 1368

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPixelFormatToString macro 1385
NPixelType enumeration 1369
NPixelType_ enumeration 1369
NPixelTypeCanBeIndexed function 1368
NPixelTypeIsValid function 1369
NPlugin class 1863
about NPlugin class 1863
Changed 1869
Changing 1869
Disable 1866
Enable 1867
FileName 1867
GetError 1867
LoadTime 1867
Module 1868
NativeType 1868
NPluginStateNativeType 1868
Owner 1868
Plug 1867
PlugTime 1868
SelectedInterfaceVersion 1868
State 1868
Unplug 1867
NPlugin Unit 594
NPlugin.h 604
NPluginAddChanged function 595
NPluginAddChangedCallback function 596
NPluginAddChanging function 596
NPluginAddChangingCallback function 596
NPluginDisable function 597
NPluginEnable function 597
NPluginGetError function 597
NPluginGetErrorCode function 598
NPluginGetFileName function 598
NPluginGetFileName macro 604
NPluginGetLoadTime function 598
NPluginGetModule function 599
NPluginGetPlugTime function 599
NPluginGetSelectedInterface function 605
NPluginGetSelectedInterfaceVersion function 600
NPluginGetState function 600
NPluginInterfaceVersionInfo structure 606
NPluginInterfaceVersionInfo_ structure 606
NPluginInterfaceVersionInfoConst macro 606
NPluginInternal Unit 605
NPluginManager class 1869
about NPluginManager class 1869
AllowsUnplug 1875
DisabledPlugins 1875
DisabledPluginsChanged 1876
DisabledPluginsChanging 1876
EnsurePluginSearchPerformed 1874
GetInstances 1874
GetInterfaceVersions 1874
InterfaceType 1875
NativeType 1875
PlugAll 1874
PluginAdded 1876
Plugins 1875
PluginsChanged 1876
PluginsChanging 1876
PluginSearchPath 1875
Refresh 1874
UnplugAll 1874
NPluginManager Unit 607
NPluginManager.DisabledPluginCollection class 1872
about NPluginManager.DisabledPluginCollection class 1872
NPluginManager.h 627
NPluginManager.PluginCollection class 1872
about NPluginManager.PluginCollection class 1872
Add 1873
IndexOf 1873
this 1873
NPluginManagerAddDisabledPlugin function 609
NPluginManagerAddDisabledPlugin macro 626
NPluginManagerAddDisabledPluginsChanged function 609
NPluginManagerAddDisabledPluginsChangedCallback function 610
NPluginManagerAddDisabledPluginsChanging function 610
NPluginManagerAddDisabledPluginsChangingCallback function 611
NPluginManagerAddPlugin function 611

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPluginManagerAddPlugin macro 626
NPluginManagerAddPluginAdded function 612
NPluginManagerAddPluginAddedCallback function 612
NPluginManagerAddPluginFromFile function 612
NPluginManagerAddPluginFromFile macro 626
NPluginManagerAddPluginsChanged function 613
NPluginManagerAddPluginsChangedCallback function 613
NPluginManagerAddPluginsChanging function 613
NPluginManagerAddPluginsChangingCallback function 614
NPluginManagerAllowsUnplug function 614
NPluginManagerClearDisabledPlugins function 615
NPluginManagerEnsurePluginSearchPerformed function 615
NPluginManagerGetDisabledPlugin function 616
NPluginManagerGetDisabledPlugin macro 626
NPluginManagerGetDisabledPluginCount function 616
NPluginManagerGetInstances function 617
NPluginManagerGetInterfaceType function 617
NPluginManagerGetInterfaceType macro 627
NPluginManagerGetInterfaceVersions function 618
NPluginManagerGetPlugin function 618
NPluginManagerGetPluginByName function 619
NPluginManagerGetPluginCount function 619
NPluginManagerGetPluginSearchPath function 619
NPluginManagerGetPluginSearchPath macro 627
NPluginManagerPlugAll function 620
NPluginManagerPluginEventArgs class 1876
about NPluginManagerPluginEventArgs class 1876
NPluginManagerPluginEventArgs 1877
Plugin 1877
NPluginManagerRefresh function 620
NPluginManagerRemoveDisabledPlugin function 621
NPluginManagerRemoveDisabledPlugin macro 627
NPluginManagerRemoveDisabledPluginAt function 621
NPluginManagerRemoveDisabledPluginsChanged function 621
NPluginManagerRemoveDisabledPluginsChangedCallback function 622
NPluginManagerRemoveDisabledPluginsChanging function 622
NPluginManagerRemoveDisabledPluginsChangingCallback function 622
NPluginManagerRemovePluginAdded function 623
NPluginManagerRemovePluginAddedCallback function 623
NPluginManagerRemovePluginsChanged function 624
NPluginManagerRemovePluginsChangedCallback function 624
NPluginManagerRemovePluginsChanging function 624
NPluginManagerRemovePluginsChangingCallback function 625
NPluginManagerSetPluginSearchPath function 625
NPluginManagerSetPluginSearchPath macro 627
NPluginManagerUnplugAll function 625
NPluginModule class 1877
about NPluginModule class 1877
GetInterfaceVersions 1881
IncompatiblePlugins 1881
InterfaceType 1881
NativeType 1881
NPluginModule 1880
PluginName 1881
Priority 1882
NPluginModule Unit 629
NPluginModuleCreate function 631
NPluginModuleGetIncompatiblePlugins function 630
NPluginModuleGetIncompatiblePlugins macro 634
NPluginModuleGetInterfaceType function 631
NPluginModuleGetInterfaceType macro 634
NPluginModuleGetInterfaceVersions function 631
NPluginModuleGetPluginName function 632
NPluginModuleGetPluginName macro 635
NPluginModuleGetPriority function 634
NPluginModuleInternal Unit 635
NPluginModuleSetIncompatiblePlugins macro 635
NPluginModuleSetIncompatiblePluginsN function 632
NPluginModuleSetInterfaceType macro 635
NPluginModuleSetInterfaceTypeN function 633
NPluginModuleSetInterfaceVersions function 636
NPluginModuleSetPlug function 636
NPluginModuleSetPluginName macro 635
NPluginModuleSetPluginNameN function 633
NPluginModuleSetPriority function 633
NPluginModuleSetUnplug function 636
NPluginPlug function 600
NPluginRemoveChanged function 601
NPluginRemoveChangedCallback function 601

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NPluginRemoveChanging function 601
NPluginRemoveChangingCallback function 602
NPluginState enumeration 603
NPluginState_ enumeration 603
NPluginUnplug function 602
NPoint type 1226
NPoint_ structure 1223
NPointD type 1227
NPointD_ structure 1224
NPointerParse macro 1052
NPointerParseChars macro 1053
NPointerParseN function 960
NPointerParseStrOrChars function 961
NPointerParseStrOrChars macro 1053
NPointerParseVN function 961
NPointerParseVN macro 1053
NPointerToChars function 961
NPointerToChars macro 1053
NPointerToString function 962
NPointerToString macro 1054
NPointerTryParse macro 1054
NPointerTryParseChars macro 1054
NPointerTryParseN function 962
NPointerTryParseStrOrChars function 963
NPointerTryParseStrOrChars macro 1055
NPointerTryParseVN function 963
NPointerTryParseVN macro 1055
NPointF type 1227
NPointF_ structure 1224
NProcessorInfo class 1572
about NProcessorInfo class 1572
Count 1573
GetModelInfo 1573
Is3DNowSupported 1574
IsLZCntSupported 1574
IsMmxSupported 1574
IsNeonSupported 1574
IsPopCntSupported 1574
IsSse2Supported 1574
IsSse3Supported 1574
IsSse41Supported 1574
IsSse42Supported 1574
IsSse4aSupported 1575
IsSse5Supported 1575
IsSseSupported 1575
IsSsse3Supported 1575
ModelName 1575
NativeType 1575
NProcessorVendorNativeType 1575
Vendor 1575
VendorName 1575
NProcessorInfo Unit 637
NProcessorInfo.h 649
NProcessorInfoGetCount function 638
NProcessorInfoGetModelInfo function 639
NProcessorInfoGetModelNameEx function 639
NProcessorInfoGetModelNameEx macro 649
NProcessorInfoGetModelNameN function 639
NProcessorInfoGetVendor function 640
NProcessorInfoGetVendorNameEx function 640
NProcessorInfoGetVendorNameEx macro 649
NProcessorInfoGetVendorNameN function 640
NProcessorInfoIs3DNowSupported function 641
NProcessorInfoIs3DNowSupportedEx function 641
NProcessorInfoIsLZCntSupported function 641
NProcessorInfoIsLZCntSupportedEx function 642
NProcessorInfoIsMmxSupported function 642
NProcessorInfoIsMmxSupportedEx function 642
NProcessorInfoIsNeonSupported function 642
NProcessorInfoIsPopCntSupported function 643
NProcessorInfoIsPopCntSupportedEx function 643
NProcessorInfoIsSse2Supported function 643
NProcessorInfoIsSse2SupportedEx function 644
NProcessorInfoIsSse3Supported function 644
NProcessorInfoIsSse3SupportedEx function 644
NProcessorInfoIsSse41Supported function 645
NProcessorInfoIsSse41SupportedEx function 645
NProcessorInfoIsSse42Supported function 645
NProcessorInfoIsSse42SupportedEx function 645
NProcessorInfoIsSse4aSupported function 646
NProcessorInfoIsSse4aSupportedEx function 646
NProcessorInfoIsSse5Supported function 646

