

Swelio Library

Belgian Electronic ID card access library

Table of Contents

Symbol Reference	1
Functions	1
ActivateCard Function	9
ActivateCardEx Function	10
AddRemoveMessageFilter Function	10
AllocateBuffer Function	11
AllocateDefaultHWNDAs Function	11
AllocateDefaultHWNDA Function	11
AllocateHWNDA Function	12
AllocateHWNDA Function	12
AllocateLayeredWindowA Function	12
AllocateLayeredWindowW Function	13
AllocateWindowClassA Function	13
AllocateWindowClassW Function	13
AlphaBlendBitmap Function	14
AlphaBlendNative Function	14
BringWindowToFront Function	14
CardDecryptFileA Function	14
CardDecryptFileW Function	15
CardEncryptFileA Function	15
CardEncryptFileW Function	15
CardSignCadesT Function	16
CardSignCMS Function	16
CertSignCadesT Function	17
CertSignCMS Function	17
CheckMD5 Function	17
CheckSHA1 Function	18
CheckSHA256 Function	18
ClearFileAttributesA Function	19
ClearFileAttributesW Function	19
ClearUnusedMemory Function	19
CloneFont Function	20
CopyNativeBitmap Function	20
CreateCardBuffer Function	20
CreateNativeBitmap Function	20
CreateUnicodeFileA Function	21
CreateUnicodeFileW Function	21
CreateWindowsFont Function	21

CurrentIPAddressA Function	21
CurrentIPAddressW Function	22
DeactivateCard Function	22
DeactivateCardEx Function	22
DeallocateBuffer Function	23
DeallocateHWNDAs Function	23
DeallocateHWNDA Function	23
DecryptFileAESA Function	24
DecryptFileAESW Function	24
DeleteCardBuffer Function	25
DeleteToRecycleBinA Function	25
DeleteToRecycleBinW Function	25
DestroyFont Function	26
DestroyImageBuffer Function	26
DirectoryExistsA Function	26
DirectoryExistsW Function	27
DisplayCertificate Function	27
DpiY Function	27
DrawAlphaText Function	28
DrawAlphaTextRect Function	28
DrawLayeredWindow Function	28
DrawNativeBitmap Function	29
DrawTextDirect Function	29
DrawTextDirectEx Function	29
DrawTextGlow Function	29
DrawTextLine Function	30
DrawTextOutline Function	30
DrawTextRect Function	30
EmptyRecycleBin Function	30
EmToPixels Function	31
EncodeCertificate Function	31
EncodePhoto Function	31
EncryptFileAESA Function	32
EncryptFileAESW Function	32
FileCloseA Function	33
FileCloseW Function	33
FileCopyA Function	33
FileCopyW Function	34
FileCreateRewriteA Function	34
FileCreateRewriteW Function	34
FileDeleteA Function	35
FileDeleteW Function	35

FileExistsA Function	36
FileExistsW Function	36
FileExtensionIsA Function	36
FileExtensionIsW Function	37
FileGetSizeA Function	37
FileGetSizeW Function	37
FileIsExeA Function	38
FileIsExeW Function	38
FileIsIconA Function	39
FileIsIconW Function	39
FileIsImageA Function	39
FileIsImageW Function	40
FileIsLink Function	40
FileOrFolderExistsA Function	40
FileOrFolderExistsW Function	41
FileRenameA Function	41
FileRenameW Function	41
FileWriteA Function	42
FileWriteCharA Function	42
FileWriteCharW Function	42
FileWriteNewLineA Function	43
FileWriteNewLineW Function	43
FileWriteW Function	43
fpreset Function	44
FullPathA Function	44
FullPathW Function	44
GenerateAuthenticationSignatureA Function	44
GenerateAuthenticationSignatureExA Function	45
GenerateAuthenticationSignatureExW Function	45
GenerateAuthenticationSignatureW Function	46
GenerateBMPA Function	46
GenerateBMPW Function	47
GenerateNonRepudiationSignatureA Function	47
GenerateNonRepudiationSignatureExA Function	48
GenerateNonRepudiationSignatureExW Function	48
GenerateNonRepudiationSignatureW Function	49
GeneratePNGA Function	49
GeneratePNGW Function	50
GenerateQRCodeA Function	50
GenerateQRCodeExA Function	51
GenerateQRCodeExW Function	51
GenerateQRCodeW Function	51

GetAllFiles Function	52
GetCardBufferA Function	52
GetCardBufferSize Function	52
GetCardBufferW Function	53
GetCardSerialNumber Function	53
GetCardVersion Function	53
GetEncodedCertificateSize Function	54
GetEncodedPhotoSize Function	54
GetFileMD5A Function	54
GetFileMD5W Function	55
GetFilesCountA Function	55
GetFilesCountW Function	56
GetFileSHA1A Function	56
GetFileSHA1W Function	56
GetFileSHA256A Function	57
GetFileSHA256W Function	57
GetHBitmapA Function	58
GetHBitmapW Function	58
GetISOCodeA Function	59
GetISOCodeW Function	59
GetMD5 Function	60
GetPNGA Function	60
GetPNGW Function	61
GetReaderIndexA Function	61
GetReaderIndexW Function	61
GetReaderNameA Function	62
GetReaderNameLenA Function	62
GetReaderNameLenW Function	63
GetReaderNameW Function	63
GetReadersCount Function	63
GetSelectedReaderIndex Function	64
GetSHA1 Function	64
GetSHA256 Function	64
GetStartupA Function	65
GetStartupW Function	65
GetSupportSIS Function	66
GetTextLineSize Function	66
GetTextSize Function	66
GetTextSizeEx Function	66
HibernateWindows Function	67
IsAnimatedGIFA Function	67
IsAnimatedGIFW Function	67

IsCardActivated Function	68
IsCardActivatedEx Function	68
IsCardPresent Function	68
IsCardPresentEx Function	69
IsCardStillInserted Function	69
IsCardStillInsertedEx Function	69
IsCitrixSession Function	70
IsConnectedToInternet Function	70
IsDirectoryA Function	70
IsDirectoryW Function	70
IsEIDCard Function	71
IsEIDCardEx Function	71
IsEngineActive Function	71
IsFemaleA Function	72
IsFemaleW Function	72
IsMaleA Function	72
IsMaleW Function	73
IsMediaCenter Function	73
IsMetroActive Function	73
IsMultiTouchReady Function	74
IsNativeWin64 Function	74
IsRemoteSession Function	74
IsSISCard Function	74
IsSISCardEx Function	75
IsTabletPC Function	75
IsUnicodeFileA Function	75
IsUnicodeFileW Function	76
IsValidFileNameA Function	76
IsValidFileNameW Function	76
IsValidPathNameA Function	77
IsValidPathNameW Function	77
IsWindows10 Function	78
IsWindows7 Function	78
IsWindows8 Function	78
IsWindowsVista Function	78
IsWindowsXP Function	79
IsWindowsXPSP2 Function	79
IsWow64 Function	79
LayeredWndProcA Function	79
LayeredWndProcW Function	79
LoadBitmapJPG Function	80
LoadBitmapPNG Function	80

LoadCertificateA Function	80
LoadCertificateW Function	81
LoadIdentityA Function	81
LoadIdentityW Function	81
LoadPhotoA Function	82
LoadPhotoW Function	82
LoadPNGResource Function	82
MakeCompatibleBitmap Function	83
MakeSoundFromFileA Function	83
MakeSoundFromFileW Function	83
MakeSoundFromResourceA Function	84
MakeSoundFromResourceW Function	84
PointsToPixels Function	84
PortAvailable Function	85
ReadAddressA Function	85
ReadAddressExA Function	85
ReadAddressExW Function	86
ReadAddressW Function	86
ReadAuthenticationCertificate Function	86
ReadAuthenticationCertificateEx Function	87
ReadBufferFromFileA Function	87
ReadBufferFromFileW Function	87
ReadCaCertificate Function	88
ReadCaCertificateEx Function	88
ReadIdentityA Function	89
ReadIdentityExA Function	89
ReadIdentityExW Function	89
ReadIdentityW Function	90
ReadNonRepudiationCertificate Function	90
ReadNonRepudiationCertificateEx Function	91
ReadPhoto Function	91
ReadPhotoAsBitmap Function	91
ReadPhotoAsBitmapEx Function	92
ReadPhotoEx Function	92
ReadRootCaCertificate Function	93
ReadRootCaCertificateEx Function	93
ReadRrnCertificate Function	93
ReadRrnCertificateEx Function	94
ReadSISCardA Function	94
ReadSISCardExA Function	95
ReadSISCardExW Function	95
ReadSISCardW Function	95

RecycleBinEmpty Function	96
ReloadReadersList Function	96
RemoveCallback Function	96
RemoveStartupA Function	97
RemoveStartupW Function	97
RestoreWindowSubclassA Function	97
RestoreWindowSubclassW Function	98
SaveAuthenticationCertificateA Function	98
SaveAuthenticationCertificateExW Function	98
SaveAuthenticationCertificateW Function	99
SaveCaCertificateA Function	99
SaveCaCertificateExW Function	99
SaveCaCertificateW Function	100
SaveCardToToXMLStreamExA Function	100
SaveCardToToXMLStreamExW Function	100
SaveCardToXmlA Function	101
SaveCardToXmlExA Function	101
SaveCardToXmlExW Function	102
SaveCardToXmlW Function	102
SaveIdentityA Function	103
SaveIdentityW Function	103
SaveNonRepudiationCertificateA Function	103
SaveNonRepudiationCertificateExW Function	104
SaveNonRepudiationCertificateW Function	104
SavePersonCsvToStreamA Function	104
SavePersonCsvToStreamW Function	105
SavePersonToCsvA Function	105
SavePersonToCsvExA Function	105
SavePersonToCsvExW Function	106
SavePersonToCsvW Function	106
SavePhotoA Function	106
SavePhotoAsBitmapA Function	107
SavePhotoAsBitmapExA Function	107
SavePhotoAsBitmapExW Function	108
SavePhotoAsBitmapW Function	108
SavePhotoAsJpegA Function	108
SavePhotoAsJpegExA Function	109
SavePhotoAsJpegExW Function	109
SavePhotoAsJpegW Function	109
SavePhotoW Function	110
SaveRootCaCertificateA Function	110
SaveRootCaCertificateExW Function	111

SaveRootCaCertificateW Function	111
SaveRrnCertificateA Function	111
SaveRrnCertificateExW Function	112
SaveRrnCertificateW Function	112
SelectReader Function	112
SelectReaderByNameA Function	113
SelectReaderByNameW Function	113
SendAPDU Function	114
SetCallback Function	114
SetMWCompatibility Function	114
SetStartupA Function	114
SetStartupW Function	115
SetSupportSIS Function	115
ShellCopyFileA Function	115
ShellCopyFileW Function	116
ShutdownWindows Function	116
StartEngine Function	117
StopEngine Function	117
StretchNativeBitmap Function	117
StripFileNameA Function	118
StripFileNameW Function	118
SuspendWindows Function	118
TurnMonitorOff Function	119
TurnMonitorOn Function	119
UpdateWindowPosition Function	119
VerifyPinA Function	119
VerifyPinExA Function	120
VerifyPinExW Function	120
VerifyPinW Function	120
VerifySignature Function	121
WriteBufferToFileA Function	121
WriteBufferToFileW Function	122
Structs, Records, Enums	122
tagCardEventType Enumeration	123
tagEidAddressA Structure	123
tagEidAddressW Structure	124
tagEidCertificate Structure	124
tagEidIdentityA Structure	125
tagEidIdentityW Structure	126
tagEidPicture Structure	127
tagSISRecordA Structure	127
tagSISRecordW Structure	128

CardEventType Enumeration	128
EidAddressA Structure	129
EidAddressW Structure	129
EidCertificate Structure	130
EidIdentityA Structure	130
EidIdentityW Structure	131
EidPicture Structure	132
PEidAddressA Structure	132
PEidAddressW Structure	133
PEidCertificate Structure	133
PEidIdentityA Structure	133
PEidIdentityW Structure	134
PeidPicture Structure	135
PSISRecordA Structure	136
PSISRecordW Structure	136
SISRecordA Structure	137
SISRecordW Structure	138
Files	150
CardEvents.h	150
CardStructures.h	150
Encryption.h	152
FileOperations.h	153
Graphics.h	155
NationalityConverter.h	156
quicol.h	156
Swelio.h	156
System.h	161
SystemInfo.h	162

Index

a

1 Symbol Reference

1.1 Functions

The following table lists functions in this documentation.

Functions

	Name	Description
≡	ActivateCard (see page 9)	Established communication between the card and the reader
≡	ActivateCardEx (see page 10)	Established communication between the card and the reader
≡	AddRemoveMessageFilter (see page 10)	Adds or removes a message from the User Interface Privilege Isolation (UIPI) message filter.
≡	AllocateBuffer (see page 11)	Allocates the buffer in memory
≡	AllocateDefaultHWNDA (see page 11)	This function creates the invisible tool window
≡	AllocateDefaultHWNDDW (see page 11)	This function creates the invisible tool window
≡	AllocateHWNDA (see page 12)	This function creates the invisible tool window using the provided window procedure
≡	AllocateHWNDDW (see page 12)	This function creates the invisible tool window using the provided window procedure
≡	AllocateLayeredWindowA (see page 12)	This function creates the layered window using the provided window class name
≡	AllocateLayeredWindowW (see page 13)	This function creates the layered window using the provided window class name
≡	AllocateWindowClassA (see page 13)	This function creates the standard window using the provided window class name
≡	AllocateWindowClassW (see page 13)	This function creates the standard window using the provided window class name
≡	AlphaBlendBitmap (see page 14)	This is function AlphaBlendBitmap.
≡	AlphaBlendNative (see page 14)	This is function AlphaBlendNative.
≡	BringWindowToFront (see page 14)	This function brings the specified window to the top of the z-order.
≡	CardDecryptFileA (see page 14)	Decrypt file using Belgian Id card
≡	CardDecryptFileW (see page 15)	Decrypt file using Belgian Id card
≡	CardEncryptFileA (see page 15)	Encrypt file using Belgian Id card
≡	CardEncryptFileW (see page 15)	Encrypt file using Belgian Id card
≡	CardSignCadesT (see page 16)	Sign data with eID card according to CADES-T standard
≡	CardSignCMS (see page 16)	Sign data with eID card according to CMS standard
≡	CertSignCadesT (see page 17)	This is function CertSignCadesT.
≡	CertSignCMS (see page 17)	This is function CertSignCMS.
≡	CheckMD5 (see page 17)	Checks the MD5 hash value of the memory buffer
≡	CheckSHA1 (see page 18)	Checks the SHA1 hash value of the memory buffer
≡	CheckSHA256 (see page 18)	Checks the SHA256 hash value of the memory buffer
≡	ClearFileAttributesA (see page 19)	This function sets the file attributes to normal.

◆	ClearFileAttributesW (see page 19)	This function sets the file attributes to normal.
◆	ClearUnusedMemory (see page 19)	Clears unused memory and minimized the application memory usage
◆	CloneFont (see page 20)	This is function CloneFont.
◆	CopyNativeBitmap (see page 20)	This is function CopyNativeBitmap.
◆	CreateCardBuffer (see page 20)	Creates XML buffer
◆	CreateNativeBitmap (see page 20)	This is function CreateNativeBitmap.
◆	CreateUnicodeFileA (see page 21)	Creates UNICODE file
◆	CreateUnicodeFileW (see page 21)	Creates UNICODE file
◆	CreateWindowsFont (see page 21)	This is function CreateWindowsFont.
◆	CurrentIPAddressA (see page 21)	Returns the IP address
◆	CurrentIPAddressW (see page 22)	Returns the IP address
◆	DeactivateCard (see page 22)	Terminates a connection between a smart card and a reader
◆	DeactivateCardEx (see page 22)	Terminates a connection between a smart card and a reader
◆	DeallocateBuffer (see page 23)	Deallocates the memory buffer
◆	DeallocateHWND (see page 23)	This function destroys the specified window.
◆	DeallocateHWNDW (see page 23)	This function destroys the specified window.
◆	DecryptFileAESA (see page 24)	Decrypts file using AES algorithm.
◆	DecryptFileAESW (see page 24)	Decrypts file using AES algorithm.
◆	DeleteCardBuffer (see page 25)	Deletes XML buffer
◆	DeleteToRecycleBinA (see page 25)	Deletes file to the Windows Recycle Bin
◆	DeleteToRecycleBinW (see page 25)	Deletes file to Windows Recycle Bin
◆	DestroyFont (see page 26)	This is function DestroyFont.
◆	DestroyImageBuffer (see page 26)	Destroys the memory buffer
◆	DirectoryExistsA (see page 26)	Determines whether a specified directory exists.
◆	DirectoryExistsW (see page 27)	Determines whether a specified directory exists.
◆	DisplayCertificate (see page 27)	Displays the dialog window with certificate information
◆	DpiY (see page 27)	This is function DpiY.
◆	DrawAlphaText (see page 28)	This is function DrawAlphaText.
◆	DrawAlphaTextRect (see page 28)	This is function DrawAlphaTextRect.
◆	DrawLayeredWindow (see page 28)	Repaints the surface of the layered window
◆	DrawNativeBitmap (see page 29)	This is function DrawNativeBitmap.
◆	DrawTextDirect (see page 29)	This is function DrawTextDirect.
◆	DrawTextDirectEx (see page 29)	This is function DrawTextDirectEx.
◆	DrawTextGlow (see page 29)	This is function DrawTextGlow.
◆	DrawTextLine (see page 30)	This is function DrawTextLine.
◆	DrawTextOutline (see page 30)	This is function DrawTextOutline.
◆	DrawTextRect (see page 30)	This is function DrawTextRect.
◆	EmptyRecycleBin (see page 30)	Empties the recycle bin
◆	EmToPixels (see page 31)	This is function EmToPixels.