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NProcessorInfoIsSse5SupportedEx function 647
NProcessorInfoIsSseSupported function 647
NProcessorInfoIsSseSupportedEx function 647
NProcessorInfoIsSsse3Supported function 648
NProcessorInfoIsSsse3SupportedEx function 648
NProcessorVendor enumeration 648
NProcessorVendor_ enumeration 648
NPropertyBag class 1576
about NPropertyBag class 1576
Add 1580
AddRange 1580
ApplyTo 1580
Clear 1581
Contains 1581
ContainsKey 1581
CopyTo 1581
Count 1584
GetEnumerator 1581
IsReadOnly 1585
Keys 1585
NativeType 1585
NPropertyBag 1579
Parse 1582
Remove 1582
RemoveAt 1583
this 1583
TryGetValue 1583
TryParse 1584
Values 1585
NPropertyBag.KeyCollection class 1579
about NPropertyBag.KeyCollection class 1579
Contains 1579
NPropertyBag.ValueCollection class 1580
about NPropertyBag.ValueCollection class 1580
npsDisabled enumeration member 603
npsDuplicate enumeration member 603
npsIncompatibleWithOtherPlugins enumeration member 603
npsInterfaceTypeMismatch enumeration member 603
npsInterfaceVersionMismatch enumeration member 603
npsInvalidInterface enumeration member 603
npsInvalidModule enumeration member 603
npsLoadError enumeration member 603
npsNone enumeration member 603
npsNotRecognized enumeration member 603
npsPlugged enumeration member 603
npsPluggingError enumeration member 603
npsUnplugged enumeration member 603
npsUnused enumeration member 603
nptGrayscale enumeration member 1369
nptRgb enumeration member 1369
nptUndefined enumeration member 1369
npvAmd enumeration member 648
npvCentaur enumeration member 648
npvCyrix enumeration member 648
npvIntel enumeration member 648
npvNationalSemiconductor enumeration member 648
npvNexGen enumeration member 648
npvRiseTechnology enumeration member 648
npvSiS enumeration member 648
npvTransmeta enumeration member 648
npvUmc enumeration member 648
npvUnknown enumeration member 648
NRange structure 1585
about NRange structure 1585
From 1586
NativeType 1586
NRange 1585
To 1586
ToString 1586
NRangeToString function 964
NRangeToString macro 1055
NRational structure 1587
about NRational structure 1587
Denominator 1588
Empty 1587
NativeType 1588
NRational 1587
Numerator 1588
ToString 1588
NRational type 1007
NRational_ structure 1002
NRationalToString function 964

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NRationalToString macro 1055
NReAlloc function 508
NRect type 1226
NRect_ structure 1222
NRectD type 1227
NRectD_ structure 1225
NRectF type 1227
NRectF_ structure 1225
NReDeprecate Unit 651
NResult type 1007
NResultParseN function 964
NResultParseStrOrChars function 965
NResultParseVN function 965
NResultParseVN macro 1055
NResultToChars function 966
NResultToChars macro 1055
NResultToString function 966
NResultToString macro 1056
NResultTryParseN function 967
NResultTryParseStrOrChars function 967
NResultTryParseStrOrChars macro 1056
NResultTryParseVN function 967
NResultTryParseVN macro 1056
NRgb type 1391
NRgb Unit 1389
NRgb.h 1391
NRgb_ structure 1390
NRgbImage Unit 1392
NRgbImage.h 1395
NRgbImageGetMaxValue function 1392
NRgbImageGetMinValue function 1392
NRgbImageGetPixel function 1393
NRgbImageSetMaxValue function 1393
NRgbImageSetMinValue function 1394
NRgbImageSetPixel function 1394
NRgbToString function 1390
NRgbToString macro 1391
NRWLock Unit 651
NRWLockDispose function 651
NRWLockEnterRead function 652
NRWLockEnterWrite function 652
NRWLockExitRead function 652
NRWLockExitWrite function 652
NRWLockInit function 653
NRWLockTryEnterRead function 653
NRWLockTryEnterWrite function 653
NSByte type 1008
NsedGetBestMatch function 1456
NsedGetBestMatchCount function 1456
NsedGetBestMatchesEx function 1443
NsedGetDetectionDetails function 1447
NsedGetExtractionStatus function 1444
NsedGetFaces function 1443
NsedGetFrameIndex function 1445
NsedGetGrayscaleImage function 1446
NsedGetOriginalImage function 1444
NsedGetSourceIndex function 1446
NsedGetSubjectTemplate function 1445
NsedGetTemplate function 1445
NsedGetTimeStamp function 1447
NsedGetTraceIndex function 1446
NsedMatch structure 1884
about NsedMatch structure 1884
NativeType 1884
Score 1884
TemplateId 1884
NSeekOrigin enumeration 497
NSeekOrigin_ enumeration 497
NSeekOriginIsValid function 495
NSemaphore Unit 654
NSemaphoreCreate function 654
NSemaphoreCreateFromOSHandle function 654
NSemaphoreGetOSHandle function 655
NSemaphoreRelease function 655
nsetAny enumeration member 1457
nsetFacesDetected enumeration member 1457
nsetImageCaptured enumeration member 1457
nsetSubjectAppeared enumeration member 1457
nsetSubjectDisappeared enumeration member 1457
nsetSubjectTrack enumeration member 1457
NShort type 1008
NSingle type 1008

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSingleGetNaN function 968
NSingleGetNegativeInfinity function 968
NSingleGetPositiveInfinity function 968
NSingleIsInfinity function 969
NSingleIsNaN function 969
NSingleIsNegativeInfinity function 969
NSingleIsPositiveInfinity function 970
NSingleParse macro 1056
NSingleParseChars macro 1056
NSingleParseN function 970
NSingleParseStrOrChars function 970
NSingleParseStrOrChars macro 1057
NSingleParseVN function 971
NSingleParseVN macro 1057
NSingleToChars function 971
NSingleToChars macro 1057
NSingleToString function 972
NSingleToString macro 1057
NSingleTryParse macro 1058
NSingleTryParseChars macro 1058
NSingleTryParseN function 972
NSingleTryParseStrOrChars function 972
NSingleTryParseStrOrChars macro 1058
NSingleTryParseVN function 973
NSingleTryParseVN macro 1059
NSize type 1226
NSize_ structure 1223
NSizeD type 1227
NSizeD_ structure 1225
NSizeF type 1227
NSizeF_ structure 1226
NSizeType type 1008
NSizeTypeParse macro 1059
NSizeTypeParseChars macro 1059
NSizeTypeParseN function 973
NSizeTypeParseStrOrChars function 974
NSizeTypeParseStrOrChars macro 1059
NSizeTypeParseVN function 974
NSizeTypeParseVN macro 1060
NSizeTypeToChars function 975
NSizeTypeToChars macro 1060
NSizeTypeToString function 975
NSizeTypeToString macro 1060
NSizeTypeTryParse macro 1060
NSizeTypeTryParseChars macro 1061
NSizeTypeTryParseN function 975
NSizeTypeTryParseStrOrChars function 976
NSizeTypeTryParseStrOrChars macro 1061
NSizeTypeTryParseVN function 976
NSizeTypeTryParseVN macro 1061
nsoBegin enumeration member 497
nsoCurrent enumeration member 497
nsoEnd enumeration member 497
NSP_DETECTION_DETECT_MORE_FACES macro 1458
NSP_DETECTION_FACE_CONFIDENCE_THRESHOLD macro 1463
NSP_DETECTION_FACE_DETECTION_SCALE_COUNT macro 1458
NSP_DETECTION_FACE_ROLL_ANGLE_BASE macro 1458
NSP_DETECTION_MAX_IOD macro 1463
NSP_DETECTION_MAX_ROLL_ANGLE_DEVIATION macro 1459
NSP_DETECTION_MAX_YAW_ANGLE_DEVIATION macro 1459
NSP_DETECTION_MIN_IOD macro 1463
NSP_EXTRACTION_BLINK_CONFIDENCE_THRESHOLD macro 1459
NSP_EXTRACTION_DARK_GLASSES_CONFIDENCE_THRESHOLD macro 1459
NSP_EXTRACTION_DETECT_BLINK macro 1459
NSP_EXTRACTION_DETECT_DARK_GLASSES macro 1460
NSP_EXTRACTION_DETECT_EXPRESSION macro 1460
NSP_EXTRACTION_DETECT_GENDER macro 1460
NSP_EXTRACTION_DETECT_GLASSES macro 1460
NSP_EXTRACTION_DETECT_MOUTH_OPEN macro 1460
NSP_EXTRACTION_ENROLLMENT_TEMPLATE_SIZE macro 1461
NSP_EXTRACTION_EXPRESSION_CONFIDENCE_THRESHOLD macro 1461
NSP_EXTRACTION_FACE_QUALITY_THRESHOLD macro 1463
NSP_EXTRACTION_GENDER_CONFIDENCE_THRESHOLD macro 1461

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NSP_EXTRACTION_GLASSES_CONFIDENCE_THRESHOLD
macro 1461

NSP_EXTRACTION_IDENTIFICATION_TEMPLATE_SIZE
macro 1461

NSP_EXTRACTION_MAX_RECORDS_PER_TEMPLATE
macro 1464

NSP_EXTRACTION_MOUTH_OPEN_CONFIDENCE_THRESHOLD
macro 1462

NSP_IDENTIFICATION_MATCHING_THRESHOLD macro 1464

NSP_IDENTIFICATION_MAX_BEST_MATCH_COUNT
macro 1462

NSP_MAX_THREAD_COUNT macro 1462

NSSizeType type 1009

NSSizeTypeParse macro 1061

NSSizeTypeParseChars macro 1062

NSSizeTypeParseN function 977

NSSizeTypeParseStrOrChars function 977

NSSizeTypeParseStrOrChars macro 1062

NSSizeTypeParseVN function 978

NSSizeTypeParseVN macro 1062

NSSizeTypeToChars function 978

NSSizeTypeToChars macro 1063

NSSizeTypeToString function 979

NSSizeTypeToString macro 1063

NSSizeTypeTryParse macro 1063

NSSizeTypeTryParseChars macro 1063

NSSizeTypeTryParseN function 979

NSSizeTypeTryParseStrOrChars function 979

NSSizeTypeTryParseStrOrChars macro 1064

NSSizeTypeTryParseVN function 980

NSSizeTypeTryParseVN macro 1064

NStopwatch Unit 656

NStopwatchCreate function 656

NStopwatchGetElapsed function 656

NStopwatchGetElapsedMilliseconds function 657

NStopwatchGetElapsedTicks function 657

NStopwatchGetFrequency function 657

NStopwatchGetTimestamp function 658

NStopwatchIsHighResolution function 658

NStopwatchIsRunning function 658

NStopwatchReset function 659

NStopwatchRestart function 659

NStopwatchStart function 659

NStopwatchStartNew function 660

NStopwatchStop function 660

NStream class 1844

- about NStream class 1844
- AsStream 1847
- CanRead 1850
- CanSeek 1850
- CanWrite 1850
- CopyTo 1847
- Flush 1847
- FromStream 1847
- Length 1850
- NativeType 1850
- Null 1850
- Position 1851
- Read 1848
- ReadByte 1848
- Seek 1848
- SetLength 1849
- Synchronized 1849
- Write 1849
- WriteByte 1850

NStream Unit 660

NStreamCanRead function 661

NStreamCanSeek function 661

NStreamCanWrite function 662

NStreamCopyTo function 662

NStreamCopyToWithBufferSize function 662

NStreamFlush function 663

NStreamGetLength function 663

NStreamGetNull function 664

NStreamGetPosition function 664

NStreamRead function 664

NStreamReadByte function 665

NStreamReader Unit 668

NStreamReaderCreate function 669

NStreamReaderCreateFromFile function 669

NStreamReaderCreateFromFile macro 672

NStreamReaderCreateFromFileWithEncoding function 669

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStreamReaderCreateFromFileWithEncoding macro 672
NStreamReaderCreateFromFileWithEncodingAndBufferSize function 670
NStreamReaderCreateFromFileWithEncodingAndBufferSize macro 673
NStreamReaderCreateWithEncoding function 670
NStreamReaderCreateWithEncodingAndBufferSize function 670
NStreamReaderGetBaseStream function 671
NStreamReaderGetCurrentEncoding function 671
NStreamReaderGetNull function 672
NStreamReaderIsEndOfStream function 672
NStreamReadN function 665
NStreamSeek function 665
NStreamSetLength function 666
NStreamSetPosition function 666
NStreamSynchronized function 666
NStreamWrite function 667
NStreamWriteByte function 667
NStreamWriteN function 667
NStreamWriter Unit 673
NStreamWriterCreate function 674
NStreamWriterCreateFromFile function 674
NStreamWriterCreateFromFile macro 677
NStreamWriterCreateFromFileWithEncoding function 674
NStreamWriterCreateFromFileWithEncoding macro 678
NStreamWriterCreateFromFileWithEncodingAndBufferSize function 675
NStreamWriterCreateFromFileWithEncodingAndBufferSize macro 678
NStreamWriterCreateWithEncoding function 675
NStreamWriterCreateWithEncodingAndBufferSize function 676
NStreamWriterGetAutoFlush function 676
NStreamWriterGetBaseStream function 676
NStreamWriterGetNull function 677
NStreamWriterSetAutoFlush function 677
NString Unit 678
NString.h 762
NStringAppend macro 729
NStringAppendChars macro 729
NStringAppendStrOrChars function 687
NStringAppendStrOrChars macro 730
NStringBuilder Unit 769
NStringBuilderAppend function 776
NStringBuilderAppend macro 804
NStringBuilderAppendBoolean function 776
NStringBuilderAppendBoolean macro 805
NStringBuilderAppendByte function 776
NStringBuilderAppendByte macro 805
NStringBuilderAppendChar function 777
NStringBuilderAppendChar macro 805
NStringBuilderAppendChars function 777
NStringBuilderAppendChars macro 805
NStringBuilderAppendDouble function 777
NStringBuilderAppendDouble macro 805
NStringBuilderAppendEmptyLine function 778
NStringBuilderAppendEmptyLine macro 805
NStringBuilderAppendFormat function 778
NStringBuilderAppendFormat macro 806
NStringBuilderAppendFormatN function 778
NStringBuilderAppendFormatN macro 806
NStringBuilderAppendFormatVA function 779
NStringBuilderAppendFormatVA macro 806
NStringBuilderAppendFormatVAN function 779
NStringBuilderAppendFormatVAN macro 806
NStringBuilderAppendInt16 function 780
NStringBuilderAppendInt16 macro 806
NStringBuilderAppendInt32 function 780
NStringBuilderAppendInt32 macro 807
NStringBuilderAppendInt64 function 780
NStringBuilderAppendInt64 macro 807
NStringBuilderAppendLine function 781
NStringBuilderAppendLine macro 807
NStringBuilderAppendLineN function 781
NStringBuilderAppendLineN macro 807
NStringBuilderAppendN function 781
NStringBuilderAppendN macro 807
NStringBuilderAppendObject function 782
NStringBuilderAppendObject macro 807
NStringBuilderAppendPointer function 782
NStringBuilderAppendPointer macro 808
NStringBuilderAppendResult function 782
NStringBuilderAppendResult macro 808

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringBuilderAppendSByte function 783
NStringBuilderAppendSByte macro 808
NStringBuilderAppendSingle function 783
NStringBuilderAppendSingle macro 808
NStringBuilderAppendSizeType function 783
NStringBuilderAppendSizeType macro 808
NStringBuilderAppendSSizeType function 784
NStringBuilderAppendSSizeType macro 808
NStringBuilderAppendUInt16 function 784
NStringBuilderAppendUInt16 macro 809
NStringBuilderAppendUInt32 function 784
NStringBuilderAppendUInt32 macro 809
NStringBuilderAppendUInt64 function 785
NStringBuilderAppendUInt64 macro 809
NStringBuilderAppendValue function 785
NStringBuilderAppendValue macro 809
NStringBuilderAppendValueP function 785
NStringBuilderAppendValueP macro 809
NStringBuilderClear function 786
NStringBuilderClear macro 810
NStringBuilderCopyTo macro 810
NStringBuilderCopyToString macro 810
NStringBuilderCopyToStrOrChars function 786
NStringBuilderCopyToStrOrChars macro 811
NStringBuilderDetachCharArray macro 811
NStringBuilderDetachString macro 811
NStringBuilderDetachStringN function 786
NStringBuilderDetachStrOrChars function 787
NStringBuilderDetachStrOrChars macro 811
NStringBuilderDispose function 787
NStringBuilderDispose macro 812
NStringBuilderGetBuffer macro 812
NStringBuilderGetCapacity function 787
NStringBuilderGetCapacity macro 812
NStringBuilderGetChar function 788
NStringBuilderGetChar macro 812
NStringBuilderGetCharsBuffer macro 813
NStringBuilderGetLength function 788
NStringBuilderGetLength macro 813
NStringBuilderGetStringWrapper function 788
NStringBuilderGetStringWrapper macro 813
NStringBuilderGetStrOrCharsBuffer function 789
NStringBuilderGetStrOrCharsBuffer macro 813
NStringBuilderInit function 789
NStringBuilderInit macro 813
NStringBuilderInitEx function 790
NStringBuilderInitEx macro 814
NStringBuilderInitWithCapacity function 790
NStringBuilderInitWithCapacity macro 814
NStringBuilderInitWithChars macro 814
NStringBuilderInitWithString macro 814
NStringBuilderInitWithStringN function 790
NStringBuilderInitWithStringN macro 815
NStringBuilderInitWithStrOrChars function 791
NStringBuilderInitWithStrOrChars macro 815
NStringBuilderInsert function 791
NStringBuilderInsert macro 815
NStringBuilderInsertBoolean function 791
NStringBuilderInsertBoolean macro 815
NStringBuilderInsertByte function 792
NStringBuilderInsertByte macro 815
NStringBuilderInsertChar function 792
NStringBuilderInsertChar macro 816
NStringBuilderInsertChars function 792
NStringBuilderInsertChars macro 816
NStringBuilderInsertDouble function 793
NStringBuilderInsertDouble macro 816
NStringBuilderInsertInt16 function 793
NStringBuilderInsertInt16 macro 816
NStringBuilderInsertInt32 function 794
NStringBuilderInsertInt32 macro 816
NStringBuilderInsertInt64 function 794
NStringBuilderInsertInt64 macro 816
NStringBuilderInsertN function 794
NStringBuilderInsertN macro 817
NStringBuilderInsertObject function 795
NStringBuilderInsertObject macro 817
NStringBuilderInsertPointer function 795
NStringBuilderInsertPointer macro 817
NStringBuilderInsertResult function 796
NStringBuilderInsertResult macro 817
NStringBuilderInsertSByte function 796