EncodeCertificate (see page 31)	Performs Base64 encoding of the certificate
EncodePhoto (see page 31)	Performs Base64 encoding of the photo
EncryptFileAESA (see page 32)	Encrypts file using AES algorithm.
EncryptFileAESW (see page 32)	Encrypts file using AES algorithm.
FileCloseA (see page 33)	Concludes input/output (I/O) to a file opened using the FileCreateRewrite function.
FileCloseW (see page 33)	Concludes input/output (I/O) to a file opened using the FileCreateRewrite function.
FileCopyA (see page 33)	The CopyFile function copies an existing file to a new file.
FileCopyW (see page 34)	The CopyFile function copies an existing file to a new file.
FileCreateRewriteA (see page 34)	Creates new or overwrites existing file
FileCreateRewriteW (see page 34)	Creates new or overwrites existing file
FileDeleteA (see page 35)	Deletes a file from disk.
FileDeleteW (see page 35)	Deletes a file from disk.
FileExistsA (see page 36)	Tests whether a specified file exists.
FileExistsW (see page 36)	Tests whether a specified file exists.
FileExtensionIsA (see page 36)	Checks the file extension
FileExtensionIsW (see page 37)	Checks the file extension
FileGetSizeA (see page 37)	Retrieves the size of a specified file.
FileGetSizeW (see page 37)	Retrieves the size of a specified file.
FileIsExeA (see page 38)	Checks if the file is a Windows executable
FileIsExeW (see page 38)	Checks if the file is a Windows executable
FileIsIconA (see page 39)	Checks if the file is a Windows icon (.ico) file
FileIsIconW (see page 39)	Checks if the file is a Windows icon (.ico) file
FileIsImageA (see page 39)	Checks if the file is an image file
FileIsImageW (see page 40)	Checks if the file is an image file
FileIsLink (see page 40)	Checks to see if the file specified by file name is a Microsoft Windows shortcut (.Lnk) file (and is neither a file nor a folder).
FileOrFolderExistsA (see page 40)	Checks if the file or folder with the given name exists
FileOrFolderExistsW (see page 41)	Checks if the file or folder with the given name exists
FileRenameA (see page 41)	Renames the file
FileRenameW (see page 41)	Renames the file
FileWriteA (see page 42)	Writes string to the file
FileWriteCharA (see page 42)	Writes one character to the file
FileWriteCharW (see page 42)	Writes one character to the file
FileWriteNewLineA (see page 43)	Writes new line sequence to the file
FileWriteNewLineW (see page 43)	Writes new line sequence to the file
FileWriteW (see page 43)	Writes string to the file
fpreset (see page 44)	This is function fpreset.
FullPathA (see page 44)	Gets the full path to the file based on file name
FullPathW (see page 44)	Gets the full path to the file based on file name
GenerateAuthenticationSignatureA (see page 44)	Generate authentication signature
GenerateAuthenticationSignatureExA (see page 45)	Generate authentication signature
GenerateAuthenticationSignatureExW (see page 45)	Generate authentication signature

GenerateAuthenticationSignatureW (see page 46)	Generate authentication signature
GenerateBMPA (see page 46)	Generates Windows Bitmap file with QR Code image
GenerateBMPW (see page 47)	Generates Windows Bitmap file with QR Code image
GenerateNonRepudiationSignatureA (see page 47)	Generate non repudiation signature
GenerateNonRepudiationSignatureExA (see page 48)	Generate non repudiation signature
GenerateNonRepudiationSignatureExW (see page 48)	Generate non repudiation signature
GenerateNonRepudiationSignatureW (see page 49)	Generate non repudiation signature
GeneratePNGA (see page 49)	Generates PNG file with QR Code image
GeneratePNGW (see page 50)	Generates PNG file with QR Code image
GenerateQRCodeA (see page 50)	Read eID card and save the identity information and address to PNG QR Code file
GenerateQRCodeExA (see page 51)	Read eID card and save the identity information and address to PNG QR Code file
GenerateQRCodeExW (see page 51)	Read eID card and save the identity information and address to PNG QR Code file
GenerateQRCodeW (see page 51)	Read eID card and save the identity information and address to PNG QR Code file
GetAllFiles (see page 52)	Returns the names of files in a specified directory.
GetCardBufferA (see page 52)	This is function GetCardBufferA.
GetCardBufferSize (see page 52)	This function returns the size of the buffer needed to hold the information from the eID card in the XML or CSV format
GetCardBufferW (see page 53)	This is function GetCardBufferW.
GetCardSerialNumber (see page 53)	This is function GetCardSerialNumber.
GetCardVersion (see page 53)	Get the applet version number for card in the reader with specified number
GetEncodedCertificateSize (see page 54)	Returns the size of the Base64 encoded certificate
GetEncodedPhotoSize (see page 54)	Calculates buffer size for Base64 encoded photo
GetFileMD5A (see page 54)	Gets the MD5 hash value for the file
GetFileMD5W (see page 55)	Gets the MD5 hash value for the file
GetFilesCountA (see page 55)	Calculates the number of files in the given folder
GetFilesCountW (see page 56)	Calculates the number of files in the given folder
GetFileSHA1A (see page 56)	Gets the SHA1 hash value for the file
GetFileSHA1W (see page 56)	Gets the SHA1 hash value for the file
GetFileSHA256A (see page 57)	Gets the SHA256 hash value for the file
GetFileSHA256W (see page 57)	Gets the SHA256 hash value for the file
GetHBitmapA (see page 58)	Generates Windows Bitmap in memory with QR Code image
GetHBitmapW (see page 58)	Generates Windows Bitmap in memory with QR Code image
GetISOCodeA (see page 59)	Returns the country ISO code based on the nationality string
GetISOCodeW (see page 59)	Returns the country ISO code based on the nationality string
GetMD5 (see page 60)	Gets the MD5 hash value for the content of the memory buffer
GetPNGA (see page 60)	Writes PNG image to the memory buffer.
GetPNGW (see page 61)	Writes PNG image to the memory buffer.
GetReaderIndexA (see page 61)	Returns the zero-based reader index with specified name
GetReaderIndexW (see page 61)	Returns the zero-based reader index with specified name
GetReaderNameA (see page 62)	Returns the name of the card reader
GetReaderNameLenA (see page 62)	Returns the length of the reader name

GetReaderNameLenW (see page 63)	Returns the length of the reader name
GetReaderNameW (see page 63)	Returns the name of the card reader
GetReadersCount (see page 63)	Get number of card readers connected to PC
GetSelectedReaderIndex (see page 64)	Returns the index of the active smart card reader
GetSHA1 (see page 64)	Gets the SHA1 hash value for the content of the memory buffer
GetSHA256 (see page 64)	Gets the SHA256 hash value for the content of the memory buffer
GetStartupA (see page 65)	Checks if the application is registered to run when Windows starts
GetStartupW (see page 65)	Checks if the application is registered to run when Windows starts
GetSupportSIS (see page 66)	Checks if the SIS cards are supported by the engine
GetTextLineSize (see page 66)	This is function GetTextLineSize.
GetTextSize (see page 66)	This is function GetTextSize.
GetTextSizeEx (see page 66)	This is function GetTextSizeEx.
HibernateWindows (see page 67)	Hibernates Windows
IsAnimatedGIFA (see page 67)	Checks if the file is an animated GIF image file
IsAnimatedGIFW (see page 67)	Checks if the file is an animated GIF image file
IsCardActivated (see page 68)	Checks the connection between a smart card and a reader
IsCardActivatedEx (see page 68)	Checks the connection between a smart card and a reader
IsCardPresent (see page 68)	Checks if the card is present in the card reader
IsCardPresentEx (see page 69)	Checks if the card is present in the card reader
IsCardStillInserted (see page 69)	Checks if the card is still inserted in the card reader
IsCardStillInsertedEx (see page 69)	Checks if the card is still inserted in the card reader
IsCitrixSession (see page 70)	Checks if application is running in Citrix session
IsConnectedToInternet (see page 70)	Checks if PC is connected to Internet
IsDirectoryA (see page 70)	Verifies that a path is a valid directory.
IsDirectoryW (see page 70)	Verifies that a path is a valid directory.
IsEIDCard (see page 71)	Check if Belgian EID card is inserted into card reader
IsEIDCardEx (see page 71)	Check if Belgian EID card is inserted into card reader
IsEngineActive (see page 71)	Checks if the Swelio Engine is activated
IsFemaleA (see page 72)	Checks if the card owner is female
IsFemaleW (see page 72)	Checks if the card owner is female
IsMaleA (see page 72)	Checks if the card owner is male
IsMaleW (see page 73)	Checks if the card owner is male
IsMediaCenter (see page 73)	Checks if the Media Center version of Windows is installed
IsMetroActive (see page 73)	Checks if metro interface is active
IsMultiTouchReady (see page 74)	Checks if the system is multi touch ready
IsNativeWin64 (see page 74)	Checks if the application is native 64 bit executable
IsRemoteSession (see page 74)	Checks if application is running in RDP session
IsSISCard (see page 74)	Check if Belgian SIS card is inserted into card reader
IsSISCardEx (see page 75)	Check if Belgian SIS card is inserted into card reader
IsTabletPC (see page 75)	Checks if the application is running on the Tablet PC
IsUnicodeFileA (see page 75)	Checks if the file is UNICODE file
IsUnicodeFileW (see page 76)	Checks if the file is UNICODE file
IsValidFileNameA (see page 76)	Checks if provided string is a valid file name
IsValidFileNameW (see page 76)	Checks if provided string is a valid file name
IsValidPathNameA (see page 77)	Checks if provided string is a valid file path
IsValidPathNameW (see page 77)	Checks if provided string is a valid file path
IsWindows10 (see page 78)	Checks if PC is running Windows 10 or better
IsWindows7 (see page 78)	Checks if PC is running Windows 7 or better

IsWindows8 (see page 78)	Checks if PC is Running Windows 8 or better
IsWindowsVista (see page 78)	Checks if PC is running Windows Vista or better
IsWindowsXP (see page 79)	Checks if PC is running Windows XP
IsWindowsXPSP2 (see page 79)	Checks if PC is running Windows XP with Service Pack 2 installed
IsWow64 (see page 79)	Checks if the 32 bit application runs on 64 bit Windows
LayeredWndProcA (see page 79)	The default window procedure for the layered window
LayeredWndProcW (see page 79)	The default window procedure for the layered window
LoadBitmapJPG (see page 80)	This is function LoadBitmapJPG.
LoadBitmapPNG (see page 80)	This is function LoadBitmapPNG.
LoadCertificateA (see page 80)	Reads the certificate from a file
LoadCertificateW (see page 81)	Reads the certificate from a file
LoadIdentityA (see page 81)	Reads the raw identity information from a file
LoadIdentityW (see page 81)	Reads the raw identity information from a file
LoadPhotoA (see page 82)	Loads photo from a file
LoadPhotoW (see page 82)	Loads photo from a file
LoadPNGResource (see page 82)	This is function LoadPNGResource.
MakeCompatibleBitmap (see page 83)	This is function MakeCompatibleBitmap.
MakeSoundFromFileA (see page 83)	Plays the wave sound from the file
MakeSoundFromFileW (see page 83)	Plays the wave sound from the file
MakeSoundFromResourceA (see page 84)	Plays the wave sound from the resource
MakeSoundFromResourceW (see page 84)	Plays the wave sound from the resource
PointsToPixels (see page 84)	This is function PointsToPixels.
PortAvailable (see page 85)	Checks if the port with specified number is available
ReadAddressA (see page 85)	Read address information from Belgian eID card
ReadAddressExA (see page 85)	Read address information from Belgian eID card
ReadAddressExW (see page 86)	Read address information from Belgian eID card
ReadAddressW (see page 86)	Read address information from Belgian eID card
ReadAuthenticationCertificate (see page 86)	Read Authentication Certificate to memory
ReadAuthenticationCertificateEx (see page 87)	Read Authentication Certificate to memory
ReadBufferFromFileA (see page 87)	Reads the content of the file to the memory buffer
ReadBufferFromFileW (see page 87)	Reads the content of the file to the memory buffer
ReadCaCertificate (see page 88)	Read Ca Certificate to memory
ReadCaCertificateEx (see page 88)	Read Ca Certificate to memory
ReadIdentityA (see page 89)	Read identity information from Belgian eID card
ReadIdentityExA (see page 89)	Read identity information from Belgian eID card
ReadIdentityExW (see page 89)	Read identity information from Belgian eID card
ReadIdentityW (see page 90)	Read identity information from Belgian eID card
ReadNonRepudiationCertificate (see page 90)	Read Non Repudiation Certificate to memory
ReadNonRepudiationCertificateEx (see page 91)	Read Non Repudiation Certificate to memory
ReadPhoto (see page 91)	Reads a photo from a card
ReadPhotoAsBitmap (see page 91)	Reads the picture from the card, converts it to bitmap and returns the bitmap handle Description: Reads the photo from the Belgian eID card and returns the bitmap handle Reading the photo from the card is a time consuming operation. Do it only when needed.

ReadPhotoAsBitmapEx (see page 92)	Reads the picture from the card, converts it to bitmap and returns the bitmap handle Description: Reads the photo from the Belgian eID card and returns the Windows bitmap handle Reading the photo from the card is a time consuming operation. Do it only when needed.
ReadPhotoEx (see page 92)	Reads a photo from a card
ReadRootCaCertificate (see page 93)	Read Root Ca Certificate to memory
ReadRootCaCertificateEx (see page 93)	Read Root Ca Certificate to memory
ReadRrnCertificate (see page 93)	Read Rrn Certificate to memory
ReadRrnCertificateEx (see page 94)	Read Rrn Certificate to memory
ReadSISCardA (see page 94)	Read Belgian SIS card.
ReadSISCardExA (see page 95)	Read Belgian SIS card.
ReadSISCardExW (see page 95)	Read Belgian SIS card.
ReadSISCardW (see page 95)	Read Belgian SIS card.
RecycleBinEmpty (see page 96)	Returns TRUE if Windows Recycle Bin is empty
ReloadReadersList (see page 96)	Reloads the list of the available card readers
RemoveCallback (see page 96)	Remove callback procedure for card events
RemoveStartupA (see page 97)	Removes the application from the list of the automatically started applications
RemoveStartupW (see page 97)	Removes the application from the list of the automatically started applications
RestoreWindowSubclassA (see page 97)	Restores window standard procedure
RestoreWindowSubclassW (see page 98)	Restores window standard procedure
SaveAuthenticationCertificateA (see page 98)	Save Authentication Certificate to a file
SaveAuthenticationCertificateExW (see page 98)	Save Authentication Certificate to a file
SaveAuthenticationCertificateW (see page 99)	Save Authentication Certificate to a file
SaveCaCertificateA (see page 99)	Save Ca Certificate to a file
SaveCaCertificateExW (see page 99)	Save Ca Certificate to a file
SaveCaCertificateW (see page 100)	Save Ca Certificate to a file
SaveCardToToXMLStreamExA (see page 100)	Read eID card and save the information to XML buffer
SaveCardToToXMLStreamExW (see page 100)	Read eID card and save the information to XML buffer
SaveCardToXmlA (see page 101)	Read eID card and save the information to XML file
SaveCardToXmlExA (see page 101)	Read eID card and save the information to XML file
SaveCardToXmlExW (see page 102)	Read eID card and save the information to XML file
SaveCardToXmlW (see page 102)	Read eID card and save the information to XML file
SaveIdentityA (see page 103)	Saves identity information to a file
SaveIdentityW (see page 103)	Saves identity information to a file
SaveNonRepudiationCertificateA (see page 103)	Save Non Repudiation Certificate to a file
SaveNonRepudiationCertificateExW (see page 104)	Save Non Repudiation Certificate to a file
SaveNonRepudiationCertificateW (see page 104)	Save Non Repudiation Certificate to a file
SavePersonCsvToStreamA (see page 104)	This is function SavePersonCsvToStreamA.

SavePersonCsvToStreamW (see page 105)	This is function SavePersonCsvToStreamW.
SavePersonToCsvA (see page 105)	Read eID card and save the identity information and address to CSV file
SavePersonToCsvExA (see page 105)	Read eID card and save the identity information and address to CSV file
SavePersonToCsvExW (see page 106)	Read eID card and save the identity information and address to CSV file
SavePersonToCsvW (see page 106)	Read eID card and save the identity information and address to CSV file
SavePhotoA (see page 106)	Save photo to a file
SavePhotoAsBitmapA (see page 107)	Save the picture from the card to Windows Bitmap file Description: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsBitmapExA (see page 107)	Reads the picture from the card and saves it to Windows Bitmap file Description: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsBitmapExW (see page 108)	Reads the picture from the card and saves it to Windows Bitmap file Description: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsBitmapW (see page 108)	Save the picture from the card to Windows Bitmap file Description: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsJpegA (see page 108)	Save the picture from the card to JPG file Description: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsJpegExA (see page 109)	Save the picture from the card to JPG file Description: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsJpegExW (see page 109)	Save the picture from the card to JPG file Description: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsJpegW (see page 109)	Save the picture from the card to JPG file Description: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoW (see page 110)	Saves photo to a file
SaveRootCaCertificateA (see page 110)	Save Root Ca Certificate to a file
SaveRootCaCertificateExW (see page 111)	Save Root Ca Certificate to a file
SaveRootCaCertificateW (see page 111)	Save Root Ca Certificate to a file
SaveRrnCertificateA (see page 111)	Save RRN Certificate to a file
SaveRrnCertificateExW (see page 112)	Save RRN Certificate to a file
SaveRrnCertificateW (see page 112)	Save RRN Certificate to a file
SelectReader (see page 112)	When more than 1 reader connected, select the reader with specified number

✚	SelectReaderByNameA (see page 113)	Select active smart card reader by providing the reader name
✚	SelectReaderByNameW (see page 113)	Select active smart card reader by providing the reader name
✚	SendAPDU (see page 114)	This is function SendAPDU.
✚	SetCallback (see page 114)	Activates callback procedure for card status change event
✚	SetMWCompatibility (see page 114)	Set the compatibility mode with the old version of the official EID MiddleWare
✚	SetStartupA (see page 114)	Register application to run when Windows starts
✚	SetStartupW (see page 115)	Register application to run when Windows starts
✚	SetSupportSIS (see page 115)	Activates or deactivates SIS card support by engine
✚	ShellCopyFileA (see page 115)	Copies file to the new location
✚	ShellCopyFileW (see page 116)	Copies file to the new location
✚	ShutdownWindows (see page 116)	Logs off the interactive user, shuts down the system.
✚	StartEngine (see page 117)	Activates the Swelio Engine.
✚	StopEngine (see page 117)	Deactivates the Swelio Engine
✚	StretchNativeBitmap (see page 117)	This is function StretchNativeBitmap.
✚	StripFileNameA (see page 118)	Replaces environment variable names with values
✚	StripFileNameW (see page 118)	Replaces environment variable names with values
✚	SuspendWindows (see page 118)	Suspends Windows
✚	TurnMonitorOff (see page 119)	Turns the monitor off
✚	TurnMonitorOn (see page 119)	Turns the monitor on
✚	UpdateWindowPosition (see page 119)	Updated the window position
✚	VerifyPinA (see page 119)	Verify PIN code
✚	VerifyPinExA (see page 120)	Verify PIN code
✚	VerifyPinExW (see page 120)	Verify PIN code
✚	VerifyPinW (see page 120)	Verify PIN code
✚	VerifySignature (see page 121)	Verifies the signature from the specified hash value.
✚	WriteBufferToFileA (see page 121)	Writes the memory buffer to file
✚	WriteBufferToFileW (see page 122)	Writes the memory buffer to file

1.1.1 ActivateCard Function

Established communication between the card and the reader

C++

```
BOOL WINAPI ActivateCard( );
```

File

Swelio.h (see page 156)

Returns

Returns TRUE if the card is activated, otherwise returns FALSE

Description

The ActivateCard function establishes a connection between the calling application and a smart card contained by a specific reader. If no card exists in the specified reader, an error is returned.

Example

```
if (IsCardPresent())
{
```

```
    ActivateCard();  
}
```

1.1.2 ActivateCardEx Function

Established communication between the card and the reader

C++

```
BOOL WINAPI ActivateCardEx(int readerNumber);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader

Returns

Returns TRUE if the card is activated, otherwise returns FALSE

Description

The `ActivateCard` (see page 9) function establishes a connection between the calling application and a smart card contained by a specific reader. If no card exists in the specified reader, an error is returned.

1.1.3 AddRemoveMessageFilter Function

Adds or removes a message from the User Interface Privilege Isolation (UIPI) message filter.

C++

```
void WINAPI AddRemoveMessageFilter(UINT message, DWORD dwFlags);
```

File

System.h (see page 161)

Parameters

Parameters	Description
UINT message	Specifies the message to add to or remove from the filter.
DWORD dwFlags	Specifies the action to be performed. One of the following values. <ul style="list-style-type: none">MSGFLT_ADD - Adds the message to the filter. This has the effect of allowing the message to be received.MSGFLT_REMOVE - Removes the message from the filter. This has the effect of blocking the message.

Description

This function changes the message filter for Windows Vista or better. UIPI is a security feature that prevents messages from being received from a lower integrity level sender. All such messages with a value above `WM_USER` are blocked by default. The filter, somewhat contrary to intuition, is a list of messages that are allowed through. Therefore, adding a message to the filter allows that message to be received from a lower integrity sender, while removing a message blocks that message from being received.

Certain messages with a value less than `WM_USER` are required to pass through the filter regardless of the filter setting. You can call this function to remove one of those messages from the filter and it will return TRUE. However, the message

will still be received by the calling process.

1.1.4 AllocateBuffer Function

Allocates the buffer in memory

C++

```
void* WINAPI AllocateBuffer(int bufferSize);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
int bufferSize	The size of the buffer

Returns

The pointer to the allocated memory block

Description

AllocateBuffer allocates a block of the given size on the heap, and returns the address of this memory. The bytes of the allocated buffer are not set to zero. To dispose of the buffer, use DeallocateBuffer (see page 23) function.

1.1.5 AllocateDefaultHWNDW Function

This function creates the invisible tool window

C++

```
HWND WINAPI AllocateDefaultHWNDW();
```

File

System.h (see page 161)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the invisible zero-size tool window that can be used for internal purposes, like processing the special Windows messages for synchronization, etc...

1.1.6 AllocateDefaultHWNDW Function

This function creates the invisible tool window

C++

```
HWND WINAPI AllocateDefaultHWNDW();
```

File

System.h (see page 161)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the invisible zero-size tool window that can be used for internal purposes, like processing the special Windows messages for synchronization, etc...

1.1.7 AllocateHWND A Function

This function creates the invisible tool window using the provided window procedure

C++

```
HWND WINAPI AllocateHWND A(LONG_PTR method) ;
```

File

System.h (🔗 see page 161)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the invisible zero-size tool window that can be used for internal purposes, like processing the special Windows messages for synchronization, etc...