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringBuilderInsertSByte macro 817
NStringBuilderInsertSingle function 796
NStringBuilderInsertSingle macro 817
NStringBuilderInsertSizeType function 797
NStringBuilderInsertSizeType macro 818
NStringBuilderInsertSSizeType function 797
NStringBuilderInsertSSizeType macro 818
NStringBuilderInsertUInt16 function 797
NStringBuilderInsertUInt16 macro 818
NStringBuilderInsertUInt32 function 798
NStringBuilderInsertUInt32 macro 818
NStringBuilderInsertUInt64 function 798
NStringBuilderInsertUInt64 macro 818
NStringBuilderInsertValue function 799
NStringBuilderInsertValueP function 799
NStringBuilderRemove function 799
NStringBuilderRemove macro 819
NStringBuilderReplace function 800
NStringBuilderReplace macro 819
NStringBuilderReplaceChars macro 819
NStringBuilderReplaceCharsInRange macro 819
NStringBuilderReplaceInRange function 800
NStringBuilderReplaceInRange macro 820
NStringBuilderReplaceString macro 820
NStringBuilderReplaceStringInRange macro 820
NStringBuilderReplaceStringInRangeN function 801
NStringBuilderReplaceStringN function 801
NStringBuilderReplaceStringN macro 821
NStringBuilderReplaceStrOrChars function 801
NStringBuilderReplaceStrOrChars macro 821
NStringBuilderReplaceStrOrCharsInRange function 802
NStringBuilderReplaceStrOrCharsInRange macro 821
NStringBuilderSetCapacity function 802
NStringBuilderSetCapacity macro 821
NStringBuilderSetChar function 803
NStringBuilderSetChar macro 822
NStringBuilderSetLength function 803
NStringBuilderSetLength macro 822
NStringBuilderToString function 803
NStringBuilderToString macro 822
NStringBuilderToStringRange function 804
NStringBuilderToStringRange macro 822
NStringClone function 687
NStringCompare macro 730
NStringCompareChars macro 730
NStringCompareN macro 731
NStringCompareRange macro 731
NStringCompareRangeChars macro 732
NStringCompareRangeN macro 732
NStringCompareRangeNA function 688
NStringCompareRangeNW function 688
NStringCompareRangeStrOrChars macro 732
NStringCompareRangeStrOrCharsA function 689
NStringCompareRangeStrOrCharsW function 689
NStringCompareStrOrChars macro 732
NStringConcat macro 733
NStringConcatArray function 690
NStringConcatArray macro 733
NStringConcatChars macro 733
NStringConcatMany function 690
NStringConcatMany macro 734
NStringConcatManyVA function 691
NStringConcatManyVA macro 734
NStringConcatN function 691
NStringConcatStrOrChars function 692
NStringConcatStrOrChars macro 734
NStringContainsChars macro 734
NStringContainsStr macro 735
NStringContainsStrN function 692
NStringContainsStrOrChars function 692
NStringContainsStrOrChars macro 735
NStringCopyTo macro 735
NStringCopyToString macro 736
NStringCopyToStrOrChars function 693
NStringCopyToStrOrChars macro 736
NStringCreate macro 736
NStringCreateFromChar function 693
NStringCreateFromChar macro 737
NStringCreateFromChars macro 737
NStringCreateFromStrOrChars function 694
NStringCreateFromStrOrChars macro 737
NStringCreateWrapper function 694

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringCreateWrapper macro 738
NStringEndsWith macro 738
NStringEndsWithChars macro 738
NStringEndsWithN function 694
NStringEndsWithStrOrChars macro 739
NStringEndsWithStrOrCharsA function 695
NStringEndsWithStrOrCharsW function 695
NStringEquals macro 739
NStringEqualsChars macro 739
NStringEqualsN function 696
NStringEqualsStrOrChars macro 740
NStringEqualsStrOrCharsA function 696
NStringEqualsStrOrCharsW function 696
NStringFormat function 697
NStringFormat macro 740
NStringFormatVA function 697
NStringFormatVA macro 740
NStringFree function 698
NStringFreeArray function 698
NStringFreeElements function 698
NStringGetBuffer function 698
NStringGetBuffer macro 740
NStringGetChar macro 740
NStringGetEmpty function 699
NStringGetHashCode function 699
NStringGetLength function 699
NStringGetLength macro 740
NStringHasEmbeddedNull function 700
NStringIndexOf function 700
NStringIndexOf macro 741
NStringIndexOfAny function 701
NStringIndexOfAny macro 741
NStringIndexOfAnyFrom function 701
NStringIndexOfAnyFrom macro 741
NStringIndexOfAnyInRange function 701
NStringIndexOfAnyInRange macro 741
NStringIndexOfChars macro 741
NStringIndexOfCharsFrom macro 742
NStringIndexOfCharsInRange macro 742
NStringIndexOfFrom function 702
NStringIndexOfFrom macro 743
NStringIndexOfInRange function 702
NStringIndexOfInRange macro 743
NStringIndexOfStr macro 743
NStringIndexOfStrFrom macro 744
NStringIndexOfStrFromN function 703
NStringIndexOfStrFromN macro 744
NStringIndexOfStrInRange macro 744
NStringIndexOfStrInRangeN function 703
NStringIndexOfStrInRangeN macro 745
NStringIndexOfStrN function 704
NStringIndexOfStrN macro 745
NStringIndexOfStrOrChars function 704
NStringIndexOfStrOrChars macro 745
NStringIndexOfStrOrCharsFrom function 705
NStringIndexOfStrOrCharsFrom macro 745
NStringIndexOfStrOrCharsInRange function 705
NStringIndexOfStrOrCharsInRange macro 745
NStringInsert macro 745
NStringInsertChars macro 746
NStringInsertN function 706
NStringInsertN macro 746
NStringInsertStrOrChars function 706
NStringInsertStrOrChars macro 746
NStringIsA function 706
NStringIsEmpty function 707
NStringIsW function 707
NStringIsWhiteSpace function 707
NStringJoinArray function 708
NStringJoinArray macro 747
NStringJoinArrayVN function 708
NStringJoinArrayVN macro 747
NStringJoinMany function 709
NStringJoinMany macro 747
NStringJoinManyVA function 709
NStringJoinManyVAVN function 709
NStringJoinManyVAVN macro 747
NStringJoinManyVN function 710
NStringJoinManyVN macro 747
NStringLastIndexOf function 710
NStringLastIndexOf macro 748
NStringLastIndexOfAny function 710

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringLastIndexOfAny macro 748
NStringLastIndexOfAnyFrom function 711
NStringLastIndexOfAnyFrom macro 748
NStringLastIndexOfAnyInRange function 711
NStringLastIndexOfAnyInRange macro 748
NStringLastIndexOfChars macro 748
NStringLastIndexOfCharsFrom macro 749
NStringLastIndexOfCharsInRange macro 749
NStringLastIndexOfFrom function 712
NStringLastIndexOfFrom macro 750
NStringLastIndexOfInRange function 712
NStringLastIndexOfInRange macro 750
NStringLastIndexOfStr macro 750
NStringLastIndexOfStrFrom macro 750
NStringLastIndexOfStrFromN function 713
NStringLastIndexOfStrFromN macro 751
NStringLastIndexOfStrInRange macro 751
NStringLastIndexOfStrInRangeN function 713
NStringLastIndexOfStrInRangeN macro 752
NStringLastIndexOfStrN function 714
NStringLastIndexOfStrN macro 752
NStringLastIndexOfStrOrChars function 714
NStringLastIndexOfStrOrChars macro 752
NStringLastIndexOfStrOrCharsFrom function 715
NStringLastIndexOfStrOrCharsFrom macro 752
NStringLastIndexOfStrOrCharsInRange function 715
NStringLastIndexOfStrOrCharsInRange macro 752
NStringPadLeft function 716
NStringPadLeft macro 752
NStringPadLeftWith function 716
NStringPadLeftWith macro 753
NStringPadRight function 717
NStringPadRight macro 753
NStringPadRightWith function 717
NStringPadRightWith macro 753
NStringPrepend macro 753
NStringPrependChars macro 754
NStringPrependStrOrChars function 718
NStringPrependStrOrChars macro 754
NStringPromoteBuffer function 718
NStringPromoteBuffer macro 754
NStringReader Unit 822
NStringReaderCreate macro 823
NStringReaderCreateFromChars macro 824
NStringReaderCreateFromStrOrChars function 823
NStringReaderCreateFromStrOrChars macro 824
NStringReaderCreateN function 823
NStringRemove function 718
NStringRemove macro 754
NStringRemoveFrom function 719
NStringRemoveFrom macro 754
NStringReplace function 719
NStringReplace macro 755
NStringReplaceChars macro 755
NStringReplaceStr macro 755
NStringReplaceStrOrChars function 720
NStringReplaceStrOrChars macro 756
NStringSet function 720
NStringSplit macro 756
NStringSplitWithCount function 720
NStringSplitWithCount macro 756
NStringSplitWithStrs macro 757
NStringSplitWithStrsAndCount function 721
NStringSplitWithStrsAndCount macro 757
NStringSplitWithStrsAndCountN function 721
NStringSplitWithStrsN macro 757
NStringStartsWith macro 758
NStringStartsWithChars macro 758
NStringStartsWithN function 722
NStringStartsWithStrOrChars macro 759
NStringStartsWithStrOrCharsA function 722
NStringStartsWithStrOrCharsW function 723
NStringSubstring function 723
NStringSubstring macro 759
NStringSubstringFrom function 724
NStringSubstringFrom macro 759
NStringToCharArray macro 759
NStringToCharArrayRange macro 759
NStringToLower function 724
NStringToNewString macro 760
NStringToNewStringRange macro 760
NStringToString function 724

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NStringToString macro 761
NStringToStringOld function 725
NStringToStringOld macro 761
NStringToStrOrCharArray function 725
NStringToStrOrCharArray macro 761
NStringToStrOrCharArrayRange function 726
NStringToStrOrCharArrayRange macro 761
NStringToUpper function 726
NStringTrim function 726
NStringTrimAny function 727
NStringTrimAny macro 761
NStringTrimEnd function 727
NStringTrimEndAny function 727
NStringTrimEndAny macro 762
NStringTrimStart function 728
NStringTrimStartAny function 728
NStringTrimStartAny macro 762
NStringWriter Unit 824
NStringWriter.h 826
NStringWriterCreate function 825
NStringWriterCreateEx function 825
NStringWriterCreateWithCapacity function 825
NStringWriterDetachString function 826
NSucceeded macro 434
NSurveillance class 1885
about NSurveillance class 1885
AddEvent 1895
AddTemplate 1895, 1896
Any 1901
DetectionDetectMoreFaces 1897
DetectionFaceConfidenceThreshold 1897
DetectionFaceDetectionScaleCount 1897
DetectionFaceRollAngleBase 1897
DetectionMaxIod 1897
DetectionMaxRollAngleDeviation 1897
DetectionMaxYawAngleDeviation 1898
DetectionMinIod 1898
DeviceManager 1898
DllName 1891
ExtractionBlinkConfidenceThreshold 1898
ExtractionDarkGlassesConfidenceThreshold 1898
ExtractionDetectBlink 1898
ExtractionDetectDarkGlasses 1899
ExtractionDetectExpression 1899
ExtractionDetectGender 1899
ExtractionDetectGlasses 1899
ExtractionDetectMouthOpen 1899
ExtractionEnrollmentTemplateSize 1899
ExtractionExpressionConfidenceThreshold 1900
ExtractionFaceQualityThreshold 1900
ExtractionGenderConfidenceThreshold 1900
ExtractionGlassesConfidenceThreshold 1900
ExtractionIdentificationTemplateSize 1900
ExtractionMaxRecordsPerTemplate 1900
ExtractionMouthOpenConfidenceThreshold 1901
FacesDetected 1901
GetTemplateIds 1896
IdentificationMatchingThreshold 1901
IdentificationMaxBestMatchCount 1901
ImageCaptured 1902
IsRunning 1901
IsRunningChanged 1902
MaxThreadCount 1901
NativeType 1901
NSurveillance 1890
ParameterDetectionDetectMoreFaces 1891
ParameterDetectionFaceConfidenceThreshold 1891
ParameterDetectionFaceDetectionScaleCount 1891
ParameterDetectionFaceRollAngleBase 1891
ParameterDetectionMaxIod 1891
ParameterDetectionMaxRollAngleDeviation 1891
ParameterDetectionMaxYawAngleDeviation 1892
ParameterDetectionMinIod 1892
ParameterExtractionBlinkConfidenceThreshold 1892
ParameterExtractionDarkGlassesConfidenceThreshold 1892
ParameterExtractionDetectBlink 1892
ParameterExtractionDetectDarkGlasses 1892
ParameterExtractionDetectExpression 1892
ParameterExtractionDetectGender 1892
ParameterExtractionDetectGlasses 1893
ParameterExtractionDetectMouthOpen 1893
ParameterExtractionEnrollmentTemplateSize 1893

Copyright © 2009-2014 Neurotechnology. All rights reserved.

ParameterExtractionExpressionConfidenceThreshold 1893
ParameterExtractionFaceQualityThreshold 1893
ParameterExtractionGenderConfidenceThreshold 1893
ParameterExtractionGlassesConfidenceThreshold 1894
ParameterExtractionIdentificationTemplateSize 1894
ParameterExtractionMaxRecordsPerTemplate 1894
ParameterExtractionMouthOpenConfidenceThreshold 1894
ParameterIdentificationMatchingThreshold 1894
ParameterIdentificationMaxBestMatchCount 1895
ParameterMaxThreadCount 1895
RemoveAllTemplates 1896
RemoveEvent 1896
RemoveTemplate 1896
Start 1896
Stop 1897
SubjectAppeared 1902
SubjectDisappeared 1902
SubjectTrack 1902
NSurveillance Library 1439, 1915
NSurveillance Module 1440
NSurveillance.h 1465
NSurveillanceAddEventCallbackEx function 1447
NSurveillanceAddIsRunningChangedCallback function 1448
NSurveillanceAddSourceEx function 1448
NSurveillanceAddSourceEx2 function 1449
NSurveillanceAddTemplateEx function 1449
NSurveillanceClearSourcesEx function 1450
NSurveillanceCreate function 1451
NSurveillanceEventArgs class 1902
about NSurveillanceEventArgs class 1902
EventDetails 1902
EventType 1903
NSurveillanceEventDetails class 1903
about NSurveillanceEventDetails class 1903
BestMatches 1906
DetectionDetails 1906
ExtractionStatus 1907
FrameIndex 1907
GetFaces 1906
GetSubjectTemplate 1906
GrayscaleImage 1907
NativeType 1907
OriginalImage 1907
Owner 1907
SourceIndex 1907
Template 1907
TimeStamp 1908
TraceIndex 1908
NSurveillanceEventDetails.BestMatchCollection class 1906
about NSurveillanceEventDetails.BestMatchCollection class 1906
NSurveillanceEventDetails.h 1464
NSurveillanceEventType enumeration 1457
NSurveillanceEventType_ enumeration 1457
NSurveillanceGetDeviceManager function 1451
NSurveillanceGetInfo function 1450
NSurveillanceGetSourceCountEx function 1451
NSurveillanceGetSourceEx function 1452
NSurveillanceGetSourceEx2 function 1452
NSurveillanceGetTemplateIdsEx function 1452
NSurveillanceIsRunning function 1453
NSurveillanceParams.h 1466
NSurveillanceRemoveAllTemplatesEx function 1453
NSurveillanceRemoveEventCallbackEx function 1454
NSurveillanceRemoveIsRunningChangedCallback function 1454
NSurveillanceRemoveSourceEx function 1455
NSurveillanceRemoveTemplateEx function 1455
NSurveillanceStartEx function 1455
NSurveillanceStopEx function 1456
NSyncEvent Unit 827
NSyncEventCreate function 827
NSyncEventCreateFromOSHandle function 828
NSyncEventGetOSHandle function 828
NSyncEventReset macro 829
NSyncEventSet function 828
ntcAChar enumeration member 896
ntcArray enumeration member 896
ntcBoolean enumeration member 896
ntcBuffer enumeration member 896
ntcByte enumeration member 896
ntcChar enumeration member 896

Copyright © 2009-2014 Neurotechnology. All rights reserved.

ntcComplex enumeration member 896
ntcDateTime enumeration member 896
ntcDouble enumeration member 896
ntcGuid enumeration member 896
ntcInt16 enumeration member 896
ntcInt32 enumeration member 896
ntcInt64 enumeration member 896
ntcNone enumeration member 896
ntcObject enumeration member 896
ntcOther enumeration member 896
ntcPointer enumeration member 896
ntcRational enumeration member 896
ntcResult enumeration member 896
ntcSByte enumeration member 896
ntcSingle enumeration member 896
ntcSizeType enumeration member 896
ntcSSizeType enumeration member 896
ntcString enumeration member 896
ntcTimeSpan enumeration member 896
ntcUInt16 enumeration member 896
ntcUInt32 enumeration member 896
ntcUInt64 enumeration member 896
ntcURational enumeration member 896
ntcValue enumeration member 896
ntcWChar enumeration member 896
NTemplate class 1762
about NTemplate class 1762
CalculateSize 1766
Check 1766, 1767
Clear 1767
Faces 1768
Fingers 1768
GetSize 1767
Irises 1768
NativeType 1768
NTemplate 1765, 1766
Palms 1768
Voices 1768
NTemplate Unit 219
NTemplate.h 254
NTemplateAddFaces macro 245
NTemplateAddFacesCopy macro 246
NTemplateAddFacesCopyEx function 221
NTemplateAddFacesEx function 221
NTemplateAddFacesFromMemory macro 246
NTemplateAddFacesFromMemoryEx function 222
NTemplateAddFingers macro 247
NTemplateAddFingersCopy macro 247
NTemplateAddFingersCopyEx function 222
NTemplateAddFingersEx function 223
NTemplateAddFingersFromMemory macro 248
NTemplateAddFingersFromMemoryEx function 223
NTemplateAddIrises macro 249
NTemplateAddIrisesCopy macro 249
NTemplateAddIrisesCopyEx function 224
NTemplateAddIrisesEx function 225
NTemplateAddIrisesFromMemory macro 249
NTemplateAddIrisesFromMemoryEx function 226
NTemplateAddPalms macro 250
NTemplateAddPalmsCopy macro 251
NTemplateAddPalmsCopyEx function 225
NTemplateAddPalmsEx function 226
NTemplateAddPalmsFromMemory macro 251
NTemplateAddPalmsFromMemoryEx function 227
NTemplateAddVoices function 228
NTemplateAddVoicesCopy macro 252
NTemplateAddVoicesCopyEx function 227
NTemplateAddVoicesFromMemory function 229
NTemplateCalculateSize function 230
NTemplateCheck function 229
NTemplateClear function 229
NTemplateClone macro 252
NTemplateCreate macro 252
NTemplateCreateEx function 232
NTemplateCreateFromMemory function 232
NTemplateGetFaces function 234
NTemplateGetFacesEx function 231
NTemplateGetFingers function 234
NTemplateGetFingersEx function 233
NTemplateGetIrises function 235
NTemplateGetIrisesEx function 235
NTemplateGetPalms function 236