1.1.8 AllocateHWND W Function

This function creates the invisible tool window using the provided window procedure

C++

```
HWND WINAPI AllocateHWND W(LONG_PTR method) ;
```

File

System.h (🔗 see page 161)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the invisible zero-size tool window that can be used for internal purposes, like processing the special Windows messages for synchronization, etc...

1.1.9 AllocateLayeredWindow A Function

This function creates the layered window using the provided window class name

C++

```
HWND WINAPI AllocateLayeredWindow A(LPCSTR className) ;
```

File

System.h (🔗 see page 161)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the layered window using the provided window class name

1.1.10 AllocateLayeredWindowW Function

This function creates the layered window using the provided window class name

C++

```
HWND WINAPI AllocateLayeredWindowW(LPCWSTR className);
```

File

System.h (see page 161)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the layered window using the provided window class name

1.1.11 AllocateWindowClassA Function

This function creates the standard window using the provided window class name

C++

```
HWND WINAPI AllocateWindowClassA(LPCSTR className);
```

File

System.h (see page 161)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the standard window using the provided window class name

1.1.12 AllocateWindowClassW Function

This function creates the standard window using the provided window class name

C++

```
HWND WINAPI AllocateWindowClassW(LPCWSTR className);
```

File

System.h (see page 161)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the standard window using the provided window class name

1.1.13 AlphaBlendBitmap Function

This is function AlphaBlendBitmap.

C++

```
void WINAPI AlphaBlendBitmap(HBITMAP src, HDC hdc, int left, int top, int width, int height, int alpha);
```

File

Graphics.h (🔗 see page 155)

1.1.14 AlphaBlendNative Function

This is function AlphaBlendNative.

C++

```
void WINAPI AlphaBlendNative(HDC hdcDest, int xoriginDest, int yoriginDest, int wDest, int hDest, HDC hdcSrc, int xoriginSrc, int yoriginSrc, int wSrc, int hSrc, BYTE alpha);
```

File

Graphics.h (🔗 see page 155)

1.1.15 BringWindowToFront Function

This function brings the specified window to the top of the z-order.

C++

```
void WINAPI BringWindowToFront(HWND window);
```

File

System.h (🔗 see page 161)

Parameters

Parameters	Description
HWND window	Handle to the window to bring to the top of the z-order.

1.1.16 CardDecryptFileA Function

Decrypt file using Belgian Id card

C++

```
BOOL WINAPI CardDecryptFileA(LPSTR szSource, LPSTR szDestination);
```

File

Encryption.h (🔗 see page 152)

Parameters

Parameters	Description
LPSTR szSource	The name of the encrypted file
LPSTR szDestination	The name of the decrypted file

Description

Decrypt file which was encrypted using CardEncryptFile function

1.1.17 CardDecryptFileW Function

Decrypt file using Belgian Id card

C++

```
BOOL WINAPI CardDecryptFileW(LPWSTR szSource, LPWSTR szDestination);
```

File

Encryption.h (see page 152)

Parameters

Parameters	Description
LPWSTR szSource	The name of the encrypted file
LPWSTR szDestination	The name of the decrypted file

Description

Decrypt file which was encrypted using CardEncryptFile function

1.1.18 CardEncryptFileA Function

Encrypt file using Belgian Id card

C++

```
BOOL WINAPI CardEncryptFileA(LPSTR szSource, LPSTR szDestination);
```

File

Encryption.h (see page 152)

Parameters

Parameters	Description
LPSTR szSource	The name of the source file
LPSTR szDestination	The name of the encrypted file

Description

Encrypt file using Belgian Id card. The card must be inserted in the reader

1.1.19 CardEncryptFileW Function

Encrypt file using Belgian Id card

C++

```
BOOL WINAPI CardEncryptFileW(LPWSTR szSource, LPWSTR szDestination);
```

File

Encryption.h (see page 152)

Parameters

Parameters	Description
LPWSTR szSource	The name of the source file
LPWSTR szDestination	The name of the encrypted file

Description

Encrypt file using Belgian Id card. The card must be inserted in the reader

1.1.20 CardSignCadesT Function

Sign data with eID card according to CADES-T standard

C++

```
BOOL WINAPI CardSignCadesT(int readerNumber, BYTE * data, UINT dataLen, BYTE * signature, UINT * signatureLen);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
BYTE * data	the data to sign
UINT dataLen	the size of the data buffer
BYTE * signature	the signature buffer
UINT * signatureLen	the size of the signature buffer

Returns

Returns true if the operation is successful, otherwise returns false

Description

Create CADES-T signature for data buffer. Can be used for digital signature of PDF documents in combination with external PDF library

1.1.21 CardSignCMS Function

Sign data with eID card according to CMS standard

C++

```
BOOL WINAPI CardSignCMS(int readerNumber, BYTE * data, UINT dataLen, BYTE * signature, UINT * signatureLen);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
BYTE * data	the data to sign
UINT dataLen	the size of the data buffer
BYTE * signature	the signature buffer
UINT * signatureLen	the size of the signature buffer

Returns

Returns true if the operation is successful, otherwise returns false

Description

Create CMS signature for data buffer. Can be used for digital signature of PDF documents in combination with external PDF library

1.1.22 CertSignCadesT Function

This is function CertSignCadesT.

C++

```
BOOL WINAPI CertSignCadesT(LPWSTR certificate, LPWSTR password, BYTE * data, UINT dataLen,  
BYTE * signature, UINT * signatureLen);
```

File

Swelio.h ([see page 156](#))

1.1.23 CertSignCMS Function

This is function CertSignCMS.

C++

```
BOOL WINAPI CertSignCMS(LPWSTR certificate, LPWSTR password, BYTE * data, UINT dataLen,  
BYTE * signature, UINT * signatureLen);
```

File

Swelio.h ([see page 156](#))

1.1.24 CheckMD5 Function

Checks the MD5 hash value of the memory buffer

C++

```
BOOL WINAPI CheckMD5(BYTE* source, int sourceSize, BYTE* buffer, int bufferSize);
```

File

Encryption.h ([see page 152](#))

Parameters

Parameters	Description
BYTE* source	The source bytes
int sourceSize	The size of the source buffer
BYTE* buffer	The hash value buffer
int bufferSize	The size of the hash value buffer

Returns

Returns TRUE if the hash value is correct, otherwise returns false

Description

This function checks if the provided value of the hash is valid

1.1.25 CheckSHA1 Function

Checks the SHA1 hash value of the memory buffer

C++

```
BOOL WINAPI CheckSHA1(BYTE* source, int sourceSize, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 152)

Parameters

Parameters	Description
BYTE* source	The source bytes
int sourceSize	The size of the source buffer
BYTE* buffer	The hash value buffer
int bufferSize	The size of the hash value buffer

Returns

Returns TRUE if the hash value is correct, otherwise returns false

Description

This function checks if the provided value of the hash is valid

1.1.26 CheckSHA256 Function

Checks the SHA256 hash value of the memory buffer

C++

```
BOOL WINAPI CheckSHA256(BYTE* source, int sourceSize, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 152)

Parameters

Parameters	Description
BYTE* source	The source bytes
int sourceSize	The size of the source buffer

BYTE* buffer	The hash value buffer
int bufferSize	The size of the hash value buffer

Returns

Returns TRUE if the hash value is correct, otherwise returns false

Description

This function checks if the provided value of the hash is valid

1.1.27 ClearFileAttributesA Function

This function sets the file attributes to normal.

C++

```
void WINAPI ClearFileAttributesA(LPSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPSTR fileName	The name of the file

Description

This function removed additional file attributes, like system, read-only and hidden.

1.1.28 ClearFileAttributesW Function

This function sets the file attributes to normal.

C++

```
void WINAPI ClearFileAttributesW(LPWSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file

Description

This function removed additional file attributes, like system, read-only and hidden.

1.1.29 ClearUnusedMemory Function

Clears unused memory and minimized the application memory usage

C++

```
void WINAPI ClearUnusedMemory();
```

File

System.h ([↗](#) see page 161)

1.1.30 CloneFont Function

This is function CloneFont.

C++

```
HFONT WINAPI CloneFont(HFONT hFont);
```

File

Graphics.h ([↗](#) see page 155)

1.1.31 CopyNativeBitmap Function

This is function CopyNativeBitmap.

C++

```
void WINAPI CopyNativeBitmap(HBITMAP src, HDC dstDC, int width, int height, int left, int top);
```

File

Graphics.h ([↗](#) see page 155)

1.1.32 CreateCardBuffer Function

Creates XML buffer

C++

```
void* WINAPI CreateCardBuffer();
```

File

Swelio.h ([↗](#) see page 156)

Returns

The memory buffer to store information

Description

Use this function to create XML buffer

1.1.33 CreateNativeBitmap Function

This is function CreateNativeBitmap.

C++

```
HBITMAP WINAPI CreateNativeBitmap(INT width, INT height, __deref_opt_out void ** ppvBits);
```

File

Graphics.h (🔗 see page 155)

1.1.34 CreateUnicodeFileA Function

Creates UNICODE file

C++

```
void WINAPI CreateUnicodeFileA(LPCSTR fileName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPCSTR fileName	The name of the file

1.1.35 CreateUnicodeFileW Function

Creates UNICODE file

C++

```
void WINAPI CreateUnicodeFileW(LPCWSTR fileName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPCWSTR fileName	The name of the file

1.1.36 CreateWindowsFont Function

This is function CreateWindowsFont.

C++

```
HFONT WINAPI CreateWindowsFont(LPCWSTR fontFamily, INT size, INT fontStyle, INT fontQuality);
```

File

Graphics.h (🔗 see page 155)

1.1.37 CurrentIPAddressA Function

Returns the IP address

C++

```
void WINAPI CurrentIPAddressA(LPSTR address, UINT len);
```

File

SystemInfo.h (see page 162)

1.1.38 CurrentIPAddressW Function

Returns the IP address

C++

```
void WINAPI CurrentIPAddressW(LPWSTR address, UINT len);
```

File

SystemInfo.h (see page 162)

1.1.39 DeactivateCard Function

Terminates a connection between a smart card and a reader

C++

```
void WINAPI DeactivateCard();
```

File

Swelio.h (see page 156)

Description

The DeactivateCard function terminates a connection previously opened between the calling application and a smart card in the target reader.

1.1.40 DeactivateCardEx Function

Terminates a connection between a smart card and a reader

C++

```
void WINAPI DeactivateCardEx(int readerNumber);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader

Description

The DeactivateCard (see page 22) function terminates a connection previously opened between the calling application and a smart card in the target reader.

1.1.41 DeallocateBuffer Function

Deallocates the memory buffer

C++

```
void WINAPI DeallocateBuffer(void* buffer);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
void* buffer	The pointer to the memory buffer

Description

DeallocateBuffer frees a memory block previously allocated with AllocateBuffer (see page 11). Use this procedure to dispose of a memory block obtained with AllocateBuffer (see page 11).

1.1.42 DeallocateHWND A Function

This function destroys the specified window.

C++

```
BOOL WINAPI DeallocateHWND A (HWND hwnd);
```

File

System.h (see page 161)

Parameters

Parameters	Description
HWND hwnd	Handle to the window to be destroyed

Description

This function restores the window default procedure and destroys the window

1.1.43 DeallocateHWNDW Function

This function destroys the specified window.

C++

```
BOOL WINAPI DeallocateHWNDW (HWND hwnd);
```

File

System.h (see page 161)

Parameters

Parameters	Description
HWND hwnd	Handle to the window to be destroyed

Description

This function restores the window default procedure and destroys the window

1.1.44 DecryptFileAESA Function

Decrypts file using AES algorithm.

C++

```
BOOL WINAPI DecryptFileAESA(LPSTR szSource, LPSTR szDestination, LPSTR szPassword);
```

File

Encryption.h (📄 see page 152)

Parameters

Parameters	Description
LPSTR szSource	The source file name
LPSTR szDestination	The decrypted file name
LPSTR szPassword	The password

Returns

Returns TRUE if the file is successfully decrypted, otherwise returns FALSE.

Description

Use this function to decrypt the file using AES algorithm

1.1.45 DecryptFileAESW Function

Decrypts file using AES algorithm.

C++

```
BOOL WINAPI DecryptFileAESW(LPWSTR szSource, LPWSTR szDestination, LPWSTR szPassword);
```

File

Encryption.h (📄 see page 152)

Parameters

Parameters	Description
LPWSTR szSource	The source file name
LPWSTR szDestination	The decrypted file name
LPWSTR szPassword	The password

Returns

Returns TRUE if the file is successfully decrypted, otherwise returns FALSE.

Description

Use this function to decrypt the file using AES algorithm

1.1.46 DeleteCardBuffer Function

Deletes XML buffer

C++

```
void WINAPI DeleteCardBuffer(void* buffer);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
void* buffer	The memory buffer to store information

Description

Use this function to delete the buffer

1.1.47 DeleteToRecycleBinA Function

Deletes file to the Windows Recycle Bin

C++

```
void WINAPI DeleteToRecycleBinA(LPSTR fileName, BOOL silent);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
BOOL silent	Do not display a progress dialog box

Description

Use this function to delete the file to Windows Recycle Bin

1.1.48 DeleteToRecycleBinW Function

Deletes file to Windows Recycle Bin

C++

```
void WINAPI DeleteToRecycleBinW(LPWSTR fileName, BOOL silent);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
BOOL silent	Do not display a progress dialog box

Description

Use this function to delete the file to Windows Recycle Bin

1.1.49 DestroyFont Function

This is function DestroyFont.

C++

```
void WINAPI DestroyFont(HFONT hFont);
```

File

Graphics.h ([↗](#) see page 155)

1.1.50 DestroyImageBuffer Function

Destroys the memory buffer

C++

```
void WINAPI DestroyImageBuffer(void* buffer);
```

File

quricol.h ([↗](#) see page 156)

Parameters

Parameters	Description
void* buffer	The memory buffer

Description

Destroys the memory buffer created to hold the image returned by GetPNGA ([↗](#) see page 60) (Ansi) or GetPNGW ([↗](#) see page 61) (Unicode) functions

1.1.51 DirectoryExistsA Function

Determines whether a specified directory exists.

C++

```
BOOL WINAPI DirectoryExistsA(LPSTR fileName);
```

File

FileOperations.h ([↗](#) see page 153)

Parameters

Parameters	Description
LPSTR fileName	The name of the directory

Returns

Returns TRUE if the directory exists, otherwise returns FALSE

Description

Call DirectoryExists to determine whether the directory specified by the Directory parameter exists. If the directory exists, the function returns True. If the directory does not exist, the function returns False. If a full path name is entered, DirectoryExists searches for the directory along the designated path. Otherwise, the Directory parameter is interpreted as a relative path name from the current directory.

1.1.52 DirectoryExistsW Function

Determines whether a specified directory exists.

C++

```
BOOL WINAPI DirectoryExistsW(LPWSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPWSTR fileName	The name of the directory

Returns

Returns TRUE if the directory exists, otherwise returns FALSE

Description

Call DirectoryExists to determine whether the directory specified by the Directory parameter exists. If the directory exists, the function returns True. If the directory does not exist, the function returns False. If a full path name is entered, DirectoryExists searches for the directory along the designated path. Otherwise, the Directory parameter is interpreted as a relative path name from the current directory.

1.1.53 DisplayCertificate Function

Displays the dialog window with certificate information

C++

```
void WINAPI DisplayCertificate(PeIdCertificate certificate);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PeIdCertificate certificate	The certificate data

Description

Use this function to show the certificate for the user

1.1.54 DpiY Function

This is function DpiY.

C++

```
int WINAPI DpiY();
```

File

Graphics.h (🔗 see page 155)

1.1.55 DrawAlphaText Function

This is function DrawAlphaText.

C++

```
void WINAPI DrawAlphaText(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF color, int width, int height, UINT flags);
```

File

Graphics.h (🔗 see page 155)

1.1.56 DrawAlphaTextRect Function

This is function DrawAlphaTextRect.

C++

```
void WINAPI DrawAlphaTextRect(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF color, LPRECT lpRect, UINT flags);
```

File

Graphics.h (🔗 see page 155)

1.1.57 DrawLayeredWindow Function

Repaints the surface of the layered window

C++

```
void WINAPI DrawLayeredWindow(HWND handle, int left, int top, int width, int height, HDC buffer, COLORREF colorKey, byte alpha, BOOL redrawOnly);
```

File

System.h (🔗 see page 161)

Parameters

Parameters	Description
HWND handle	The handle of the window
int left	The horizontal coordinate of the window
int top	The vertical coordinate of the window
int width	The width of the window
int height	The height of the window
HDC buffer	Handle to a DC for the surface that defines the layered window

COLORREF colorKey	COLORREF structure that specifies the color key to be used when composing the layered window.
byte alpha	Specifies an alpha transparency value to be used on the entire source bitmap
BOOL redrawOnly	Only redraw and do not update the window position

1.1.58 DrawNativeBitmap Function

This is function DrawNativeBitmap.

C++

```
void WINAPI DrawNativeBitmap(HBITMAP src, HBITMAP dst, int width, int height);
```

File

Graphics.h (🔗 see page 155)

1.1.59 DrawTextDirect Function

This is function DrawTextDirect.

C++

```
void WINAPI DrawTextDirect(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF color, int left, int top);
```

File

Graphics.h (🔗 see page 155)

1.1.60 DrawTextDirectEx Function

This is function DrawTextDirectEx.

C++

```
void WINAPI DrawTextDirectEx(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF color, COLORREF background, int left, int top);
```

File

Graphics.h (🔗 see page 155)

1.1.61 DrawTextGlow Function

This is function DrawTextGlow.

C++

```
void WINAPI DrawTextGlow(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF foreColor, int left, int top);
```

File

Graphics.h (🔗 see page 155)

1.1.62 DrawTextLine Function

This is function DrawTextLine.

C++

```
void WINAPI DrawTextLine(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF color, int left, int top);
```

File

Graphics.h (📄 see page 155)

1.1.63 DrawTextOutline Function

This is function DrawTextOutline.

C++

```
void WINAPI DrawTextOutline(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF foreColor, COLORREF backColor, int left, int top);
```

File

Graphics.h (📄 see page 155)

1.1.64 DrawTextRect Function

This is function DrawTextRect.

C++

```
void WINAPI DrawTextRect(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF color, COLORREF background, LPRECT lpRect, UINT flags);
```

File

Graphics.h (📄 see page 155)

1.1.65 EmptyRecycleBin Function

Empties the recycle bin

C++

```
void WINAPI EmptyRecycleBin();
```

File

System.h (📄 see page 161)

Description

Removes all files from the Windows recycle bin

1.1.66 EmToPixels Function

This is function EmToPixels.

C++

```
int WINAPI EmToPixels(int em);
```

File

Graphics.h (🔗 see page 155)

1.1.67 EncodeCertificate Function

Performs Base64 encoding of the certificate

C++

```
int WINAPI EncodeCertificate(PEidCertificate certificate, BYTE* buffer, int bufferSize);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
PEidCertificate certificate	The certificate data
BYTE* buffer	The Base64 encoded certificate buffer
int bufferSize	The size of the buffer

Returns

Returns the size of the buffer needed to hold the encoded certificate

1.1.68 EncodePhoto Function

Performs Base64 encoding of the photo

C++

```
int WINAPI EncodePhoto(PEidPicture photo, BYTE* buffer, int bufferSize);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
PEidPicture photo	The pointer to EidPicture (🔗 see page 132) structure
BYTE* buffer	The Base64 encoded photo buffer
int bufferSize	The size of the buffer

Returns

Returns the size of the buffer needed to hold the encoded photo

Description

Use this function for Base64 encoding of the photo

1.1.69 EncryptFileAESA Function

Encrypts file using AES algorithm.

C++

```
BOOL WINAPI EncryptFileAESA(LPSTR szSource, LPSTR szDestination, LPSTR szPassword);
```

File

Encryption.h (📄 see page 152)

Parameters

Parameters	Description
LPSTR szSource	The source file name
LPSTR szDestination	The encrypted file name
LPSTR szPassword	The password

Returns

Returns TRUE if the file is successfully encrypted, otherwise returns FALSE

Description

Use this function to encrypt the file using AES algorithm

1.1.70 EncryptFileAESW Function

Encrypts file using AES algorithm.

C++

```
BOOL WINAPI EncryptFileAESW(LPWSTR szSource, LPWSTR szDestination, LPWSTR szPassword);
```

File

Encryption.h (📄 see page 152)

Parameters

Parameters	Description
LPWSTR szSource	The source file name
LPWSTR szDestination	The encrypted file name
LPWSTR szPassword	The password

Returns

Returns TRUE if the file is successfully encrypted, otherwise returns FALSE

Description

Use this function to encrypt the file using AES algorithm

1.1.71 FileCloseA Function

Concludes input/output (I/O) to a file opened using the FileCreateRewrite function.

C++

```
void WINAPI FileCloseA(HANDLE handle) ;
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
HANDLE handle	The handle of the file

Description

Closes the file handle of the specified file when its not in use anymore

1.1.72 FileCloseW Function

Concludes input/output (I/O) to a file opened using the FileCreateRewrite function.