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTemplateGetPalmsEx function 236
NTemplateGetSize macro 253
NTemplateGetSizeMem function 237
NTemplateGetVoices function 237
NTemplateGetVoicesEx function 239
NTemplatePack function 238
NTemplateRemoveFaces function 240
NTemplateRemoveFingers function 240
NTemplateRemoveIrises function 241
NTemplateRemovePalms function 241
NTemplateRemoveVoices function 242
NTemplates Unit 256
NTemplates.h 256
NTemplateSaveToMemory macro 253
NTemplateSetFaces function 242
NTemplateSetFingers function 243
NTemplateSetIrises function 243
NTemplateSetPalms function 243
NTemplateSetVoices function 244
NTemplateUnpack function 244
NTextReader Unit 829
NTextReaderGetNull function 830
NTextReaderPeekChar function 830
NTextReaderPeekChar macro 832
NTextReaderReadChar function 831
NTextReaderReadChar macro 833
NTextReaderReadChars function 831
NTextReaderReadChars macro 833
NTextReaderReadLine function 831
NTextReaderReadToEnd function 832
NTextReaderSynchronized function 832
NTextWriter Unit 833
NTextWriterFlush function 835
NTextWriterGetEncoding function 835
NTextWriterGetNewLine function 835
NTextWriterGetNull function 836
NTextWriterSetNewLine function 836
NTextWriterSetNewLine macro 849
NTextWriterSynchronized function 836
NTextWriterWrite function 837
NTextWriterWrite macro 849
NTextWriterWriteBoolean function 837
NTextWriterWriteBooleanLine function 837
NTextWriterWriteChar function 838
NTextWriterWriteChar macro 849
NTextWriterWriteCharLine function 838
NTextWriterWriteCharLine macro 849
NTextWriterWriteChars function 838
NTextWriterWriteChars macro 850
NTextWriterWriteCharsLine function 839
NTextWriterWriteCharsLine macro 850
NTextWriterWriteDouble function 839
NTextWriterWriteDoubleLine function 839
NTextWriterWriteEmptyLine function 840
NTextWriterWriteFormat function 840
NTextWriterWriteFormat macro 850
NTextWriterWriteFormatLine function 840
NTextWriterWriteFormatLine macro 850
NTextWriterWriteFormatLineVA function 841
NTextWriterWriteFormatLineVA macro 850
NTextWriterWriteFormatVA function 841
NTextWriterWriteFormatVA macro 850
NTextWriterWriteInt32 function 842
NTextWriterWriteInt32Line function 842
NTextWriterWriteInt64 function 842
NTextWriterWriteInt64Line function 843
NTextWriterWriteLine function 843
NTextWriterWriteLine macro 851
NTextWriterWriteObject function 843
NTextWriterWriteObjectLine function 844
NTextWriterWritePointer function 844
NTextWriterWritePointerLine function 844
NTextWriterWriteSingle function 845
NTextWriterWriteSingleLine function 845
NTextWriterWriteSizeType function 845
NTextWriterWriteSizeTypeLine function 846
NTextWriterWriteSSizeType function 846
NTextWriterWriteSSizeTypeLine function 846
NTextWriterWriteUInt32 function 847
NTextWriterWriteUInt32Line function 847
NTextWriterWriteUInt64 function 847
NTextWriterWriteUInt64Line function 848

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTextWriterWriteValueLineP function 848
NTextWriterWriteValueP function 848
NThread Unit 851
NThread.h 855
NThreadCreate function 851
NThreadCreateFromOSHandle function 852
NThreadCreateParameterized function 852
NThreadGetCurrent function 853
NThreadGetCurrentId function 853
NThreadGetId function 853
NThreadGetOSHandle function 854
NThreadJoin function 854
NThreadSleep function 854
NTimeSpan Unit 855
NTimeSpan_ type 866
NTimeSpanAdd function 856
NTimeSpanCompare function 857
NTimeSpanCreate function 857
NTimeSpanCreateFromDays function 858
NTimeSpanCreateFromHours function 858
NTimeSpanCreateFromMilliseconds function 858
NTimeSpanCreateFromMinutes function 859
NTimeSpanCreateFromSeconds function 859
NTimeSpanCreateFromTicks function 859
NTimeSpanDecode function 860
NTimeSpanDuration function 860
NTimeSpanGetDays function 860
NTimeSpanGetHours function 861
NTimeSpanGetMilliseconds function 861
NTimeSpanGetMinutes function 861
NTimeSpanGetSeconds function 862
NTimeSpanGetTicks function 862
NTimeSpanGetTotalDays function 862
NTimeSpanGetTotalHours function 863
NTimeSpanGetTotalMilliseconds function 863
NTimeSpanGetTotalMinutes function 863
NTimeSpanGetTotalSeconds function 864
NTimeSpanIsTimeValid function 864
NTimeSpanIsValid function 865
NTimeSpanNegate function 865
NTimeSpanSubtract function 865
NTimeSpanToString function 866
NTimeSpanToString macro 868
NTls Unit 868
NTlsCreate function 868
NTlsCreateForObject function 868
NTlsFree function 869
NTlsGetObject function 869
NTlsGetValue function 869
NTlsSetObject function 870
NTlsSetValue function 870
NTrue macro 1064
NType class 1588
about NType class 1588
BaseType 1597
CopyParameters 1592
CopyPropertyValues 1592
GetParameter 1592
GetPropertyValue 1593
HasOwnerType 1597
IdentifierToString 1593
IsAbstract 1597
IsAssignableFrom 1594
IsBasic 1597
IsCallback 1597
IsCloneable 1598
IsComparable 1598
IsDeprecated 1598
IsDisposable 1598
IsEnum 1598
IsEquatable 1598
IsFlagsEnum 1598
IsHandle 1598
IsInstanceOfType 1594
IsMemorySerializable 1598
IsObject 1599
IsParsable 1599
IsPrimitive 1599
IsPublic 1599
IsSealed 1599
IsSerializable 1599
IsSignNeutral 1599

Copyright © 2009-2014 Neurotechnology. All rights reserved.

IsStatic 1599
IsStruct 1599
IsSubclassOf 1594
Module 1600
Name 1600
NativeType 1600
NTypeCodeNativeType 1600
OwnerType 1600
Reset 1594
ResetPropertyValue 1595
RootType 1600
RootTypeCode 1600
SetParameter 1595
SetPropertyValue 1595, 1596
ToNativeType 1596
ToType 1596, 1597
TypeCode 1600
UseInsteadType 1600
ValueSize 1601
NType Unit 871
NType.h 900
NTypeAreValuesEqual function 873
NTypeCode enumeration 896
NTypeCode_ enumeration 896
NTypeCompareValues function 873
NTypeCopyParameters macro 897
NTypeCopyPropertyValues function 874
NTypeCopyValue function 874
NTypeDescriptor Unit 903
NTypeDescriptor.h 905
NTypeDescriptorGetDefaultPropertyName function 903
NTypeDescriptorGetDefaultPropertyNameForType function 903
NTypeDescriptorGetProperties function 904
NTypeDescriptorGetPropertiesForType function 904
NTypeDisposeValue function 875
NTypeDisposeValues function 875
NTypeFreeValues function 875
NTypeGetBaseType function 876
NTypeGetModule function 876
NTypeGetName macro 898
NTypeGetNameEx function 876
NTypeGetNameEx macro 898
NTypeGetNameN function 877
NTypeGetOwnerType function 877
NTypeGetParameter macro 898
NTypeGetParameterEx function 877
NTypeGetPropertyValueN function 878
NTypeGetPropertyValueP function 878
NTypeGetPropertyValueP macro 898
NTypeGetRootType function 879
NTypeGetRootTypeCode function 879
NTypeGetTypeCode function 879
NTypeGetUseInsteadType function 880
NTypeGetValueHashCode function 880
NTypeGetValueSize function 881
NTypeHasOwnerType function 881
NTypeIdentifierToString function 881
NTypeIdentifierToStringFN function 881
NTypeIdentifierToStringVN function 882
NTypeIsAbstract function 882
NTypeIsAssignableFrom function 882
NTypeIsBasic function 882
NTypeIsCallback function 883
NTypeIsCloneable function 883
NTypeIsComparable function 884
NTypeIsDeprecated function 884
NTypeIsDisposable function 884
NTypeIsEnum function 885
NTypeIsEquatable function 885
NTypeIsFlagsEnum function 885
NTypeIsHandle function 886
NTypeIsInstanceOfType function 886
NTypeIsInstanceOfTypeP function 886
NTypeIsMemorySerializable function 887
NTypeIsObject function 887
NTypeIsParsable function 887
NTypeIsPrimitive function 888
NTypeIsPublic function 888
NTypeIsSealed function 888
NTypeIsSerializable function 889
NTypeIsSignNeutral function 889
NTypeIsStatic function 889