C++

```
void WINAPI FileCloseW(HANDLE handle) ;
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
HANDLE handle	The handle of the file

Description

Closes the file handle of the specified file when its not in use anymore

1.1.73 FileCopyA Function

The CopyFile function copies an existing file to a new file.

C++

```
BOOL WINAPI FileCopyA(LPSTR oldName, LPSTR newName) ;
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPSTR oldName	The name of the source file
LPSTR newName	The name of the destination file

Returns

The result of the function is TRUE when the file is successfully copied to the new location, otherwise the result is FALSE.

Description

This function makes a copy of the file with the new name or path.

1.1.74 FileCopyW Function

The CopyFile function copies an existing file to a new file.

C++

```
BOOL WINAPI FileCopyW(LPWSTR oldName, LPWSTR newName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPWSTR oldName	The name of the source file
LPWSTR newName	The name of the destination file

Returns

The result of the function is TRUE when the file is successfully copied to the new location, otherwise the result is FALSE.

Description

This function makes a copy of the file with the new name or path.

1.1.75 FileCreateRewriteA Function

Creates new or overwrites existing file

C++

```
HANDLE WINAPI FileCreateRewriteA(LPCSTR fileName);
```

File

FileOperations.h (see page 153)

Returns

The result of the function is the handle of the file

Description

This function creates the new file with provided file name if the file with given name does not exists. If the file exists, it will be overwritten and the current content of the file will be lost

1.1.76 FileCreateRewriteW Function

Creates new or overwrites existing file

C++

```
HANDLE WINAPI FileCreateRewriteW(LPCWSTR fileName);
```

File

FileOperations.h (see page 153)

Returns

The result of the function is the handle of the file

Description

This function creates the new file with provided file name if the file with given name does not exists. If the file exists, it will be overwritten and the current content of the file will be lost

1.1.77 FileDeleteA Function

Deletes a file from disk.

C++

```
void WINAPI FileDeleteA(LPSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPSTR fileName	The name of the file

Description

DeleteFile deletes the file named by fileName from the disk.

1.1.78 FileDeleteW Function

Deletes a file from disk.

C++

```
void WINAPI FileDeleteW(LPWSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file

Description

DeleteFile deletes the file named by fileName from the disk.

1.1.79 FileExistsA Function

Tests whether a specified file exists.

C++

```
BOOL WINAPI FileExistsA(LPSTR fileName);
```

File
FileOperations.h (see page 153)

Returns
FileExists returns TRUE if the file specified by FileName exists. If the file does not exist, FileExists returns FALSE.

Description
Use this function to check if the file with provided name exists.

1.1.80 FileExistsW Function

Tests whether a specified file exists.

C++

```
BOOL WINAPI FileExistsW(LPWSTR fileName);
```

File
FileOperations.h (see page 153)

Returns
FileExists returns TRUE if the file specified by FileName exists. If the file does not exist, FileExists returns FALSE.

Description
Use this function to check if the file with provided name exists.

1.1.81 FileExtensionIsA Function

Checks the file extension

C++

```
BOOL WINAPI FileExtensionIsA(LPCSTR fileName, LPCSTR ext);
```

File
FileOperations.h (see page 153)

Parameters

Parameters	Description
LPCSTR fileName	The name of the file
LPCSTR ext	The file name extension

Returns
Returns true if the file has a specified extension, otherwise returns false.

Description

This function checks if the file has a given extension

1.1.82 FileExtensionIsW Function

Checks the file extension

C++

```
BOOL WINAPI FileExtensionIsW(LPCWSTR fileName, LPCWSTR ext);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPCWSTR fileName	The name of the file
LPCWSTR ext	The file name extension

Returns

Returns true if the file has a specified extension, otherwise returns false.

Description

This function checks if the file has a given extension

1.1.83 FileGetSizeA Function

Retrieves the size of a specified file.

C++

```
DWORD WINAPI FileGetSizeA(LPCSTR fileName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPCSTR fileName	The name of the file

Returns

The size of the file in bytes.

Description

This function determines the size of the file specified by the file name.

1.1.84 FileGetSizeW Function

Retrieves the size of a specified file.

C++

```
DWORD WINAPI FileGetSizeW(LPCWSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPCWSTR fileName	The name of the file

Returns

The size of the file in bytes.

Description

This function determines the size of the file specified by the name of the file

1.1.85 FileIsExeA Function

Checks if the file is a Windows executable

C++

```
BOOL WINAPI FileIsExeA(LPSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPSTR fileName	The name of the file

Returns

Returns TRUE if the file is a Windows executable, otherwise returns FALSE.

1.1.86 FileIsExeW Function

Checks if the file is a Windows executable

C++

```
BOOL WINAPI FileIsExeW(LPWSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file

Returns

Returns TRUE if the file is a Windows executable, otherwise returns FALSE.

1.1.87 FileIsIconA Function

Checks if the file is a Windows icon (.ico) file

C++

```
BOOL WINAPI FileIsIconA(LPSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPSTR fileName	The name of the file

Returns

Returns TRUE if the file is a Windows icon (.ico) file, otherwise returns FALSE.

1.1.88 FileIsIconW Function

Checks if the file is a Windows icon (.ico) file

C++

```
BOOL WINAPI FileIsIconW(LPWSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file

Returns

Returns TRUE if the file is a Windows icon (.ico) file, otherwise returns FALSE.

1.1.89 FileIsImageA Function

Checks if the file is an image file

C++

```
BOOL WINAPI FileIsImageA(LPCSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPCSTR fileName	The name of the file

Returns

Returns TRUE if the file is an image file, otherwise returns FALSE.

1.1.90 FileIsImageW Function

Checks if the file is an image file

C++

```
BOOL WINAPI FileIsImageW(LPCWSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPCWSTR fileName	The name of the file

Returns

Returns TRUE if the file is an image file, otherwise returns FALSE.

1.1.91 FileIsLink Function

Checks to see if the file specified by file name is a Microsoft Windows shortcut (.Lnk) file (and is neither a file nor a folder).

C++

```
BOOL WINAPI FileIsLink(LPCWSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPCWSTR fileName	The name of the file to check

Returns

Returns TRUE if the file is a Microsoft Windows shortcut, otherwise returns FALSE

Description

This function is used to check if the file with provided file name is a Windows shortcut or not

1.1.92 FileOrFolderExistsA Function

Checks if the file or folder with the given name exists

C++

```
BOOL WINAPI FileOrFolderExistsA(LPSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPSTR fileName	The file or folder name

Returns

Returns TRUE if the file or folder exists, otherwise returns FALSE.

1.1.93 FileOrFolderExistsW Function

Checks if the file or folder with the given name exists

C++

```
BOOL WINAPI FileOrFolderExistsW(LPWSTR fileName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPWSTR fileName	The file or folder name

Returns

Returns TRUE if the file or folder exists, otherwise returns FALSE.

1.1.94 FileRenameA Function

Renames the file

C++

```
BOOL WINAPI FileRenameA(LPSTR oldName, LPSTR newName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPSTR oldName	File to be renamed
LPSTR newName	New name of the file

Returns

Returns TRUE if the file was successfully renamed, otherwise returns FALSE

1.1.95 FileRenameW Function

Renames the file

C++

```
BOOL WINAPI FileRenameW(LPWSTR oldName, LPWSTR newName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPWSTR oldName	File to be renamed
LPWSTR newName	New name of the file

Returns

Returns TRUE if the file was successfully renamed, otherwise returns FALSE

1.1.96 FileWriteA Function

Writes string to the file

C++

```
void WINAPI FileWriteA(HANDLE handle, LPSTR text);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
HANDLE handle	The handle of the file
LPSTR text	The text to write

1.1.97 FileWriteCharA Function

Writes one character to the file

C++

```
void WINAPI FileWriteCharA(HANDLE handle, CHAR text);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
HANDLE handle	The handle of the file
CHAR text	The character to write

1.1.98 FileWriteCharW Function

Writes one character to the file

C++

```
void WINAPI FileWriteCharW(HANDLE handle, WCHAR text);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
HANDLE handle	The handle of the file
WCHAR text	The character to write

1.1.99 FileWriteNewLineA Function

Writes new line sequence to the file

C++

```
void WINAPI FileWriteNewLineA(HANDLE handle);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
HANDLE handle	The handle of the file

1.1.100 FileWriteNewLineW Function

Writes new line sequence to the file

C++

```
void WINAPI FileWriteNewLineW(HANDLE handle);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
HANDLE handle	The handle of the file

1.1.101 FileWriteW Function

Writes string to the file

C++

```
void WINAPI FileWriteW(HANDLE handle, LPWSTR text);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
HANDLE handle	The handle of the file

LPWSTR text	The text to write
-------------	-------------------

1.1.102 fpreset Function

This is function fpreset.

C++

```
void fpreset() ;
```

File

System.h (🔗 see page 161)

1.1.103 FullPathA Function

Gets the full path to the file based on file name

C++

```
BOOL WINAPI FullPathA(LPSTR fileName, LPSTR fullName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
LPSTR fullName	The full path to the file

1.1.104 FullPathW Function

Gets the full path to the file based on file name

C++

```
BOOL WINAPI FullPathW(LPWSTR fileName, LPWSTR fullName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
LPWSTR fullName	The full path to the file

1.1.105 GenerateAuthenticationSignatureA Function

Generate authentication signature

C++

```
BOOL WINAPI GenerateAuthenticationSignatureA(LPSTR pinCode, BYTE* dataHash, int hashSize,
```

```
BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
LPSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate authentication signature using provided hash value

1.1.106 GenerateAuthenticationSignatureExA Function

Generate authentication signature

C++

```
BOOL WINAPI GenerateAuthenticationSignatureExA(int readerNumber, LPSTR pinCode, BYTE* dataHash, int hashSize, BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate authentication signature using provided hash value

1.1.107 GenerateAuthenticationSignatureExW Function

Generate authentication signature

C++

```
BOOL WINAPI GenerateAuthenticationSignatureExW(int readerNumber, LPWSTR pinCode, BYTE* dataHash, int hashSize, BYTE* signature, LPDWORD signatureSize);
```


File

Swelio.h (📄 see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate authentication signature using provided hash value

1.1.108 GenerateAuthenticationSignatureW Function

Generate authentication signature

C++

```
BOOL WINAPI GenerateAuthenticationSignatureW(LPWSTR pinCode, BYTE* dataHash, int hashSize,
BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (📄 see page 156)

Parameters

Parameters	Description
LPWSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate authentication signature using provided hash value

1.1.109 GenerateBMPA Function

Generates Windows Bitmap file with QR Code image

C++

```
void WINAPI GenerateBMPA(LPSTR fileName, LPSTR text, int margin, int size, int level);
```

File

quicol.h (🔗 see page 156)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
LPSTR text	The text to encode
int margin	The margin from the border in points
int size	The size of the one point in pixels
int level	The error correction level

Description

Generate Windows Bitmap file with encoded text as QR Code image

1.1.110 GenerateBMPW Function

Generates Windows Bitmap file with QR Code image

C++

```
void WINAPI GenerateBMPW(LPWSTR fileName, LPWSTR text, int margin, int size, int level);
```

File

quicol.h (🔗 see page 156)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
LPWSTR text	The text to encode
int margin	The margin from the border in points
int size	The size of the one point in pixels
int level	The error correction level

Description

Generate Windows Bitmap file with encoded text as QR Code image

1.1.111 GenerateNonRepudiationSignatureA Function

Generate non repudiation signature

C++

```
BOOL WINAPI GenerateNonRepudiationSignatureA(LPSTR pinCode, BYTE* dataHash, int hashSize, BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
LPSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer

int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate non repudiation signature using provided hash value

1.1.112 GenerateNonRepudiationSignatureExA Function

Generate non repudiation signature

C++

```
BOOL WINAPI GenerateNonRepudiationSignatureExA(int readerNumber, LPSTR pinCode, BYTE* dataHash, int hashSize, BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate non repudiation signature using provided hash value

1.1.113 GenerateNonRepudiationSignatureExW Function

Generate non repudiation signature

C++

```
BOOL WINAPI GenerateNonRepudiationSignatureExW(int readerNumber, LPWSTR pinCode, BYTE* dataHash, int hashSize, BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR pinCode	The card PIN code value

BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate non repudiation signature using provided hash value

1.1.114 GenerateNonRepudiationSignatureW Function

Generate non repudiation signature

C++

```
BOOL WINAPI GenerateNonRepudiationSignatureW(LPWSTR pinCode, BYTE* dataHash, int hashSize,
BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate non repudiation signature using provided hash value

1.1.115 GeneratePNGA Function

Generates PNG file with QR Code image

C++

```
void WINAPI GeneratePNGA(LPSTR fileName, LPSTR text, int margin, int size, int level);
```

File

quircol.h (see page 156)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
LPSTR text	The text to encode
int margin	The margin from the border in points

int size	The size of the one point in pixels
int level	The error correction level

Description

Generates PNG file with encoded text as QR Code image

1.1.116 GeneratePNGW Function

Generates PNG file with QR Code image

C++

```
void WINAPI GeneratePNGW(LPWSTR fileName, LPWSTR text, int margin, int size, int level);
```

File

quicol.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
LPWSTR text	The text to encode
int margin	The margin from the border in points
int size	The size of the one point in pixels
int level	The error correction level

Description

Generates PNG file with encoded text as QR Code image

1.1.117 GenerateQRCodeA Function

Read eID card and save the identity information and address to PNG QR Code file

C++

```
BOOL WINAPI GenerateQRCodeA(LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and generate the QR Code PNG image

1.1.118 GenerateQRCodeExA Function

Read eID card and save the identity information and address to PNG QR Code file

C++

```
BOOL WINAPI GenerateQRCodeExA(int readerNumber, LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and generate the QR Code PNG image

1.1.119 GenerateQRCodeExW Function

Read eID card and save the identity information and address to PNG QR Code file

C++

```
BOOL WINAPI GenerateQRCodeExW(int readerNumber, LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and generate the QR Code PNG image

1.1.120 GenerateQRCodeW Function

Read eID card and save the identity information and address to PNG QR Code file

C++

```
BOOL WINAPI GenerateQRCodeW(LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and generate the QR Code PNG image

1.1.121 GetAllFiles Function

Returns the names of files in a specified directory.

C++

```
void WINAPI GetAllFiles(FOLDERENUMPROC lpEnumProc, LPWSTR folderName, LPWSTR searchMask, LPARAM lParam);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
FOLDERENUMPROC lpEnumProc	Callback function
LPWSTR folderName	The name of the folder
LPWSTR searchMask	The search string to match against the names of files in path.
LPARAM lParam	Specifies an application-defined value to be passed to the callback function

Description

This function enumerates all files in the specified folder which names match the searchMask parameter and calls the callback function passing the name of the file to it.

1.1.122 GetCardBufferA Function

This is function GetCardBufferA.

C++

```
void WINAPI GetCardBufferA(void* buffer, void* strDest, int count);
```

File

Swelio.h (see page 156)

1.1.123 GetCardBufferSize Function

This function returns the size of the buffer needed to hold the information from the eID card in the XML or CSV format

C++

```
int WINAPI GetCardBufferSize(void* buffer);
```

File

Swelio.h (see page 156)

Returns

The size of the XML or CSV buffer

Description

Use this function to get the size of the XML buffer before allocating the buffer in memory

1.1.124 GetCardBufferW Function

This is function GetCardBufferW.

C++

```
void WINAPI GetCardBufferW(void* buffer, void* strDest, int count);
```

File

Swelio.h (see page 156)

1.1.125 GetCardSerialNumber Function

This is function GetCardSerialNumber.

C++

```
BOOL WINAPI GetCardSerialNumber(int readerNumber, BYTE* serialNumber, LPDWORD serialNumberSize);
```

File

Swelio.h (see page 156)

1.1.126 GetCardVersion Function

Get the applet version number for card in the reader with specified number

C++

```
int WINAPI GetCardVersion(int readerNumber);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The reader index, starting from 0

Returns

The card applet version number

Description

Get the version number of eid card applet using the zero-based reader index. The first reader has number 0, second - 1, etc... You can read the information only from one selected reader at once.

1.1.127 GetEncodedCertificateSize Function

Returns the size of the Base64 encoded certificate

C++

```
int WINAPI GetEncodedCertificateSize(PEidCertificate certificate);
```

File

Swelio.h ([see page 156](#))

Parameters

Parameters	Description
PEidCertificate certificate	The certificate data

Returns

Returns the size of the buffer needed to hold the encoded certificate

Description

Use this function to calculate the size of the buffer needed to encode the certificate

1.1.128 GetEncodedPhotoSize Function

Calculates buffer size for Base64 encoded photo

C++

```
int WINAPI GetEncodedPhotoSize(PeidPicture photo);
```

File

Swelio.h ([see page 156](#))

Parameters

Parameters	Description
PeidPicture photo	The pointer to EidPicture (see page 132) structure

Returns

The desired size of the buffer

Description

Use this function to calculate the size of the buffer needed for Base64 encoding of the photo This can be useful for including the photo data to the text document, for example to XML file

1.1.129 GetFileMD5A Function

Gets the MD5 hash value for the file

C++

```
BOOL WINAPI GetFileMD5A(LPSTR fileName, BYTE* buffer, int bufferSize);
```

File

Encryption.h (🔗 see page 152)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
BYTE* buffer	The buffer to store the hash value
int bufferSize	The size of the buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates MD5 hash value for the given file

1.1.130 GetFileMD5W Function

Gets the MD5 hash value for the file

C++

```
BOOL WINAPI GetFileMD5W(LPWSTR fileName, BYTE* buffer, int bufferSize);
```

File

Encryption.h (🔗 see page 152)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
BYTE* buffer	The buffer to store the hash value
int bufferSize	The size of the buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates MD5 hash value for the given file

1.1.131 GetFilesCountA Function

Calculates the number of files in the given folder

C++

```
int WINAPI GetFilesCountA(LPSTR folderName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPSTR folderName	The name of the folder

Returns

The number of files in the given folder

1.1.132 GetFilesCountW Function

Calculates the number of files in the given folder

C++

```
int WINAPI GetFilesCountW(LPWSTR folderName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPWSTR folderName	The name of the folder

Returns

The number of files in the given folder

1.1.133 GetFileSHA1A Function

Gets the SHA1 hash value for the file

C++

```
BOOL WINAPI GetFileSHA1A(LPSTR fileName, BYTE* buffer, int bufferSize);
```

File

Encryption.h (🔗 see page 152)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
BYTE* buffer	The buffer to store the hash value
int bufferSize	The size of the buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates SHA1 hash value for the given file

1.1.134 GetFileSHA1W Function

Gets the SHA1 hash value for the file

C++

```
BOOL WINAPI GetFileSHA1W(LPWSTR fileName, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 152)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
BYTE* buffer	The buffer to store the hash value
int bufferSize	The size of the buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates SHA1 hash value for the given file

1.1.135 GetFileSHA256A Function

Gets the SHA256 hash value for the file

C++

```
BOOL WINAPI GetFileSHA256A(LPSTR fileName, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 152)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
BYTE* buffer	The buffer to store the hash value
int bufferSize	The size of the buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates SHA256 hash value for the given file

1.1.136 GetFileSHA256W Function

Gets the SHA256 hash value for the file

C++

```
BOOL WINAPI GetFileSHA256W(LPWSTR fileName, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 152)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
BYTE* buffer	The buffer to store the hash value
int bufferSize	The size of the buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates SHA256 hash value for the given file

1.1.137 GetHBitmapA Function

Generates Windows Bitmap in memory with QR Code image

C++

```
HBITMAP WINAPI GetHBitmapA(LPSTR text, int margin, int size, int level);
```

File

quricol.h (see page 156)

Parameters

Parameters	Description
LPSTR text	The text to encode
int margin	The margin from the border in points
int size	The size of the one point in pixels
int level	The error correction level

Returns

The result of the function is HBITMAP handle. You have to destroy it by yourself when its not needed anymore.

Description

Generates Windows Bitmap in memory file with encoded text as QR Code image

1.1.138 GetHBitmapW Function

Generates Windows Bitmap in memory with QR Code image

C++

```
HBITMAP WINAPI GetHBitmapW(LPWSTR text, int margin, int size, int level);
```

File

quricol.h (see page 156)

Parameters

Parameters	Description
LPWSTR text	The text to encode
int margin	The margin from the border in points
int size	The size of the one point in pixels

int level	The error correction level
-----------	----------------------------

Returns

The result of the function is HBITMAP handle. You have to destroy it by yourself when its not needed anymore.

Description

Generates Windows Bitmap in memory file with encoded text as QR Code image

1.1.139 GetISOCodeA Function

Returns the country ISO code based on the nationality string

C++

```
BOOL WINAPI GetISOCodeA(LPCSTR nationality, LPSTR iso, int bufferSize);
```

File

NationalityConverter.h (🔗 see page 156)

Parameters

Parameters	Description
LPCSTR nationality	The nationality string
LPSTR iso	The ISO code memory buffer
int bufferSize	The size if the memory buffer

Returns

Returns TRUE if the ISO code is successfully obtained; FALSE otherwise

Description

This function converts the nationality string stored on ID card to the country ISO code

1.1.140 GetISOCodeW Function

Returns the country ISO code based on the nationality string

C++

```
BOOL WINAPI GetISOCodeW(LPCWSTR nationality, LPWSTR iso, int bufferSize);
```

File

NationalityConverter.h (🔗 see page 156)

Parameters

Parameters	Description
LPCWSTR nationality	The nationality string
LPWSTR iso	The ISO code memory buffer
int bufferSize	The size if the memory buffer

Returns

Returns TRUE if the ISO code is successfully obtained; FALSE otherwise

Description

This function converts the nationality string stored on ID card to the country ISO code

1.1.141 GetMD5 Function

Gets the MD5 hash value for the content of the memory buffer

C++

```
BOOL WINAPI GetMD5(BYTE* source, int sourceSize, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 152)

Parameters

Parameters	Description
BYTE* source	The source memory block
int sourceSize	The size of the source memory block
BYTE* buffer	The buffer for the hash value
int bufferSize	The size of the destination buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates MD5 hash value for the given memory buffer

1.1.142 GetPNGA Function

Writes PNG image to the memory buffer.