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NTypeIsStruct function 890
NTypeIsSubclassOf function 890
NTypeParseValue function 890
NTypeParseValue macro 899
NTypeParseValueVN function 891
NTypeParseValueVN macro 899
NTypeReset function 891
NTypeResetPropertyValue function 892
NTypeResetPropertyValueP macro 899
NTypes class 1601
about NTypes class 1601
ACharNativeType 1604
BooleanNativeType 1605
CharNativeType 1605
CharSet 1602
CreateCallback 1602, 1603
DateTimeNativeType 1605
DoubleNativeType 1605
FreeObjectPointer 1603
FreeStringArray 1603
FreeStringElements 1603
GetObjectFromPointer 1603
GetPointerForObject 1604
GuidNativeType 1605
Int16NativeType 1605
Int32NativeType 1605
Int64NativeType 1605
Int8NativeType 1605
IntPtrNativeType 1606
IsMemoryTypeValid 1604
IsMemoryTypeValidForFree 1604
NameStringPairNativeType 1606
NativeType 1606
NAttributesNativeType 1606
NMemoryTypeNativeType 1606
NOSFamilyNativeType 1606
OSFamilyCurrent 1606
PointerNativeType 1606
ResultNativeType 1606
SingleNativeType 1607
StringEncoding 1602
StringNativeType 1607
TimeSpanNativeType 1607
UInt16NativeType 1607
UInt32NativeType 1607
UInt64NativeType 1607
UInt8NativeType 1607
UIntPtrNativeType 1607
WCharApi 1602
WCharNativeType 1607
NTypes Unit 905
NTypes.h 1078
NTypeSetParameter macro 899
NTypeSetParameterEx function 892
NTypeSetPropertyValueN function 892
NTypeSetPropertyValueP function 893
NTypeSetPropertyValueP macro 900
NTypeSetValue function 893
NTypesGetPointer function 980
NTypeTryParseValue function 894
NTypeTryParseValue macro 900
NTypeTryParseValueVN function 894
NTypeTryParseValueVN macro 900
NTypeValueToString function 895
NTypeValueToString macro 900
NTypeValueToStringN function 895
NUInt type 1009
NUInt16 type 1009
NUInt16Parse macro 1064
NUInt16ParseChars macro 1065
NUInt16ParseN function 981
NUInt16ParseStrOrChars function 981
NUInt16ParseStrOrChars macro 1065
NUInt16ParseVN function 981
NUInt16ParseVN macro 1065
NUInt16ToChars function 982
NUInt16ToChars macro 1065
NUInt16ToString function 982
NUInt16ToString macro 1066
NUInt16TryParse macro 1066
NUInt16TryParseChars macro 1066
NUInt16TryParseN function 983

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NUInt16TryParseStrOrChars function 983
NUInt16TryParseStrOrChars macro 1067
NUInt16TryParseVN function 984
NUInt16TryParseVN macro 1067
NUInt32 type 1009
NUInt32Parse macro 1067
NUInt32ParseChars macro 1067
NUInt32ParseN function 984
NUInt32ParseStrOrChars function 984
NUInt32ParseStrOrChars macro 1068
NUInt32ParseVN function 985
NUInt32ParseVN macro 1068
NUInt32ToChars function 985
NUInt32ToChars macro 1068
NUInt32ToString function 986
NUInt32ToString macro 1068
NUInt32TryParse macro 1068
NUInt32TryParseChars macro 1069
NUInt32TryParseN function 986
NUInt32TryParseStrOrChars function 987
NUInt32TryParseStrOrChars macro 1069
NUInt32TryParseVN function 987
NUInt32TryParseVN macro 1069
NUInt64 type 1009
NUInt64Parse macro 1070
NUInt64ParseChars macro 1070
NUInt64ParseN function 988
NUInt64ParseStrOrChars function 988
NUInt64ParseStrOrChars macro 1070
NUInt64ParseVN function 988
NUInt64ParseVN macro 1071
NUInt64ToChars function 989
NUInt64ToChars macro 1071
NUInt64ToString function 989
NUInt64ToString macro 1071
NUInt64TryParse macro 1071
NUInt64TryParseChars macro 1071
NUInt64TryParseN function 990
NUInt64TryParseStrOrChars function 990
NUInt64TryParseStrOrChars macro 1072
NUInt64TryParseVN function 991
NUInt64TryParseVN macro 1072
NUInt8 type 1009
NUInt8Parse macro 1072
NUInt8ParseChars macro 1073
NUInt8ParseN function 991
NUInt8ParseStrOrChars function 991
NUInt8ParseStrOrChars macro 1073
NUInt8ParseVN function 992
NUInt8ParseVN macro 1073
NUInt8ToChars function 992
NUInt8ToChars macro 1073
NUInt8ToString function 993
NUInt8ToString macro 1074
NUInt8TryParse macro 1074
NUInt8TryParseChars macro 1074
NUInt8TryParseN function 993
NUInt8TryParseStrOrChars function 994
NUInt8TryParseStrOrChars macro 1075
NUInt8TryParseVN function 994
NUInt8TryParseVN macro 1075
NULL macro 1075
NULong type 1010
NURational structure 1608
about NURational structure 1608
Denominator 1609
Empty 1608
NativeType 1609
Numerator 1609
NURational 1608
ToString 1609
NURational type 1010
NURational_ structure 1002
NURationalToString function 995
NURationalToString macro 1075
NUShort type 1010
NValue class 1609
about NValue class 1609
Attributes 1639
ChangeType 1619
FromObject 1619, 1620
FromPointer 1621

Copyright © 2009-2014 Neurotechnology. All rights reserved.

FromResult 1621
FromString 1622
FromType 1622, 1623
IsWrapper 1639
KeyValuePairNativeType 1639
LongSize 1639
NameValuePairNativeType 1639
NativeSize 1639
NativeType 1639
NValue 1613, 1614, 1615, 1616, 1617, 1618
Ptr 1639
Size 1639
ToBoolean 1623
ToByte 1624
ToChar 1625
ToDouble 1625, 1626
ToInt16 1626
ToInt32 1627
ToInt64 1627, 1628
ToIntPtr 1628, 1629
ToObject 1629, 1630
ToPointer 1630, 1631
ToResult 1631
ToSByte 1632
ToSingle 1633
ToString 1633, 1634
ToType 1634, 1635
ToUInt16 1636
ToUInt32 1636, 1637
ToUInt64 1637, 1638
ToUIntPtr 1638
TypeCode 1640
ValueType 1640
NValue Unit 1094
NValue.h 1121
NValueChangeType function 1098
NValueChangeTypeP function 1099
NValueCreate function 1099
NValueCreateCustom function 1100
NValueCreateCustomN function 1100
NValueCreateFromBoolean function 1101
NValueCreateFromByte function 1101
NValueCreateFromChar function 1101
NValueCreateFromChar macro 1120
NValueCreateFromDouble function 1102
NValueCreateFromInt16 function 1102
NValueCreateFromInt32 function 1102
NValueCreateFromInt64 function 1103
NValueCreateFromObject function 1103
NValueCreateFromObjectP function 1104
NValueCreateFromPointer function 1104
NValueCreateFromResult function 1104
NValueCreateFromSByte function 1105
NValueCreateFromSingle function 1105
NValueCreateFromSizeType function 1105
NValueCreateFromSSizeType function 1106
NValueCreateFromString function 1106
NValueCreateFromString macro 1120
NValueCreateFromUInt16 function 1106
NValueCreateFromUInt32 function 1107
NValueCreateFromUInt64 function 1107
NValueCreateP function 1108
NValueGetAttributes function 1108
NValueGetFree function 1108
NValueGetPtr function 1109
NValueGetSize function 1109
NValueGetTypeCode function 1109
NValueGetValueType function 1110
NValueSet function 1110
NValueSetP function 1110
NValueToBoolean function 1111
NValueToByte function 1111
NValueToChar function 1112
NValueToChar macro 1120
NValueToDouble function 1112
NValueToInt16 function 1112
NValueToInt32 function 1113
NValueToInt64 function 1113
NValueToObject function 1114
NValueToObjectP function 1114
NValueToPointer function 1114
NValueToResult function 1115

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NValueToSByte function 1115
NValueToSingle function 1115
NValueToSizeType function 1116
NValueToSSizeType function 1116
NValueToString function 1117
NValueToUInt16 function 1117
NValueToUInt32 function 1117
NValueToUInt64 function 1118
NValueToValue function 1118
NValueToValueP function 1119
NVersion structure 1640
!= 1643
< 1643
<= 1644
== 1644
> 1644
>= 1644
about NVersion structure 1640
CompareTo 1642
Empty 1641
Equals 1642
GetHashCode 1642
Major 1645
Minor 1645
NativeType 1645
NVersion 1641
ToString 1643
ushort 1645
NVersion type 1010
NVersion_ type 1010
NVersionRange structure 1645
!= 1650
== 1650
about NVersionRange structure 1645
Contains 1647
Empty 1647
Equals 1648
From 1650
GetHashCode 1648
Intersect 1648
IntersectsWith 1649
NativeType 1650
NVersionRange 1646
To 1651
ToString 1649
uint 1650
NVersionRange type 1011
NVersionRange_ type 1011
NVersionRangeContains function 995
NVersionRangeContainsRange function 995
NVersionRangeIntersect function 996
NVersionRangeIntersectsWith function 996
NVersionRangeToString function 996
NVersionRangeToString macro 1075
NVersionToString function 997
NVersionToString macro 1075
NVideoFormat Unit 1395
NVideoFormat.h 1412
NVideoFormatCreate function 1398
NVideoFormatGetFrameRate function 1398
NVideoFormatGetHeight function 1398
NVideoFormatGetInterlaceMode function 1399
NVideoFormatGetPixelAspectRatio function 1399
NVideoFormatGetWidth function 1399
NVideoFormatMediaSubtypeToString function 1400
NVideoFormatMediaSubtypeToString macro 1412
NVideoFormatSetFrameRate function 1400
NVideoFormatSetHeight function 1401
NVideoFormatSetInterlaceMode function 1401
NVideoFormatSetPixelAspectRatio function 1401
NVideoFormatSetWidth function 1402
NVideoInterlaceMode enumeration 1354
NVideoInterlaceMode_ enumeration 1354
NVideoInterlaceModeIsValid function 1353
NVideoReader Unit 1414
NVideoReader.h 1418
NVideoReaderCreateFromFile function 1415
NVideoReaderCreateFromFile macro 1418
NVideoReaderFree macro 1418
NVideoReaderGetFrame function 1415
NVideoReaderGetFrameCount function 1416
NVideoReaderGetFrameHeight function 1416

Copyright © 2009-2014 Neurotechnology. All rights reserved.

NVideoReaderGetFrameRate function 1417
NVideoReaderGetFrameWidth function 1417
NVideoWriter Module 1419
NVideoWriter.h 1422
NVideoWriterCreateFile function 1420
NVideoWriterCreateFile macro 1421
NVideoWriterFree macro 1422
NVideoWriterOptions.h 1423
NVideoWriterOptionsCreateWithGui function 1421
NVideoWriterOptionsFree macro 1422
NVideoWriterWriteFrame function 1420
nvimInterlaced enumeration member 1354
nvimProgressive enumeration member 1354
nvimUnknown enumeration member 1354
NWaitObject Unit 1123
NWaitObjectWaitFor function 1123
NWaitObjectWaitForTimed function 1123
NWCharParse macro 1076
NWCharParseChars macro 1076
NWCharParseN function 997
NWCharParseStrOrChars function 998
NWCharParseStrOrChars macro 1076
NWCharParseVN function 998
NWCharParseVN macro 1077
NWCharToChars function 998
NWCharToChars macro 1077
NWCharToString function 999
NWCharToString macro 1077
NWCharTryParse macro 1077
NWCharTryParseChars macro 1077
NWCharTryParseN function 999
NWCharTryParseStrOrChars function 1000
NWCharTryParseStrOrChars macro 1078
NWCharTryParseVN function 1000
NWCharTryParseVN macro 1078
NWindows Unit 1124

O

Object enumeration member 1652
ObjectMissing enumeration member 1769
ObjectNotFound enumeration member 1769
ObjectsNotRemoved enumeration member 1769
Obsolete scores 45
Ok enumeration member 1769, 1823
Other enumeration member 1652, 1773, 1774
Overview 10

P

Palm enumeration member 1771
PalmGeometry enumeration member 1771
PalmPrint enumeration member 1771
PalmScanner enumeration member 1824
Paused enumeration member 1862
PausePending enumeration member 1862
PlainArch enumeration member 1775
PlainLeftFourFingers enumeration member 1776
PlainLeftThumb enumeration member 1776
PlainRightFourFingers enumeration member 1776
PlainRightThumb enumeration member 1776
PlainThumbs enumeration member 1776
PlainWhorl enumeration member 1775
Plugged enumeration member 1882
PluggingError enumeration member 1882
Png Unit 1423
Png.h 1428
PNG_DEFAULT_COMPRESSION_LEVEL macro 1428
PngInfoGetCompressionLevel function 1424
PngInfoSetCompressionLevel function 1424
PngLoadImageFromFile function 1424
PngLoadImageFromMemory function 1425
PngLoadImageFromStream function 1425
PngSaveImageToFile function 1426
PngSaveImageToMemory function 1426
PngSaveImageToStream function 1427
Pointer enumeration member 1652
Preventing race conditions 23
Prosilica GigE Vision 19
Protected enumeration member 1651
pValue variable 870

Q

QualityCheckExposureFailed enumeration member 1779

Copyright © 2009-2014 Neurotechnology. All rights reserved.

QualityCheckGrayscaleDensityFailed enumeration member 1779
QualityCheckSharpnessFailed enumeration member 1779

R

RadialLoop enumeration member 1775
RaisedBrows enumeration member 1780
Rational enumeration member 1652
Recommendations and Constraints 15
Reference
.NET 1468
API 60
RemoveDevice 25
Required Libraries 10
Result enumeration member 1652
Retina enumeration member 1771
Right enumeration member 1770, 1772
RightBackOfHand enumeration member 1770
RightEyePatch enumeration member 1780
RightFullPalm enumeration member 1776
RightHypothenar enumeration member 1776
RightIndex enumeration member 1776
RightIndexFinger enumeration member 1776
RightIndexLeftIndexFingers enumeration member 1776
RightIndexMiddleFingers enumeration member 1776
RightIndexMiddleRingFingers enumeration member 1776
RightInterdigital enumeration member 1776
RightLittle enumeration member 1776
RightLittleFinger enumeration member 1770, 1776
RightLowerPalm enumeration member 1776
RightMiddle enumeration member 1776
RightMiddleFinger enumeration member 1770, 1776
RightMiddleRingFingers enumeration member 1776
RightMiddleRingLittleFingers enumeration member 1776
RightOther enumeration member 1776
RightPalm enumeration member 1770
RightPointerFinger enumeration member 1770
RightRing enumeration member 1776
RightRingFinger enumeration member 1770, 1776
RightRingLittleFingers enumeration member 1776
RightSlantLoop enumeration member 1775
RightThenar enumeration member 1776
RightThumb enumeration member 1770, 1776
RightUpperPalm enumeration member 1776
RightWrist enumeration member 1770
RightWritersPalm enumeration member 1776
RiseTechnology enumeration member 1652
Running enumeration member 1862

S

Samples 49
SByte enumeration member 1652
Scar enumeration member 1775
Scarf enumeration member 1780
Scent enumeration member 1771
SelectByFaceThenFuse enumeration member 1781
SelectByIrisThenFuse enumeration member 1781
SensorCleaning enumeration member 1823
SetBiometricDeviceSpoofDetection 38
SetCaptureDeviceCurrentFormat 35
SignatureDynamics enumeration member 1771
SignatureOrSign enumeration member 1771
SilentOperation enumeration member 1823
Single Computer License 3
Single enumeration member 1652
SiS enumeration member 1652
SizeType enumeration member 1652
Small enumeration member 1779
Smile enumeration member 1780
SmileOpenedJaw enumeration member 1780
Specified enumeration member 1780
SpoofDetected enumeration member 1769
Squinting enumeration member 1780
SSizeType enumeration member 1652
Standard enumeration member 1651
StartBiometricDeviceSequence 39
StartCaptureDeviceCapturing 32
StartPending enumeration member 1862
StopCaptureDeviceCapturing 32
Stopped enumeration member 1862
StopPending enumeration member 1862
String enumeration member 1652

Copyright © 2009-2014 Neurotechnology. All rights reserved.

StroboCharge enumeration member 1823
SubjectAppeared enumeration member 1908
SubjectDisappeared enumeration member 1908
SubjectTrack enumeration member 1908
Support 1920
Supported Platforms 1
Swipe enumeration member 1773
System Requirements 2

T

TeethVisible enumeration member 1780
TemplateCreated enumeration member 1779
TentedArch enumeration member 1775
ThermalFace enumeration member 1771
ThermalFaceImage enumeration member 1771
ThermalHand enumeration member 1771
ThermalHandImage enumeration member 1771
Tiff Unit 1429
Tiff.h 1430
TiffLoadImageFromFile function 1429
TiffLoadImageFromMemory function 1429
TiffLoadImageFromStream function 1430
Timeout enumeration member 1769
TimeSpan enumeration member 1652
TooClose enumeration member 1769
TooEast enumeration member 1769
TooFar enumeration member 1769
TooFast enumeration member 1769
TooFewObjects enumeration member 1769
TooHard enumeration member 1769
TooLong enumeration member 1769
TooManyObjects enumeration member 1769
TooNarrow enumeration member 1769
TooNorth enumeration member 1769
TooShort enumeration member 1769
TooSkewed enumeration member 1769
TooSlow enumeration member 1769
TooSoft enumeration member 1769
TooSouth enumeration member 1769
TooWest enumeration member 1769
TooWide enumeration member 1769
Transmeta enumeration member 1652
Tutorials 48

U

UInt16 enumeration member 1652
UInt32 enumeration member 1652
UInt64 enumeration member 1652
UlnarLoop enumeration member 1775
Umc enumeration member 1652
Unicode enumeration member 1651
Unknown enumeration member 1652, 1772, 1773, 1774, 1775, 1776, 1778, 1780
UnknownFourFingers enumeration member 1776
UnknownPalm enumeration member 1776
UnknownThreeFingers enumeration member 1776
UnknownTwoFingers enumeration member 1776
Unplugged enumeration member 1882
Unspecified enumeration member 1778, 1780
Unused enumeration member 1882
UpdateDeviceList 27
URational enumeration member 1652
Using 18
Using Devices 18
Using IP Cameras 18

V

Value enumeration member 1652
Vein enumeration member 1771
VeinPattern enumeration member 1771
Via enumeration member 1652
Voice enumeration member 1771
Volume License Manager 3

W

WChar enumeration member 1652
What's new 5
Whorl enumeration member 1775
Win32Global enumeration member 1651
Win32Local enumeration member 1651
WrongDirection enumeration member 1769
Wsq Unit 1431

Copyright © 2009-2014 Neurotechnology. All rights reserved.

Wsq.h 1438

WSQ_DEFAULT_BIT_RATE macro 1437

WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_DEBIAN_AMD64
macro 1437

WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_DEBIAN_I386
macro 1437

WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_INTEL
macro 1438

WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_INTEL64
macro 1438

WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_MACOSX_POWERPC
macro 1438

WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_WIN32_X86
macro 1438

WSQ_IMPLEMENTATION_NEUROTECHNOLOGY_WIN64_X64
macro 1438

WsqInfoGetBitRate function 1432

WsqInfoGetImplementationNumber function 1432

WsqInfoGetNistCom function 1433

WsqInfoHasNistCom function 1433

WsqInfoSetBitRate function 1433

WsqInfoSetHasNistCom function 1434

WsqLoadImageFromFile function 1434

WsqLoadImageFromMemory function 1434

WsqLoadImageFromStream function 1435

WsqSaveImageToFile function 1435

WsqSaveImageToMemory function 1436

WsqSaveImageToStream function 1437

wxWidgets Compilation 54

Copyright © 2009-2014 Neurotechnology. All rights reserved.