C++

```
void WINAPI GetPNGA(LPSTR text, int margin, int size, int level, LPINT bufSize, __deref_opt_out void ** ppvBits);
```

File

quicol.h (see page 156)

Parameters

Parameters	Description
LPSTR text	The text to encode
int margin	The margin from the image border in points
int size	The size of the point in pixels
int level	The error correction level
LPINT bufSize	The size of the output buffer
__deref_opt_out void ** ppvBits	The buffer when the resulting image is stored

Description

Writes PNG image to the memory buffer. Can be useful for web development.

1.1.143 GetPNGW Function

Writes PNG image to the memory buffer.

C++

```
void WINAPI GetPNGW(LPWSTR text, int margin, int size, int level, LPINT bufSize,
__deref_opt_out void ** ppvBits);
```

File

quicol.h (see page 156)

Parameters

Parameters	Description
LPWSTR text	The text to encode
int margin	The margin from the image border in points
int size	The size of the point in pixels
int level	The error correction level
LPINT bufSize	The size of the output buffer
__deref_opt_out void ** ppvBits	The buffer when the resulting image is stored

Description

Writes PNG image to the memory buffer. Can be useful for web development.

1.1.144 GetReaderIndexA Function

Returns the zero-based reader index with specified name

C++

```
int WINAPI GetReaderIndexA(LPSTR readerName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR readerName	The name of the reader

Returns

The zero-based reader index

Description

Use this function to get the zero-based index of the card reader with specified name

1.1.145 GetReaderIndexW Function

Returns the zero-based reader index with specified name

C++

```
int WINAPI GetReaderIndexW(LPWSTR readerName);
```


File

Swelio.h (📄 see page 156)

Parameters

Parameters	Description
LPWSTR readerName	The name of the reader

Returns

The zero-based reader index

Description

Use this function to get the zero-based index of the card reader with specified name

1.1.146 GetReaderNameA Function

Returns the name of the card reader

C++

```
int WINAPI GetReaderNameA(int readerNumber, LPSTR strDest, int count);
```

File

Swelio.h (📄 see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
LPSTR strDest	Destination string
int count	The number of characters to be copied

Returns

Returns the reader name length

Description

Returns the name of the card reader with the specified zero-based index

1.1.147 GetReaderNameLenA Function

Returns the length of the reader name

C++

```
int WINAPI GetReaderNameLenA(int readerNumber);
```

File

Swelio.h (📄 see page 156)

Returns

The length of the reader name

Description

Returns the length of the reader name for the smart card reader with specified zero-based index

1.1.148 GetReaderNameLenW Function

Returns the length of the reader name

C++

```
int WINAPI GetReaderNameLenW(int readerNumber);
```

File

Swelio.h (see page 156)

Returns

The length of the reader name

Description

Returns the length of the reader name for the smart card reader with specified zero-based index

1.1.149 GetReaderNameW Function

Returns the name of the card reader

C++

```
int WINAPI GetReaderNameW(int readerNumber, LPWSTR strDest, int count);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
LPWSTR strDest	Destination string
int count	Number of characters to be copied

Returns

The character count of the reader name

Description

Returns the name of the card reader with the specified zero-based index

1.1.150 GetReadersCount Function

Get number of card readers connected to PC

C++

```
int WINAPI GetReadersCount(VOID);
```

File

Swelio.h (see page 156)

Returns

The number of the connected smart card readers

Description

Checks how many smart card readers are connected to PC. If there is no readers connected then the usage of the Swelio Engine is not possible. The engine can control the change of the number of the card readers and can raise an event when the reader is connected or disconnected from PC

1.1.151 GetSelectedReaderIndex Function

Returns the index of the active smart card reader

C++

```
int WINAPI GetSelectedReaderIndex();
```

File

Swelio.h (see page 156)

Returns

The index of the selected card reader. The first reader has index 0.

Description

The zero-based index of the selected card reader. If there is only one reader is connected to PC then this reader has the index 0 and it's a default (selected) reader.

1.1.152 GetSHA1 Function

Gets the SHA1 hash value for the content of the memory buffer

C++

```
BOOL WINAPI GetSHA1(BYTE* source, int sourceSize, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 152)

Parameters

Parameters	Description
BYTE* source	The source memory block
int sourceSize	The size of the source memory block
BYTE* buffer	The buffer for the hash value
int bufferSize	The size of the destination buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates SHA1 hash value for the given memory buffer

1.1.153 GetSHA256 Function

Gets the SHA256 hash value for the content of the memory buffer

C++

```
BOOL WINAPI GetSHA256(BYTE* source, int sourceSize, BYTE* buffer, int bufferSize);
```

File

Encryption.h (🔗 see page 152)

Parameters

Parameters	Description
BYTE* source	The source memory block
int sourceSize	The size of the source memory block
BYTE* buffer	The buffer for the hash value
int bufferSize	The size of the destination buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates SHA256 hash value for the given memory buffer

1.1.154 GetStartupA Function

Checks if the application is registered to run when Windows starts

C++

```
BOOL WINAPI GetStartupA(LPCSTR appName);
```

File

System.h (🔗 see page 161)

Parameters

Parameters	Description
LPCSTR appName	The name of the application

1.1.155 GetStartupW Function

Checks if the application is registered to run when Windows starts

C++

```
BOOL WINAPI GetStartupW(LPCWSTR appName);
```

File

System.h (🔗 see page 161)

Parameters

Parameters	Description
LPCWSTR appName	The name of the application

1.1.156 GetSupportSIS Function

Checks if the SIS cards are supported by the engine

C++

```
BOOL WINAPI GetSupportSIS ( ) ;
```

File

Swelio.h (🔗 see page 156)

Returns

Returns TRUE if SIS card support is activated, otherwise returns FALSE

Description

The SIS card reading operation takes more time than the reading of the eID card. By default when the card is inserted in the reader the engine will try to detect the card type and the card insertion event will be raised for eID cards only. If you want to support the SIS cards in your application then you have to activate it using SetSupportSIS (🔗 see page 115) function. Use GetSupportSIS function to check if the SIS card support is activated.

1.1.157 GetTextLineSize Function

This is function GetTextLineSize.

C++

```
SIZE WINAPI GetTextLineSize(LPCWSTR s, HFONT hFont) ;
```

File

Graphics.h (🔗 see page 155)

1.1.158 GetTextSize Function

This is function GetTextSize.

C++

```
SIZE WINAPI GetTextSize(LPCWSTR s, HFONT hFont, UINT flags) ;
```

File

Graphics.h (🔗 see page 155)

1.1.159 GetTextSizeEx Function

This is function GetTextSizeEx.

C++

```
SIZE WINAPI GetTextSizeEx(LPCWSTR s, HFONT hFont, UINT proposedWidth, UINT flags, BOOL margins) ;
```

File

Graphics.h (🔗 see page 155)

1.1.160 HibernateWindows Function

Hibernates Windows

C++

```
bool WINAPI HibernateWindows();
```

File

System.h (🔗 see page 161)

1.1.161 IsAnimatedGIFA Function

Checks if the file is an animated GIF image file

C++

```
bool WINAPI IsAnimatedGIFA(LPSTR fileName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPSTR fileName	The name of the file

Returns

Returns TRUE if the file is an animated GIF image file, otherwise returns FALSE.

1.1.162 IsAnimatedGIFW Function

Checks if the file is an animated GIF image file

C++

```
bool WINAPI IsAnimatedGIFW(LPWSTR fileName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file

Returns

Returns TRUE if the file is an animated GIF image file, otherwise returns FALSE.

1.1.163 IsCardActivated Function

Checks the connection between a smart card and a reader

C++

```
BOOL WINAPI IsCardActivated();
```

File

Swelio.h (see page 156)

Description

This function checks the connection between the calling application and a smart card in the target reader.

1.1.164 IsCardActivatedEx Function

Checks the connection between a smart card and a reader

C++

```
BOOL WINAPI IsCardActivatedEx(int readerNumber);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader

Description

This function checks the connection between the calling application and a smart card in the target reader.

1.1.165 IsCardPresent Function

Checks if the card is present in the card reader

C++

```
BOOL WINAPI IsCardPresent();
```

File

Swelio.h (see page 156)

Returns

Returns TRUE if the card is inserted in the reader, otherwise returns FALSE

Description

Use IsCardPresent function to check if the card is inserted in the card reader or not

1.1.166 IsCardPresentEx Function

Checks if the card is present in the card reader

C++

```
BOOL WINAPI IsCardPresentEx(int readerNumber) ;
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader

Returns

Returns TRUE if the card is inserted in the reader, otherwise returns FALSE

Description

Use isCardPresent function to check if the card is inserted in the card reader or not

1.1.167 IsCardStillInserted Function

Checks if the card is still inserted in the card reader

C++

```
BOOL WINAPI IsCardStillInserted() ;
```

File

Swelio.h (see page 156)

Description

This function checks if the card is still present in the card reader

1.1.168 IsCardStillInsertedEx Function

Checks if the card is still inserted in the card reader

C++

```
BOOL WINAPI IsCardStillInsertedEx(int readerNumber) ;
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader

Description

This function checks if the card is still present in the card reader

1.1.169 IsCitrixSession Function

Checks if application is running in Citrix session

C++

```
BOOL WINAPI IsCitrixSession();
```

File

SystemInfo.h (see page 162)

1.1.170 IsConnectedToInternet Function

Checks if PC is connected to Internet

C++

```
BOOL WINAPI IsConnectedToInternet();
```

File

SystemInfo.h (see page 162)

1.1.171 IsDirectoryA Function

Verifies that a path is a valid directory.

C++

```
BOOL WINAPI IsDirectoryA(LPSTR folderName);
```

File

FileOperations.h (see page 153)

Returns

Returns TRUE if the path is a valid directory, or FALSE otherwise.

Description

This function verifies if provided value is the name of the folder

1.1.172 IsDirectoryW Function

Verifies that a path is a valid directory.

C++

```
BOOL WINAPI IsDirectoryW(LPWSTR folderName);
```

File

FileOperations.h (see page 153)

Returns

Returns TRUE if the path is a valid directory, or FALSE otherwise.

Description

This function verifies if provided value is the name of the folder

1.1.173 IsEIDCard Function

Check if Belgian EID card is inserted into card reader

C++

```
BOOL WINAPI IsEIDCard( ) ;
```

File

Swelio.h (🔗 see page 156)

Returns

Returns TRUE, if Belgian eID card is inserted in the reader. If there is no card in the reader or the card of other type is inserted, returns FALSE

Description

If the card is inserted in the reader, this function performs the card type check.

1.1.174 IsEIDCardEx Function

Check if Belgian EID card is inserted into card reader

C++

```
BOOL WINAPI IsEIDCardEx( int readerNumber ) ;
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.

Returns

Returns TRUE, if Belgian eID card is inserted in the reader. If there is no card in the reader or the card of other type is inserted, returns FALSE

Description

If the card is inserted in the reader, this function performs the card type check.

1.1.175 IsEngineActive Function

Checks if the Swelio Engine is activated

C++

```
BOOL WINAPI IsEngineActive( ) ;
```

File

Swelio.h (🔗 see page 156)

Returns

Returns TRUE if the Swelio Engine is active, otherwise returns FALSE.

Description

This function checks if the Engine already activated using the StartEngine (see page 117) function.

1.1.176 IsFemaleA Function

Checks if the card owner is female

C++

```
BOOL WINAPI IsFemaleA(PEidIdentityA identity);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PEidIdentityA identity	The person identity information structure

Returns

Returns TRUE if the card owner is female, otherwise returns FALSE

Description

Use this function to check the gender of the card owner

1.1.177 IsFemaleW Function

Checks if the card owner is female

C++

```
BOOL WINAPI IsFemaleW(PEidIdentityW identity);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PEidIdentityW identity	The person identity information structure

Returns

Returns TRUE if the card owner is female, otherwise returns FALSE

Description

Use this function to check the gender of the card owner

1.1.178 IsMaleA Function

Checks if the card owner is male

C++

```
BOOL WINAPI IsMaleA(PEidIdentityA identity);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
PEidIdentityA identity	The person identity information structure

Returns

Returns TRUE if the card owner is male, otherwise returns FALSE

Description

Use this function to check the gender of the card owner

1.1.179 IsMaleW Function

Checks if the card owner is male

C++

```
BOOL WINAPI IsMaleW(PEidIdentityW identity);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
PEidIdentityW identity	The person identity information structure

Returns

Returns TRUE if the card owner is male, otherwise returns FALSE

Description

Use this function to check the gender of the card owner

1.1.180 IsMediaCenter Function

Checks if the Media Center version of Windows is installed

C++

```
BOOL WINAPI IsMediaCenter();
```

File

SystemInfo.h (🔗 see page 162)

1.1.181 IsMetroActive Function

Checks if metro interface is active

C++

```
BOOL WINAPI IsMetroActive();
```

File

SystemInfo.h ([see page 162](#))

1.1.182 IsMultiTouchReady Function

Checks if the system is multi touch ready

C++

```
BOOL WINAPI IsMultiTouchReady();
```

File

SystemInfo.h ([see page 162](#))

1.1.183 IsNativeWin64 Function

Checks if the application is native 64 bit executable

C++

```
BOOL WINAPI IsNativeWin64();
```

File

SystemInfo.h ([see page 162](#))

1.1.184 IsRemoteSession Function

Checks if application is running in RDP session

C++

```
BOOL WINAPI IsRemoteSession();
```

File

SystemInfo.h ([see page 162](#))

1.1.185 IsSISCard Function

Check if Belgian SIS card is inserted into card reader

C++

```
BOOL WINAPI IsSISCard();
```

File

Swelio.h ([see page 156](#))

Returns

Returns TRUE, if Belgian SIS card is inserted in the reader. If there is no card in the reader or the card of other type is

inserted, returns FALSE

Description

If the card is inserted in the reader, this function performs the card type check.

1.1.186 IsSISCardEx Function

Check if Belgian SIS card is inserted into card reader

C++

```
BOOL WINAPI IsSISCardEx(int readerNumber);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.

Returns

Returns TRUE, if Belgian SIS card is inserted in the reader. If there is no card in the reader or the card of other type is inserted, returns FALSE

Description

If the card is inserted in the reader, this function performs the card type check.

1.1.187 IsTabletPC Function

Checks if the application is running on the Tablet PC

C++

```
BOOL WINAPI IsTabletPC();
```

File

SystemInfo.h (see page 162)

1.1.188 IsUnicodeFileA Function

Checks if the file is UNICODE file

C++

```
BOOL WINAPI IsUnicodeFileA(LPCSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPCSTR fileName	The name of the file

Returns

Returns TRUE if file is stored in UNICODE format, otherwise returns FALSE.

Description

This function checks the file encoding based on BOM (Byte Order Mark).

1.1.189 IsUnicodeFileW Function

Checks if the file is UNICODE file

C++

```
BOOL WINAPI IsUnicodeFileW(LPCWSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPCWSTR fileName	The name of the file

Returns

Returns TRUE if file is stored in UNICODE format, otherwise returns FALSE.

Description

This function checks the file encoding based on BOM (Byte Order Mark).

1.1.190 IsValidFileNameA Function

Checks if provided string is a valid file name

C++

```
BOOL WINAPI IsValidFileNameA(LPSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPSTR fileName	The name of the file

Returns

Returns TRUE if provided string is valid file name, otherwise returns FALSE

Description

Checks if provided string is a valid file name and does not contain any illegal characters

1.1.191 IsValidFileNameW Function

Checks if provided string is a valid file name

C++

```
BOOL WINAPI IsValidFileNameW(LPWSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file

Returns

Returns TRUE if provided string is valid file name, otherwise returns FALSE

Description

Checks if provided string is a valid file name and does not contain any illegal characters

1.1.192 IsValidPathNameA Function

Checks if provided string is a valid file path

C++

```
BOOL WINAPI IsValidPathNameA(LPSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPSTR fileName	The file path to check

Returns

Returns TRUE if provided string is valid file path, otherwise returns FALSE

Description

Checks if provided string is a valid file path and does not contain any illegal characters

1.1.193 IsValidPathNameW Function

Checks if provided string is a valid file path

C++

```
BOOL WINAPI IsValidPathNameW(LPWSTR fileName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPWSTR fileName	The file path to check

Returns

Returns TRUE if provided string is valid file path, otherwise returns FALSE

Description

Checks if provided string is a valid file path and does not contain any illegal characters

1.1.194 IsWindows10 Function

Checks if PC is running Windows 10 or better

C++

```
BOOL WINAPI IsWindows10( ) ;
```

File

SystemInfo.h ([↗](#) see page 162)

1.1.195 IsWindows7 Function

Checks if PC is running Windows 7 or better

C++

```
BOOL WINAPI IsWindows7( ) ;
```

File

SystemInfo.h ([↗](#) see page 162)

1.1.196 IsWindows8 Function

Checks if PC is Running Windows 8 or better

C++

```
BOOL WINAPI IsWindows8( ) ;
```

File

SystemInfo.h ([↗](#) see page 162)

1.1.197 IsWindowsVista Function

Checks if PC is running Windows Vista or better

C++

```
BOOL WINAPI IsWindowsVista( ) ;
```

File

SystemInfo.h ([↗](#) see page 162)

1.1.198 IsWindowsXP Function

Checks if PC is running Windows XP

C++

```
BOOL WINAPI IsWindowsXP();
```

File

SystemInfo.h (see page 162)

1.1.199 IsWindowsXPSP2 Function

Checks if PC is running Windows XP with Service Pack 2 installed

C++

```
BOOL WINAPI IsWindowsXPSP2();
```

File

SystemInfo.h (see page 162)

1.1.200 IsWow64 Function

Checks if the 32 bit application runs on 64 bit Windows

C++

```
BOOL WINAPI IsWow64();
```

File

SystemInfo.h (see page 162)

1.1.201 LayeredWndProcA Function

The default window procedure for the layered window

C++

```
LRESULT CALLBACK LayeredWndProcA(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam);
```

File

System.h (see page 161)

1.1.202 LayeredWndProcW Function

The default window procedure for the layered window

C++

```
LRESULT CALLBACK LayeredWndProcW(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam);
```

File

System.h (🔗 see page 161)

1.1.203 LoadBitmapJPG Function

This is function LoadBitmapJPG.

C++

```
HBITMAP WINAPI LoadBitmapJPG(LPCWSTR szFile, LPINT lpiWidth, LPINT lpiHeight, __deref_opt_out void **ppvBits);
```

File

Graphics.h (🔗 see page 155)

1.1.204 LoadBitmapPNG Function

This is function LoadBitmapPNG.

C++

```
HBITMAP WINAPI LoadBitmapPNG(LPCWSTR szFile, LPINT lpiWidth, LPINT lpiHeight, __deref_opt_out void **ppvBits);
```

File

Graphics.h (🔗 see page 155)

1.1.205 LoadCertificateA Function

Reads the certificate from a file

C++

```
void WINAPI LoadCertificateA(LPSTR fileName, PEidCertificate certificate);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
LPSTR fileName	The source file name
PEidCertificate certificate	The pointer to EidCertificate (🔗 see page 130) structure

Description

Use this function to read the certificate from the file

1.1.206 LoadCertificateW Function

Reads the certificate from a file

C++

```
void WINAPI LoadCertificateW(LPWSTR fileName, PEidCertificate certificate);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	The source file name
PEidCertificate certificate	The pointer to EidCertificate (see page 130) structure

Description

Use this function to read the certificate from the file

1.1.207 LoadIdentityA Function

Reads the raw identity information from a file

C++

```
void WINAPI LoadIdentityA(LPSTR fileName, PEidIdentityA identity);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR fileName	The name of the source file
PEidIdentityA identity	The pointer to EidIdentityA (see page 130) structure

Description

Use this function to read back the identity information stored to the file using SavIdentityA (see page 103) function

1.1.208 LoadIdentityW Function

Reads the raw identity information from a file

C++

```
void WINAPI LoadIdentityW(LPWSTR fileName, PEidIdentityW identity);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	The name of the source file

PEidIdentityW identity	The pointer to EidIdentityW (see page 131) structure
------------------------	--

Description

Use this function to read back the identity information stored to the file using SaveldentityW (see page 103) function

1.1.209 LoadPhotoA Function

Loads photo from a file

C++

```
void WINAPI LoadPhotoA(PeidPicture photo, LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PeidPicture photo	The pointer to EidPicture (see page 132) structure
LPSTR fileName	Destination file name

Description

Loads raw picture data from a file

1.1.210 LoadPhotoW Function

Loads photo from a file

C++

```
void WINAPI LoadPhotoW(PeidPicture photo, LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PeidPicture photo	The pointer to EidPicture (see page 132) structure
LPWSTR fileName	Destination file name

Description

Loads raw picture data from a file

1.1.211 LoadPNGResource Function

This is function LoadPNGResource.

C++

```
HBITMAP WINAPI LoadPNGResource(HMODULE handle, LPCWSTR szName, LPINT lpiWidth, LPINT lpiHeight, __deref_opt_out void ** ppvBits);
```

File

Graphics.h ([see page 155](#))

1.1.212 MakeCompatibleBitmap Function

This is function MakeCompatibleBitmap.

C++

```
HBITMAP WINAPI MakeCompatibleBitmap(HBITMAP source, int width, int height);
```

File

Graphics.h ([see page 155](#))

1.1.213 MakeSoundFromFileA Function

Plays the wave sound from the file

C++

```
void WINAPI MakeSoundFromFileA(LPCSTR soundName);
```

File

System.h ([see page 161](#))

Parameters

Parameters	Description
LPCSTR soundName	The name of the file

Description

This function plays a sound specified by the given file name.

1.1.214 MakeSoundFromFileW Function

Plays the wave sound from the file

C++

```
void WINAPI MakeSoundFromFileW(LPCWSTR soundName);
```

File

System.h ([see page 161](#))

Parameters

Parameters	Description
LPCWSTR soundName	The name of the file

Description

This function plays a sound specified by the given file name.

1.1.215 MakeSoundFromResourceA Function

Plays the wave sound from the resource

C++

```
void WINAPI MakeSoundFromResourceA(HMODULE hModule, LPCSTR soundName);
```

File

System.h (🔗 see page 161)

Parameters

Parameters	Description
HMODULE hModule	Handle to the executable file that contains the resource to be loaded.
LPCSTR soundName	A string that specifies the sound to play.

Description

This function plays a sound specified by the given resource name.

1.1.216 MakeSoundFromResourceW Function

Plays the wave sound from the resource

C++

```
void WINAPI MakeSoundFromResourceW(HMODULE hModule, LPCWSTR soundName);
```

File

System.h (🔗 see page 161)

Parameters

Parameters	Description
HMODULE hModule	Handle to the executable file that contains the resource to be loaded.
LPCWSTR soundName	A string that specifies the sound to play.

Description

This function plays a sound specified by the given resource name.

1.1.217 PointsToPixels Function

This is function PointsToPixels.

C++

```
int WINAPI PointsToPixels(int points);
```

File

Graphics.h (🔗 see page 155)

1.1.218 PortAvailable Function

Checks if the port with specified number is available

C++

```
BOOL WINAPI PortAvailable(int port);
```

File

SystemInfo.h (see page 162)

1.1.219 ReadAddressA Function

Read address information from Belgian eID card

C++

```
BOOL WINAPI ReadAddressA(PEidAddressA address);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PEidAddressA address	The pointer to the address information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.220 ReadAddressExA Function

Read address information from Belgian eID card

C++

```
BOOL WINAPI ReadAddressExA(int readerNumber, PEidAddressA address);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PEidAddressA address	The pointer to the address information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.221 ReadAddressExW Function

Read address information from Belgian eID card

C++

```
BOOL WINAPI ReadAddressExW(int readerNumber, PEidAddressW address);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PEidAddressW address	The pointer to the address information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.222 ReadAddressW Function

Read address information from Belgian eID card

C++

```
BOOL WINAPI ReadAddressW(PEidAddressW address);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PEidAddressW address	the pointer to the address information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.223 ReadAuthenticationCertificate Function

Read Authentication Certificate to memory

C++

```
BOOL WINAPI ReadAuthenticationCertificate(PEidCertificate certificate);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
certificate	The pointer to EidCertificate (see page 130) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Authentication Certificate from the card to EidCertificate (see page 130) structure

1.1.224 ReadAuthenticationCertificateEx Function

Read Authentication Certificate to memory

C++

```
bool WINAPI ReadAuthenticationCertificateEx(int readerNumber, PEidCertificate certificate);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
certificate	The pointer to EidCertificate (see page 130) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Authentication Certificate from the card to EidCertificate (see page 130) structure

1.1.225 ReadBufferFromFileA Function

Reads the content of the file to the memory buffer

C++

```
void WINAPI ReadBufferFromFileA(LPSTR fileName, BYTE* buffer, int bufferSize);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
BYTE* buffer	The address of the memory block
int bufferSize	The size of the memory block

Description

Use this function to retrieve the content of the file to the memory block

1.1.226 ReadBufferFromFileW Function

Reads the content of the file to the memory buffer

C++

```
void WINAPI ReadBufferFromFileW(LPWSTR fileName, BYTE* buffer, int bufferSize);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
BYTE* buffer	The address of the memory block
int bufferSize	The size of the memory block

Description

Use this function to retrieve the content of the file to the memory block

1.1.227 ReadCaCertificate Function

Read Ca Certificate to memory

C++

```
BOOL WINAPI ReadCaCertificate(PEidCertificate certificate);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
certificate	The pointer to EidCertificate (see page 130) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Ca Certificate to EidCertificate (see page 130) structure

1.1.228 ReadCaCertificateEx Function

Read Ca Certificate to memory

C++

```
BOOL WINAPI ReadCaCertificateEx(int readerNumber, PEidCertificate certificate);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
certificate	The pointer to EidCertificate (see page 130) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Ca Certificate to EidCertificate (see page 130) structure

1.1.229 ReadIdentityA Function

Read identity information from Belgian eID card

C++

```
BOOL WINAPI ReadIdentityA(PEidIdentityA identity);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PEidIdentityA identity	The pointer to the identity information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.230 ReadIdentityExA Function

Read identity information from Belgian eID card

C++

```
BOOL WINAPI ReadIdentityExA(int readerNumber, PEidIdentityA identity);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PEidIdentityA identity	The pointer to the identity information structure

Returns

Returns TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.231 ReadIdentityExW Function

Read identity information from Belgian eID card

C++

```
BOOL WINAPI ReadIdentityExW(int readerNumber, PEidIdentityW identity);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PEidIdentityW identity	The pointer to the identity information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.232 ReadIdentityW Function

Read identity information from Belgian eID card

C++

```
BOOL WINAPI ReadIdentityW(PEidIdentityW identity);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
PEidIdentityW identity	The pointer to the identity information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.233 ReadNonRepudiationCertificate Function

Read Non Repudiation Certificate to memory

C++

```
BOOL WINAPI ReadNonRepudiationCertificate(PEidCertificate certificate);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
certificate	The pointer to EidCertificate (🔗 see page 130) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Non Repudiation Certificate to EidCertificate (🔗 see page 130) structure

1.1.234 ReadNonRepudiationCertificateEx Function

Read Non Repudiation Certificate to memory

C++

```
BOOL WINAPI ReadNonRepudiationCertificateEx(int readerNumber, PEidCertificate certificate);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
certificate	The pointer to EidCertificate (see page 130) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Non Repudiation Certificate to EidCertificate (see page 130) structure

1.1.235 ReadPhoto Function

Reads a photo from a card

C++

```
BOOL WINAPI ReadPhoto(PEidPicture photo);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PEidPicture photo	The pointer to EidPicture (see page 132) structure

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

Description

Reads a photo from Belgian eID card to EidPicture (see page 132) structure. This structure holds the raw image bytes and the length of the image bytes array

1.1.236 ReadPhotoAsBitmap Function

Reads the picture from the card, converts it to bitmap and returns the bitmap handle
Description: Reads the photo from the Belgian eID card and returns the bitmap handle
Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
HBITMAP WINAPI ReadPhotoAsBitmap();
```

File

Swelio.h (see page 156)

Returns

A handle to a bitmap indicates success. NULL indicates failure.

1.1.237 ReadPhotoAsBitmapEx Function

Reads the picture from the card, converts it to bitmap and returns the bitmap handle
Description: Reads the photo from the Belgian eID card and returns the Windows bitmap handle
Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
HBITMAP WINAPI ReadPhotoAsBitmapEx(int readerNumber);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.

Returns

A handle to a bitmap indicates success. NULL indicates failure.

1.1.238 ReadPhotoEx Function

Reads a photo from a card

C++

```
BOOL WINAPI ReadPhotoEx(int readerNumber, PeidPicture photo);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PeidPicture photo	The pointer to EidPicture (see page 132) structure

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

Description

Reads a photo from Belgian eID card to EidPicture (see page 132) structure

1.1.239 ReadRootCaCertificate Function

Read Root Ca Certificate to memory

C++

```
BOOL WINAPI ReadRootCaCertificate(PEidCertificate certificate);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
certificate	The pointer to EidCertificate (see page 130) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Root Ca Certificate to EidCertificate (see page 130) structure

1.1.240 ReadRootCaCertificateEx Function

Read Root Ca Certificate to memory

C++

```
BOOL WINAPI ReadRootCaCertificateEx(int readerNumber, PEidCertificate certificate);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
certificate	The pointer to EidCertificate (see page 130) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Root Ca Certificate to EidCertificate (see page 130) structure

1.1.241 ReadRrnCertificate Function

Read Rrn Certificate to memory

C++

```
BOOL WINAPI ReadRrnCertificate(PEidCertificate certificate);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
certificate	The pointer to EidCertificate (see page 130) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Rrn Certificate to EidCertificate (see page 130) structure

1.1.242 ReadRrnCertificateEx Function

Read Rrn Certificate to memory

C++

```
BOOL WINAPI ReadRrnCertificateEx(int readerNumber, PEidCertificate certificate);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
certificate	The pointer to EidCertificate (see page 130) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Rrn Certificate to EidCertificate (see page 130) structure

1.1.243 ReadSISCardA Function

Read Belgian SIS card.

C++

```
BOOL WINAPI ReadSISCardA(PSISRecordA identity);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PSISRecordA	The pointer to SISRecordA (see page 137) structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

Description

Read the public information from the Belgian SIS card. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

1.1.244 ReadSISCardExA Function

Read Belgian SIS card.

C++

```
BOOL WINAPI ReadSISCardExA(int readerNumber, PSISRecordA identity);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PSISRecordA	The pointer to SISRecordA (see page 137) structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

Description

Read the public information from the Belgian SIS card. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

1.1.245 ReadSISCardExW Function

Read Belgian SIS card.

C++

```
BOOL WINAPI ReadSISCardExW(int readerNumber, PSISRecordW identity);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PSISRecordW	The pointer to SISRecordW (see page 138) structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

Description

Read the public information from the Belgian SIS card. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

1.1.246 ReadSISCardW Function

Read Belgian SIS card.

C++

```
BOOL WINAPI ReadSISCardW(P SISRecordW identity);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
P SISRecordW	The pointer to SISRecordW (see page 138) structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

Description

Read the public information from the Belgian SIS card. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

1.1.247 RecycleBinEmpty Function

Returns TRUE if Windows Recycle Bin is empty

C++

```
BOOL WINAPI RecycleBinEmpty();
```

File

SystemInfo.h (see page 162)

1.1.248 ReloadReadersList Function

Reloads the list of the available card readers

C++

```
void WINAPI ReloadReadersList();
```

File

Swelio.h (see page 156)

Description

When the card reader is inserted or removed you may need to reload the list of the available card readers

1.1.249 RemoveCallback Function

Remove callback procedure for card events

C++

```
void WINAPI RemoveCallback();
```

File

Swelio.h (see page 156)

Description

Use this function to deactivate card events callback procedure

1.1.250 RemoveStartupA Function

Removes the application from the list of the automatically started applications

C++

```
void WINAPI RemoveStartupA(LPCSTR appName);
```

File

System.h ([see page 161](#))

Parameters

Parameters	Description
LPCSTR appName	The name of the application

Description

For application that starts automatically when Windows starts removes it from the automatically launching applications list

1.1.251 RemoveStartupW Function

Removes the application from the list of the automatically started applications

C++

```
void WINAPI RemoveStartupW(LPCWSTR appName);
```

File

System.h ([see page 161](#))

Parameters

Parameters	Description
LPCWSTR appName	The name of the application

Description

For application that starts automatically when Windows starts removes it from the automatically launching applications list

1.1.252 RestoreWindowSubclassA Function

Restores window standard procedure

C++

```
void WINAPI RestoreWindowSubclassA(HWND hwnd);
```

File

System.h ([see page 161](#))

Parameters

Parameters	Description
HWND hwnd	The window handle

1.1.253 RestoreWindowSubclassW Function

Restores window standard procedure

C++

```
void WINAPI RestoreWindowSubclassW(HWND hwnd);
```

File

System.h (🔗 see page 161)

Parameters

Parameters	Description
HWND hwnd	The window handle

1.1.254 SaveAuthenticationCertificateA Function

Save Authentication Certificate to a file

C++

```
void WINAPI SaveAuthenticationCertificateA(LPSTR fileName);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
LPSTR fileName	File name to store the certificate

Description

Read Authentication Certificate from the card and save it to a file.

1.1.255 SaveAuthenticationCertificateExW Function

Save Authentication Certificate to a file

C++

```
void WINAPI SaveAuthenticationCertificateExW(LPWSTR fileName, int readerNumber);
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate
int readerNumber	The reader index, starting from 0

Description

Read Authentication Certificate from the card and save it to a file.

1.1.256 SaveAuthenticationCertificateW Function

Save Authentication Certificate to a file

C++

```
void WINAPI SaveAuthenticationCertificateW(LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate

Description

Read Authentication Certificate from the card and save it to a file.

1.1.257 SaveCaCertificateA Function

Save Ca Certificate to a file

C++

```
void WINAPI SaveCaCertificateA(LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR fileName	File name to store the certificate

Description

Read Ca Certificate from the card and save it to a file

1.1.258 SaveCaCertificateExW Function

Save Ca Certificate to a file

C++

```
void WINAPI SaveCaCertificateExW(LPWSTR fileName, int readerNumber);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate
int readerNumber	The reader index, starting from 0

Description

Read Ca Certificate from the card and save it to a file

1.1.259 SaveCaCertificateW Function

Save Ca Certificate to a file

C++

```
void WINAPI SaveCaCertificateW(LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate

Description

Read Ca Certificate from the card and save it to a file

1.1.260 SaveCardToToXMLStreamExA Function

Read eID card and save the information to XML buffer

C++

```
BOOL WINAPI SaveCardToToXMLStreamExA(int readerNumber, void* buffer);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
void* buffer	The memory buffer to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to XML buffer in the memory.

1.1.261 SaveCardToToXMLStreamExW Function

Read eID card and save the information to XML buffer

C++

```
BOOL WINAPI SaveCardToToXMLStreamExW(int readerNumber, void* buffer);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
void* buffer	The memory buffer to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to XML buffer in the memory.

1.1.262 SaveCardToXmlA Function

Read eID card and save the information to XML file

C++

```
BOOL WINAPI SaveCardToXmlA(LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to XML file.

1.1.263 SaveCardToXmlExA Function

Read eID card and save the information to XML file

C++

```
BOOL WINAPI SaveCardToXmlExA(int readerNumber, LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to XML file.

1.1.264 SaveCardToXmlExW Function

Read eID card and save the information to XML file

C++

```
BOOL WINAPI SaveCardToXmlExW(int readerNumber, LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to XML file.

1.1.265 SaveCardToXmlW Function

Read eID card and save the information to XML file

C++

```
BOOL WINAPI SaveCardToXmlW(LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to XML file.

1.1.266 SaveIdentityA Function

Saves indentity information to a file

C++

```
void WINAPI SaveIdentityA(LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR fileName	The name of the destination file

Description

Use this function to store the raw identity information from the Belgian eID card to a file. You can use LoadIdentityA (see page 81) to read this information from the file to EidIdentityA (see page 130) structure

1.1.267 SaveIdentityW Function

Saves indentity information to a file

C++

```
void WINAPI SaveIdentityW(LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	The name of the destination file

Description

Use this function to store the raw identity information from the Belgian eID card to a file. You can use LoadIdentityW (see page 81) to read this information from the file to EidIdentityW (see page 131) structure

1.1.268 SaveNonRepudiationCertificateA Function

Save Non Repudiation Certificate to a file

C++

```
void WINAPI SaveNonRepudiationCertificateA(LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR fileName	File name to store the certificate

Description

Read Non Repudiation Certificate from the card and save it to a file

1.1.269 SaveNonRepudiationCertificateExW Function

Save Non Repudiation Certificate to a file

C++

```
void WINAPI SaveNonRepudiationCertificateExW(LPWSTR fileName, int readerNumber);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate
int readerNumber	The reader index, starting from 0

Description

Read Non Repudiation Certificate from the card and save it to a file

1.1.270 SaveNonRepudiationCertificateW Function

Save Non Repudiation Certificate to a file

C++

```
void WINAPI SaveNonRepudiationCertificateW(LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate

Description

Read Non Repudiation Certificate from the card and save it to a file

1.1.271 SavePersonCsvToStreamA Function

This is function SavePersonCsvToStreamA.

C++

```
BOOL WINAPI SavePersonCsvToStreamA(int readerNumber, void* buffer);
```

File

Swelio.h (see page 156)

1.1.272 SavePersonCsvToStreamW Function

This is function SavePersonCsvToStreamW.

C++

```
BOOL WINAPI SavePersonCsvToStreamW(int readerNumber, void* buffer);
```

File

Swelio.h (see page 156)

1.1.273 SavePersonToCsvA Function

Read eID card and save the identity information and address to CSV file

C++

```
BOOL WINAPI SavePersonToCsvA(LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to CSV file.

1.1.274 SavePersonToCsvExA Function

Read eID card and save the identity information and address to CSV file

C++

```
BOOL WINAPI SavePersonToCsvExA(int readerNumber, LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to CSV file.

1.1.275 SavePersonToCsvExW Function

Read eID card and save the identity information and address to CSV file

C++

```
BOOL WINAPI SavePersonToCsvExW(int readerNumber, LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to CSV file.

1.1.276 SavePersonToCsvW Function

Read eID card and save the identity information and address to CSV file

C++

```
BOOL WINAPI SavePersonToCsvW(LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to CSV file.

1.1.277 SavePhotoA Function

Save photo to a file

C++

```
void WINAPI SavePhotoA(PeidPicture photo, LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PeidPicture photo	The pointer to EidPicture (see page 132) structure
LPSTR fileName	Destination file name

Description

Save the raw picture data to a file

1.1.278 SavePhotoAsBitmapA Function

Save the picture from the card to Windows Bitmap file
Description: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsBitmapA(LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.279 SavePhotoAsBitmapExA Function

Reads the picture from the card and saves it to Windows Bitmap file
Description: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsBitmapExA(int readerNumber, LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.280 SavePhotoAsBitmapExW Function

Reads the picture from the card and saves it to Windows Bitmap file Description: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsBitmapExW(int readerNumber, LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	the zero-based index of the card reader.
LPWSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.281 SavePhotoAsBitmapW Function

Save the picture from the card to Windows Bitmap file Description: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsBitmapW(LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.282 SavePhotoAsJpegA Function

Save the picture from the card to JPG file Description: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsJpegA(LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.283 SavePhotoAsJpegExA Function

Save the picture from the card to JPG file Description: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsJpegExA(int readerNumber, LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	the zero-based index of the card reader.
LPSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.284 SavePhotoAsJpegExW Function

Save the picture from the card to JPG file Description: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsJpegExW(int readerNumber, LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.285 SavePhotoAsJpegW Function

Save the picture from the card to JPG file Description: Reads the photo from the Belgian eID card and writes it to the file as

JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsJpegW(LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.286 SavePhotoW Function

Saves photo to a file

C++

```
void WINAPI SavePhotoW(PeidPicture photo, LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PeidPicture photo	The pointer to EidPicture (see page 132) structure
LPWSTR fileName	Destination file name

Description

Saves the raw picture data to a file

1.1.287 SaveRootCaCertificateA Function

Save Root Ca Certificate to a file

C++

```
void WINAPI SaveRootCaCertificateA(LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR fileName	File name to store the certificate

Description

Read Root CA certificate from the card and save it to a file

1.1.288 SaveRootCaCertificateExW Function

Save Root Ca Certificate to a file

C++

```
void WINAPI SaveRootCaCertificateExW(LPWSTR fileName, int readerNumber);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate
int readerNumber	The reader index, starting from 0

Description

Read Root CA certificate from the card and save it to a file

1.1.289 SaveRootCaCertificateW Function

Save Root Ca Certificate to a file

C++

```
void WINAPI SaveRootCaCertificateW(LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate

Description

Read Root CA certificate from the card and save it to a file

1.1.290 SaveRrnCertificateA Function

Save RRN Certificate to a file

C++

```
void WINAPI SaveRrnCertificateA(LPSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR fileName	File name to store the certificate

Description

Read RRN certificate from the card and save it to a file

1.1.291 SaveRrnCertificateExW Function

Save RRN Certificate to a file

C++

```
void WINAPI SaveRrnCertificateExW(LPWSTR fileName, int readerNumber);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate
int readerNumber	The reader index, starting from 0

Description

Read RRN certificate from the card and save it to a file

1.1.292 SaveRrnCertificateW Function

Save RRN Certificate to a file

C++

```
void WINAPI SaveRrnCertificateW(LPWSTR fileName);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate

Description

Read RRN certificate from the card and save it to a file

1.1.293 SelectReader Function

When more than 1 reader connected, select the reader with specified number

C++

```
BOOL WINAPI SelectReader(int readerNumber);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
int readerNumber	The reader index, starting from 0

Returns

TRUE if the reader is selected, FALSE if the reader with specified number does not exist

Description

Selects the default card reader using the zero-based reader index. The first reader has number 0, second - 1, etc... You can read the information only from one selected reader at once.

1.1.294 SelectReaderByNameA Function

Select active smart card reader by providing the reader name

C++

```
BOOL WINAPI SelectReaderByNameA(LPSTR readerName) ;
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
LPSTR readerName	The name of the card reader

Returns

TRUE if the reader is selected. If the reader with specified name is not found - returns FALSE

Description

Activates the reader with specified name

1.1.295 SelectReaderByNameW Function

Select active smart card reader by providing the reader name

C++

```
BOOL WINAPI SelectReaderByNameW(LPWSTR readerName) ;
```

File

Swelio.h (🔗 see page 156)

Parameters

Parameters	Description
LPWSTR readerName	The name of the card reader

Returns

Returns TRUE if the reader is selected. If the reader with specified name is not found - returns FALSE

Description

Activates the reader with specified name

1.1.296 SendAPDU Function

This is function SendAPDU.

C++

```
BOOL WINAPI SendAPDU(int readerNumber, LPCBYTE apdu, DWORD apduLen, PCHAR result, LPDWORD len);
```

File

Swelio.h (see page 156)

1.1.297 SetCallback Function

Activates callback procedure for card status change event

C++

```
void WINAPI SetCallback(CALLBACK_HANDLER callback, LPVOID userContext);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
CALLBACK_HANDLER callback	The pointer to callback procedure
LPVOID userContext	The user defined value passed to the callback procedure

Description

Your application can be notified about insertion or removal of the card from the card reader and the changes of the available card readers list (the reader is connected or disconnected from PC) Use this function to install the callback procedure

1.1.298 SetMWCompatibility Function

Set the compatibility mode with the old version of the official EID MiddleWare

C++

```
void WINAPI SetMWCompatibility();
```

File

Swelio.h (see page 156)

Description

The compatibility mode can be useful when the MiddleWare version 1.x or 2.x is installed on the target PC. Usually the more recent MiddleWare is used and this function is provided for backward compatibility only

1.1.299 SetStartupA Function

Register application to run when Windows starts

C++

```
void WINAPI SetStartupA(LPCSTR appName, LPCWSTR appPath);
```

File

System.h (see page 161)

Parameters

Parameters	Description
LPCSTR appName	The name of the application
LPCWSTR appPath	The path to the application executable

1.1.300 SetStartupW Function

Register application to run when Windows starts

C++

```
void WINAPI SetStartupW(LPCWSTR appName, LPCWSTR appPath);
```

File

System.h (see page 161)

Parameters

Parameters	Description
LPCWSTR appName	The name of the application
LPCWSTR appPath	The path to the application executable

1.1.301 SetSupportSIS Function

Activates or deactivates SIS card support by engine

C++

```
void WINAPI SetSupportSIS(BOOL value);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
BOOL value	The SIS card support status

Description

Use SetSupportSIS to activate or deactivate the SIS card detection and reading. Even if SIS card support is activated it can be used only with ACR38U card readers Other card readers are not supported.

1.1.302 ShellCopyFileA Function

Copies file to the new location

C++

```
void WINAPI ShellCopyFileA(LPSTR oldName, LPSTR newName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPSTR oldName	The source file name
LPSTR newName	The destination file name

Description

Copies file to the new location using Windows shell copy routine

1.1.303 ShellCopyFileW Function

Copies file to the new location

C++

```
void WINAPI ShellCopyFileW(LPWSTR oldName, LPWSTR newName);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPWSTR oldName	The source file name
LPWSTR newName	The destination file name

Description

Copies file to the new location using Windows shell copy routine

1.1.304 ShutdownWindows Function

Logs off the interactive user, shuts down the system.

C++

```
BOOL WINAPI ShutdownWindows(UINT flags);
```

File

System.h (see page 161)

Returns

If the function succeeds returns TRUE, otherwise returns FALSE

Description

Logs off the interactive user, shuts down the system, or shuts down and restarts the system. It sends the WM_QUERYENDSESSION message to all applications to determine if they can be terminated.

This function accepts the following parameter:

flags : The shutdown type. This parameter must include one of the following values:

Value	Meaning
EWX_LOGOFF	Shuts down all processes running in the logon session of the process that called the ExitWindowsEx function. Then it logs the user off.
EWX_POWEROFF	Shuts down the system and turns off the power. The system must support the power-off feature.
EWX_REBOOT	Shuts down the system and then restarts the system.
EWX_RESTARTAPPS	Shuts down the system and then restarts it
EWX_SHUTDOWN	Shuts down the system to a point at which it is safe to turn off the power.

1.1.305 StartEngine Function

Activates the Swelio Engine.

C++

```
BOOL WINAPI StartEngine();
```

File

Swelio.h (see page 156)

Returns

Returns TRUE if the Swelio Engine is successfully started; otherwise returns FALSE

Description

This procedure must be called first before any other functions from Swelio library can be used.

1.1.306 StopEngine Function

Deactivates the Swelio Engine

C++

```
void WINAPI StopEngine();
```

File

Swelio.h (see page 156)

Description

Deactivates the Swelio Engine and clean up the used memory. Call this procedure at the end of you application once to finalize the usage of the Swelio Engine.

1.1.307 StretchNativeBitmap Function

This is function StretchNativeBitmap.

C++

```
BOOL WINAPI StretchNativeBitmap(HBITMAP src, HBITMAP dst, int srcWidth, int srcHeight, int dstWidth, int dstHeight);
```


File

Graphics.h (🔗 see page 155)

1.1.308 StripFileNameA Function

Replaces environment variable names with values

C++

```
void WINAPI StripFileNameA(LPCSTR fileName, LPSTR fullName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPCSTR fileName	The source file name
LPSTR fullName	The expanded file name

Description

This function expands environment-variable strings and replaces them with their defined values in the file name.

1.1.309 StripFileNameW Function

Replaces environment variable names with values

C++

```
void WINAPI StripFileNameW(LPCWSTR fileName, LPWSTR fullName);
```

File

FileOperations.h (🔗 see page 153)

Parameters

Parameters	Description
LPCWSTR fileName	The source file name
LPWSTR fullName	The expanded file name

Description

This function expands environment-variable strings and replaces them with their defined values in the file name.

1.1.310 SuspendWindows Function

Suspends Windows

C++

```
BOOL WINAPI SuspendWindows();
```

File

System.h (🔗 see page 161)

1.1.311 TurnMonitorOff Function

Turns the monitor off

C++

```
void WINAPI TurnMonitorOff();
```

File

System.h (see page 161)

1.1.312 TurnMonitorOn Function

Turns the monitor on

C++

```
void WINAPI TurnMonitorOn();
```

File

System.h (see page 161)

1.1.313 UpdateWindowPosition Function

Updated the window position

C++

```
void WINAPI UpdateWindowPosition(HWND handle, int x, int y);
```

File

System.h (see page 161)

Parameters

Parameters	Description
HWND handle	The handle of the window
int x	New horizontal coordinate
int y	New vertical coordinate

1.1.314 VerifyPinA Function

Verify PIN code

C++

```
BOOL WINAPI VerifyPinA(LPSTR value);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPSTR value	PIN code to verify

Returns

TRUE when the correct PIN code is provided; otherwise returns FALSE

1.1.315 VerifyPinExA Function

Verify PIN code

C++

```
BOOL WINAPI VerifyPinExA(int readerNumber, LPSTR value);
```

File

Swelio.h ([see page 156](#))

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR value	PIN code to verify

Returns

TRUE when the correct PIN code is provided; otherwise returns FALSE

1.1.316 VerifyPinExW Function

Verify PIN code

C++

```
BOOL WINAPI VerifyPinExW(int readerNumber, LPWSTR value);
```

File

Swelio.h ([see page 156](#))

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR value	PIN code to verify

Returns

TRUE when the correct PIN code is provided; otherwise returns FALSE

1.1.317 VerifyPinW Function

Verify PIN code

C++

```
BOOL WINAPI VerifyPinW(LPWSTR value);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
LPWSTR value	PIN code to verify

Returns

TRUE when the correct PIN code is provided; otherwise returns FALSE

1.1.318 VerifySignature Function

Verifies the signature from the specified hash value.

C++

```
BOOL WINAPI VerifySignature(PEidCertificate certificate, BYTE* buffer, int bufferSize,
BYTE* signature, DWORD signatureSize);
```

File

Swelio.h (see page 156)

Parameters

Parameters	Description
PEidCertificate certificate	The public certificate
BYTE* buffer	The hash buffer
int bufferSize	The size of the hash buffer
BYTE* signature	The signature to be verified.
DWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is valid for the hash; otherwise, FALSE.

Description

Verify the signature using the public certificate of the signer

1.1.319 WriteBufferToFileA Function

Writes the memory buffer to file

C++

```
void WINAPI WriteBufferToFileA(LPSTR fileName, BYTE* buffer, int bufferSize);
```

File

FileOperations.h (see page 153)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
BYTE* buffer	The address of the memory block
int bufferSize	The size of the memory block

Description

This function stores the content of the memory buffer to the file.

1.1.320 WriteBufferToFileW Function

Writes the memory buffer to file

C++

```
void WINAPI WriteBufferToFileW(LPWSTR fileName, BYTE* buffer, int bufferSize);
```

File

FileOperations.h ([see page 153](#))

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
BYTE* buffer	The address of the memory block
int bufferSize	The size of the memory block


Description

This function stores the content of the memory buffer to the file.









1.2 Structs, Records, Enums

The following table lists structs, records, enums in this documentation.

Enumerations

	Name	Description
	tagCardEventType (see page 123)	The type of the reader event
	CardEventType (see page 128)	The type of the reader event

Structures

	Name	Description
	tagEidAddressA (see page 123)	EID address information, stored on the card - ANSI version
	tagEidAddressW (see page 124)	EID address information, stored on the card - UNICODE version
	tagEidCertificate (see page 124)	Certificate, stored on EID card
	tagEidIdentityA (see page 125)	Identity information stored on EID card - ANSI version
	tagEidIdentityW (see page 126)	Identity information stored on EID card - UNICODE version
	tagEidPicture (see page 127)	Raw picture data from EID card
	tagSISRecordA (see page 127)	Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	tagSISRecordW (see page 128)	Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	EidAddressA (see page 129)	EID address information, stored on the card - ANSI version
	EidAddressW (see page 129)	EID address information, stored on the card - UNICODE version

	EidCertificate (see page 130)	Certificate, stored on EID card
	EidIdentityA (see page 130)	Identity information stored on EID card - ANSI version
	EidIdentityW (see page 131)	Identity information stored on EID card - UNICODE version
	EidPicture (see page 132)	Raw picture data from EID card
	PEidAddressA (see page 132)	EID address information, stored on the card - ANSI version
	PEidAddressW (see page 133)	EID address information, stored on the card - UNICODE version
	PEidCertificate (see page 133)	Certificate, stored on EID card
	PEidIdentityA (see page 133)	Identity information stored on EID card - ANSI version
	PEidIdentityW (see page 134)	Identity information stored on EID card - UNICODE version
	PeidPicture (see page 135)	Raw picture data from EID card
	PSISRecordA (see page 136)	Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	PSISRecordW (see page 136)	Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	SISRecordA (see page 137)	Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	SISRecordW (see page 138)	Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

1.2.1 tagCardEventType Enumeration

The type of the reader event

C++

```
enum tagCardEventType {
    ewtUnknownEvent,
    ewtCardInsert,
    ewtCardRemove,
    ewtReadersChange
};
```

File

CardEvents.h (see page 150)

Members

Members	Description
ewtUnknownEvent	Unknown event
ewtCardInsert	The card was inserted in the reader
ewtCardRemove	The card was removed from the reader
ewtReadersChange	The readers list changed

1.2.2 tagEidAddressA Structure

EID address information, stored on the card - ANSI version

C++

```
struct tagEidAddressA {
    char street[EID_MAX_STREET_LEN+1];
```

```
char zip[EID_MAX_ZIP_LEN+1];  
char municipality[EID_MAX_MUNICIPALITY_LEN+1];  
};
```

File

CardStructures.h (see page 150)

Members

Members	Description
char street[EID_MAX_STREET_LEN+1];	Street name
char zip[EID_MAX_ZIP_LEN+1];	ZIP code
char municipality[EID_MAX_MUNICIPALITY_LEN+1];	Municipality

1.2.3 tagEidAddressW Structure

EID address information, stored on the card - UNICODE version

C++

```
struct tagEidAddressW {  
    WCHAR street[EID_MAX_STREET_LEN+1];  
    WCHAR zip[EID_MAX_ZIP_LEN+1];  
    WCHAR municipality[EID_MAX_MUNICIPALITY_LEN+1];  
};
```

File

CardStructures.h (see page 150)

Members

Members	Description
WCHAR street[EID_MAX_STREET_LEN+1];	Street name
WCHAR zip[EID_MAX_ZIP_LEN+1];	ZIP code
WCHAR municipality[EID_MAX_MUNICIPALITY_LEN+1];	Municipality

1.2.4 tagEidCertificate Structure

Certificate, stored on EID card

C++

```
struct tagEidCertificate {  
    BYTE certificate[EID_MAX_CERT_LEN+1];  
    int certificateLength;  
};
```

File

CardStructures.h (see page 150)

Members

Members	Description
BYTE certificate[EID_MAX_CERT_LEN+1];	Certificate raw data buffer
int certificateLength;	Certificate data length

1.2.5 tagEidIdentityA Structure

Identity information stored on EID card - ANSI version

C++

```
struct tagEidIdentityA {
    char cardNumber[EID_MAX_CARD_NUMBER_LEN+1];
    char chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];
    char validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];
    char validityDateEnd[EID_MAX_DATE_END_LEN +1];
    char municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];
    char nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];
    char name[EID_MAX_NAME_LEN+1];
    char firstName1[EID_MAX_FIRST_NAME1_LEN+1];
    char firstName2[EID_MAX_FIRST_NAME2_LEN+1];
    char nationality[EID_MAX_NATIONALITY_LEN+1];
    char birthLocation[EID_MAX_BIRTHPLACE_LEN+1];
    char birthDate[EID_MAX_BIRTHDATE_LEN+1];
    char sex[EID_MAX_SEX_LEN+1];
    char nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];
    int documentType;
    BOOL whiteCane;
    BOOL yellowCane;
    BOOL extendedMinority;
};
```

File

CardStructures.h (see page 150)

Members

Members	Description
char cardNumber[EID_MAX_CARD_NUMBER_LEN+1];	Electronic ID card number
char chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];	Electronic ID card physical chip number
char validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];	Card validity start date
char validityDateEnd[EID_MAX_DATE_END_LEN +1];	Card validity end date
char municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];	Card delivery municipality
char nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];	National number
char name[EID_MAX_NAME_LEN+1];	Surname
char firstName1[EID_MAX_FIRST_NAME1_LEN+1];	First name (2 first given names)
char firstName2[EID_MAX_FIRST_NAME2_LEN+1];	First name part 2 (first letter of the 3rd given name).
char nationality[EID_MAX_NATIONALITY_LEN+1];	Nationality
char birthLocation[EID_MAX_BIRTHPLACE_LEN+1];	Birth location
char birthDate[EID_MAX_BIRTHDATE_LEN+1];	Birth date
char sex[EID_MAX_SEX_LEN+1];	Sex
char nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];	Noble condition
int documentType;	Document type code (Belgian citizen card, Kids Card, Foreigner card)
BOOL whiteCane;	White cane (blind people)
BOOL yellowCane;	Yellow cane (partially sighted people)
BOOL extendedMinority;	Extended minority

1.2.6 tagEidIdentityW Structure

Identity information stored on EID card - UNICODE version

C++

```
struct tagEidIdentityW {
    WCHAR cardNumber[EID_MAX_CARD_NUMBER_LEN+1];
    WCHAR chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];
    WCHAR validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];
    WCHAR validityDateEnd[EID_MAX_DATE_END_LEN +1];
    WCHAR municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];
    WCHAR nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];
    WCHAR name[EID_MAX_NAME_LEN+1];
    WCHAR firstName1[EID_MAX_FIRST_NAME1_LEN+1];
    WCHAR firstName2[EID_MAX_FIRST_NAME2_LEN+1];
    WCHAR nationality[EID_MAX_NATIONALITY_LEN+1];
    WCHAR birthLocation[EID_MAX_BIRTHPLACE_LEN+1];
    WCHAR birthDate[EID_MAX_BIRTHDATE_LEN+1];
    WCHAR sex[EID_MAX_SEX_LEN+1];
    WCHAR nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];
    int documentType;
    BOOL whiteCane;
    BOOL yellowCane;
    BOOL extendedMinority;
};
```

File

CardStructures.h (see page 150)

Members

Members	Description
WCHAR cardNumber[EID_MAX_CARD_NUMBER_LEN+1];	Electronic ID card number
WCHAR chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];	Electronic ID card physical chip number
WCHAR validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];	Card validity start date
WCHAR validityDateEnd[EID_MAX_DATE_END_LEN +1];	Card validity end date
WCHAR municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];	Card delivery municipality
WCHAR nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];	National number
WCHAR name[EID_MAX_NAME_LEN+1];	Surname
WCHAR firstName1[EID_MAX_FIRST_NAME1_LEN+1];	First name (2 first given names)
WCHAR firstName2[EID_MAX_FIRST_NAME2_LEN+1];	First name part 2 (first letter of the 3rd given name).
WCHAR nationality[EID_MAX_NATIONALITY_LEN+1];	Nationality
WCHAR birthLocation[EID_MAX_BIRTHPLACE_LEN+1];	Birth location
WCHAR birthDate[EID_MAX_BIRTHDATE_LEN+1];	Birth date
WCHAR sex[EID_MAX_SEX_LEN+1];	Sex
WCHAR nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];	Noble condition
int documentType;	Document type code (Belgian citizen card, Kids Card, Foreigner card)
BOOL whiteCane;	White cane (blind people)
BOOL yellowCane;	Yellow cane (partially sighted people)
BOOL extendedMinority;	Extended minority

1.2.7 tagEidPicture Structure

Raw picture data from EID card

C++

```
struct tagEidPicture {
    BYTE picture[EID_MAX_PICTURE_LEN+1];
    int pictureLength;
};
```

File

CardStructures.h (see page 150)

Members

Members	Description
BYTE picture[EID_MAX_PICTURE_LEN+1];	Picture raw data buffer
int pictureLength;	Picture raw data buffer length

1.2.8 tagSISRecordA Structure

Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

C++

```
struct tagSISRecordA {
    char Name[SIS_MAX_NAME_LEN + 1];
    char FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];
    char Initial[SIS_MAX_INITIAL_LEN+ 1];
    char Sex[SIS_MAX_SEX_LEN + 1];
    char BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];
    char SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];
    char CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];
    char ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];
    char ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];
    char CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];
    char CardName[SIS_MAX_CARDNAME_LEN +1 ];
};
```

File

CardStructures.h (see page 150)

Members

Members	Description
char Name[SIS_MAX_NAME_LEN + 1];	Family name of the card owner
char FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];	First name of the card owner
char Initial[SIS_MAX_INITIAL_LEN+ 1];	Initial of the card owner
char Sex[SIS_MAX_SEX_LEN + 1];	Sex of the card owner
char BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];	Birth date of the card owner
char SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];	Social security number of the card owner
char CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];	Date of issue
char ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];	Card validity begin date
char ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];	Card validity end date

char CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];	Card number
char CardName[SIS_MAX_CARDNAME_LEN + 1];	Name of the card

1.2.9 tagSISRecordW Structure

Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

C++

```
struct tagSISRecordW {
    WCHAR Name[SIS_MAX_NAME_LEN + 1];
    WCHAR FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];
    WCHAR Initial[SIS_MAX_INITIAL_LEN+ 1];
    WCHAR Sex[SIS_MAX_SEX_LEN + 1];
    WCHAR BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];
    WCHAR SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];
    WCHAR CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];
    WCHAR ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];
    WCHAR ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];
    WCHAR CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];
    WCHAR CardName[SIS_MAX_CARDNAME_LEN +1 ];
};
```

File

CardStructures.h (see page 150)

Members

Members	Description
WCHAR Name[SIS_MAX_NAME_LEN + 1];	Family name of the card owner
WCHAR FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];	First name of the card owner
WCHAR Initial[SIS_MAX_INITIAL_LEN+ 1];	Initial of the card owner
WCHAR Sex[SIS_MAX_SEX_LEN + 1];	Sex of the card owner
WCHAR BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];	Birth date of the card owner
WCHAR SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];	Social security number of the card owner
WCHAR CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];	Date of issue
WCHAR ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];	Card validity begin date
WCHAR ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];	Card validity end date
WCHAR CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];	Card number
WCHAR CardName[SIS_MAX_CARDNAME_LEN +1];	Name of the card

1.2.10 CardEventType Enumeration

The type of the reader event

C++

```
typedef enum tagCardEventType {
    ewtUnknownEvent,
    ewtCardInsert,
    ewtCardRemove,
    ewtReadersChange
} CardEventType;
```

File

CardEvents.h (see page 150)

Members

Members	Description
ewtUnknownEvent	Unknown event
ewtCardInsert	The card was inserted in the reader
ewtCardRemove	The card was removed from the reader
ewtReadersChange	The readers list changed

1.2.11 EidAddressA Structure

EID address information, stored on the card - ANSI version

C++

```
typedef struct tagEidAddressA {  
    char street[EID_MAX_STREET_LEN+1];  
    char zip[EID_MAX_ZIP_LEN+1];  
    char municipality[EID_MAX_MUNICIPALITY_LEN+1];  
} EidAddressA, * PEidAddressA;
```

File

CardStructures.h (see page 150)

Members

Members	Description
char street[EID_MAX_STREET_LEN+1];	Street name
char zip[EID_MAX_ZIP_LEN+1];	ZIP code
char municipality[EID_MAX_MUNICIPALITY_LEN+1];	Municipality

1.2.12 EidAddressW Structure

EID address information, stored on the card - UNICODE version

C++

```
typedef struct tagEidAddressW {  
    WCHAR street[EID_MAX_STREET_LEN+1];  
    WCHAR zip[EID_MAX_ZIP_LEN+1];  
    WCHAR municipality[EID_MAX_MUNICIPALITY_LEN+1];  
} EidAddressW, * PEidAddressW;
```

File

CardStructures.h (see page 150)

Members

Members	Description
WCHAR street[EID_MAX_STREET_LEN+1];	Street name
WCHAR zip[EID_MAX_ZIP_LEN+1];	ZIP code
WCHAR municipality[EID_MAX_MUNICIPALITY_LEN+1];	Municipality

1.2.13 EidCertificate Structure

Certificate, stored on EID card

C++

```
typedef struct tagEidCertificate {  
    BYTE certificate[EID_MAX_CERT_LEN+1];  
    int certificateLength;  
} EidCertificate, * PEidCertificate;
```

File

CardStructures.h (see page 150)

Members

Members	Description
BYTE certificate[EID_MAX_CERT_LEN+1];	Certificate raw data buffer
int certificateLength;	Certificate data length

1.2.14 EidIdentityA Structure

Identity information stored on EID card - ANSI version

C++

```
typedef struct tagEidIdentityA {  
    char cardNumber[EID_MAX_CARD_NUMBER_LEN+1];  
    char chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];  
    char validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];  
    char validityDateEnd[EID_MAX_DATE_END_LEN +1];  
    char municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];  
    char nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];  
    char name[EID_MAX_NAME_LEN+1];  
    char firstName1[EID_MAX_FIRST_NAME1_LEN+1];  
    char firstName2[EID_MAX_FIRST_NAME2_LEN+1];  
    char nationality[EID_MAX_NATIONALITY_LEN+1];  
    char birthLocation[EID_MAX_BIRTHPLACE_LEN+1];  
    char birthDate[EID_MAX_BIRTHDATE_LEN+1];  
    char sex[EID_MAX_SEX_LEN+1];  
    char nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];  
    int documentType;  
    BOOL whiteCane;  
    BOOL yellowCane;  
    BOOL extendedMinority;  
} EidIdentityA, * PEidIdentityA;
```

File

CardStructures.h (see page 150)

Members

Members	Description
char cardNumber[EID_MAX_CARD_NUMBER_LEN+1];	Electronic ID card number
char chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];	Electronic ID card physical chip number
char validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];	Card validity start date
char validityDateEnd[EID_MAX_DATE_END_LEN +1];	Card validity end date
char municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];	Card delivery municipality

char nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];	National number
char name[EID_MAX_NAME_LEN+1];	Surname
char firstName1[EID_MAX_FIRST_NAME1_LEN+1];	First name (2 first given names)
char firstName2[EID_MAX_FIRST_NAME2_LEN+1];	First name part 2 (first letter of the 3rd given name).
char nationality[EID_MAX_NATIONALITY_LEN+1];	Nationality
char birthLocation[EID_MAX_BIRTHPLACE_LEN+1];	Birth location
char birthDate[EID_MAX_BIRTHDATE_LEN+1];	Birth date
char sex[EID_MAX_SEX_LEN+1];	Sex
char nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];	Noble condition
int documentType;	Document type code (Belgian citizen card, Kids Card, Foreigner card)
BOOL whiteCane;	White cane (blind people)
BOOL yellowCane;	Yellow cane (partially sighted people)
BOOL extendedMinority;	Extended minority

1.2.15 EidIdentityW Structure

Identity information stored on EID card - UNICODE version

C++

```
typedef struct tagEidIdentityW {
    WCHAR cardNumber[EID_MAX_CARD_NUMBER_LEN+1];
    WCHAR chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];
    WCHAR validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];
    WCHAR validityDateEnd[EID_MAX_DATE_END_LEN +1];
    WCHAR municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];
    WCHAR nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];
    WCHAR name[EID_MAX_NAME_LEN+1];
    WCHAR firstName1[EID_MAX_FIRST_NAME1_LEN+1];
    WCHAR firstName2[EID_MAX_FIRST_NAME2_LEN+1];
    WCHAR nationality[EID_MAX_NATIONALITY_LEN+1];
    WCHAR birthLocation[EID_MAX_BIRTHPLACE_LEN+1];
    WCHAR birthDate[EID_MAX_BIRTHDATE_LEN+1];
    WCHAR sex[EID_MAX_SEX_LEN+1];
    WCHAR nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];
    int documentType;
    BOOL whiteCane;
    BOOL yellowCane;
    BOOL extendedMinority;
} EidIdentityW, * PEidIdentityW;
```

File

CardStructures.h (see page 150)

Members

Members	Description
WCHAR cardNumber[EID_MAX_CARD_NUMBER_LEN+1];	Electronic ID card number
WCHAR chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];	Electronic ID card physical chip number
WCHAR validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];	Card validity start date
WCHAR validityDateEnd[EID_MAX_DATE_END_LEN +1];	Card validity end date
WCHAR municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];	Card delivery municipality

WCHAR nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];	National number
WCHAR name[EID_MAX_NAME_LEN+1];	Surname
WCHAR firstName1[EID_MAX_FIRST_NAME1_LEN+1];	First name (2 first given names)
WCHAR firstName2[EID_MAX_FIRST_NAME2_LEN+1];	First name part 2 (first letter of the 3rd given name).
WCHAR nationality[EID_MAX_NATIONALITY_LEN+1];	Nationality
WCHAR birthLocation[EID_MAX_BIRTHPLACE_LEN+1];	Birth location
WCHAR birthDate[EID_MAX_BIRTHDATE_LEN+1];	Birth date
WCHAR sex[EID_MAX_SEX_LEN+1];	Sex
WCHAR nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];	Noble condition
int documentType;	Document type code (Belgian citizen card, Kids Card, Foreigner card)
BOOL whiteCane;	White cane (blind people)
BOOL yellowCane;	Yellow cane (partially sighted people)
BOOL extendedMinority;	Extended minority

1.2.16 EidPicture Structure

Raw picture data from EID card

C++

```
typedef struct tagEidPicture {
    BYTE picture[EID_MAX_PICTURE_LEN+1];
    int pictureLength;
} EidPicture, * PEidPicture;
```

File

CardStructures.h (see page 150)

Members

Members	Description
BYTE picture[EID_MAX_PICTURE_LEN+1];	Picture raw data buffer
int pictureLength;	Picture raw data buffer length

1.2.17 PEidAddressA Structure

EID address information, stored on the card - ANSI version

C++

```
typedef struct tagEidAddressA {
    char street[EID_MAX_STREET_LEN+1];
    char zip[EID_MAX_ZIP_LEN+1];
    char municipality[EID_MAX_MUNICIPALITY_LEN+1];
} EidAddressA, * PEidAddressA;
```

File

CardStructures.h (see page 150)

Members

Members	Description
char street[EID_MAX_STREET_LEN+1];	Street name

char zip[EID_MAX_ZIP_LEN+1];	ZIP code
char municipality[EID_MAX_MUNICIPALITY_LEN+1];	Municipality

1.2.18 PEidAddressW Structure

EID address information, stored on the card - UNICODE version

C++

```
typedef struct tagEidAddressW {  
    WCHAR street[EID_MAX_STREET_LEN+1];  
    WCHAR zip[EID_MAX_ZIP_LEN+1];  
    WCHAR municipality[EID_MAX_MUNICIPALITY_LEN+1];  
} EidAddressW, * PEidAddressW;
```

File

CardStructures.h (see page 150)

Members

Members	Description
WCHAR street[EID_MAX_STREET_LEN+1];	Street name
WCHAR zip[EID_MAX_ZIP_LEN+1];	ZIP code
WCHAR municipality[EID_MAX_MUNICIPALITY_LEN+1];	Municipality

1.2.19 PEidCertificate Structure

Certificate, stored on EID card

C++

```
typedef struct tagEidCertificate {  
    BYTE certificate[EID_MAX_CERT_LEN+1];  
    int certificateLength;  
} EidCertificate, * PEidCertificate;
```

File

CardStructures.h (see page 150)

Members

Members	Description
BYTE certificate[EID_MAX_CERT_LEN+1];	Certificate raw data buffer
int certificateLength;	Certificate data length

1.2.20 PEidIdentityA Structure

Identity information stored on EID card - ANSI version

C++

```
typedef struct tagEidIdentityA {  
    char cardNumber[EID_MAX_CARD_NUMBER_LEN+1];  
    char chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];  
    char validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];  
    char validityDateEnd[EID_MAX_DATE_END_LEN +1];  
    char municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];  
};
```



```

char nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];
char name[EID_MAX_NAME_LEN+1];
char firstName1[EID_MAX_FIRST_NAME1_LEN+1];
char firstName2[EID_MAX_FIRST_NAME2_LEN+1];
char nationality[EID_MAX_NATIONALITY_LEN+1];
char birthLocation[EID_MAX_BIRTHPLACE_LEN+1];
char birthDate[EID_MAX_BIRTHDATE_LEN+1];
char sex[EID_MAX_SEX_LEN+1];
char nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];
int documentType;
BOOL whiteCane;
BOOL yellowCane;
BOOL extendedMinority;
} EidIdentityA, * PEidIdentityA;

```

File

CardStructures.h (see page 150)

Members

Members	Description
char cardNumber[EID_MAX_CARD_NUMBER_LEN+1];	Electronic ID card number
char chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];	Electronic ID card physical chip number
char validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];	Card validity start date
char validityDateEnd[EID_MAX_DATE_END_LEN +1];	Card validity end date
char municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];	Card delivery municipality
char nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];	National number
char name[EID_MAX_NAME_LEN+1];	Surname
char firstName1[EID_MAX_FIRST_NAME1_LEN+1];	First name (2 first given names)
char firstName2[EID_MAX_FIRST_NAME2_LEN+1];	First name part 2 (first letter of the 3rd given name).
char nationality[EID_MAX_NATIONALITY_LEN+1];	Nationality
char birthLocation[EID_MAX_BIRTHPLACE_LEN+1];	Birth location
char birthDate[EID_MAX_BIRTHDATE_LEN+1];	Birth date
char sex[EID_MAX_SEX_LEN+1];	Sex
char nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];	Noble condition
int documentType;	Document type code (Belgian citizen card, Kids Card, Foreigner card)
BOOL whiteCane;	White cane (blind people)
BOOL yellowCane;	Yellow cane (partially sighted people)
BOOL extendedMinority;	Extended minority

1.2.21 PEidIdentityW Structure

Identity information stored on EID card - UNICODE version

C++

```

typedef struct tagEidIdentityW {
    WCHAR cardNumber[EID_MAX_CARD_NUMBER_LEN+1];
    WCHAR chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];
    WCHAR validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];
    WCHAR validityDateEnd[EID_MAX_DATE_END_LEN +1];
    WCHAR municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];
    WCHAR nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];
    WCHAR name[EID_MAX_NAME_LEN+1];
    WCHAR firstName1[EID_MAX_FIRST_NAME1_LEN+1];

```

```

WCHAR firstName2[EID_MAX_FIRST_NAME2_LEN+1];
WCHAR nationality[EID_MAX_NATIONALITY_LEN+1];
WCHAR birthLocation[EID_MAX_BIRTHPLACE_LEN+1];
WCHAR birthDate[EID_MAX_BIRTHDATE_LEN+1];
WCHAR sex[EID_MAX_SEX_LEN+1];
WCHAR nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];
int documentType;
BOOL whiteCane;
BOOL yellowCane;
BOOL extendedMinority;
} EidIdentityW, * PEidIdentityW;

```

File

CardStructures.h (see page 150)

Members

Members	Description
WCHAR cardNumber[EID_MAX_CARD_NUMBER_LEN+1];	Electronic ID card number
WCHAR chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];	Electronic ID card physical chip number
WCHAR validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];	Card validity start date
WCHAR validityDateEnd[EID_MAX_DATE_END_LEN +1];	Card validity end date
WCHAR municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];	Card delivery municipality
WCHAR nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];	National number
WCHAR name[EID_MAX_NAME_LEN+1];	Surname
WCHAR firstName1[EID_MAX_FIRST_NAME1_LEN+1];	First name (2 first given names)
WCHAR firstName2[EID_MAX_FIRST_NAME2_LEN+1];	First name part 2 (first letter of the 3rd given name).
WCHAR nationality[EID_MAX_NATIONALITY_LEN+1];	Nationality
WCHAR birthLocation[EID_MAX_BIRTHPLACE_LEN+1];	Birth location
WCHAR birthDate[EID_MAX_BIRTHDATE_LEN+1];	Birth date
WCHAR sex[EID_MAX_SEX_LEN+1];	Sex
WCHAR nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];	Noble condition
int documentType;	Document type code (Belgian citizen card, Kids Card, Foreigner card)
BOOL whiteCane;	White cane (blind people)
BOOL yellowCane;	Yellow cane (partially sighted people)
BOOL extendedMinority;	Extended minority

1.2.22 PeidPicture Structure

Raw picture data from EID card

C++

```

typedef struct tagEidPicture {
    BYTE picture[EID_MAX_PICTURE_LEN+1];
    int pictureLength;
} EidPicture, * PeidPicture;

```

File

CardStructures.h (see page 150)

Members

Members	Description
BYTE picture[EID_MAX_PICTURE_LEN+1];	Picture raw data buffer
int pictureLength;	Picture raw data buffer length

1.2.23 PSISRecordA Structure

Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

C++

```
typedef struct tagSISRecordA {
    char Name[SIS_MAX_NAME_LEN + 1];
    char FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];
    char Initial[SIS_MAX_INITIAL_LEN + 1];
    char Sex[SIS_MAX_SEX_LEN + 1];
    char BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];
    char SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];
    char CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];
    char ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];
    char ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];
    char CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];
    char CardName[SIS_MAX_CARDNAME_LEN + 1];
} SISRecordA, * PSISRecordA;
```

File

CardStructures.h (see page 150)

Members

Members	Description
char Name[SIS_MAX_NAME_LEN + 1];	Family name of the card owner
char FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];	First name of the card owner
char Initial[SIS_MAX_INITIAL_LEN + 1];	Initial of the card owner
char Sex[SIS_MAX_SEX_LEN + 1];	Sex of the card owner
char BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];	Birth date of the card owner
char SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];	Social security number of the card owner
char CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];	Date of issue
char ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];	Card validity begin date
char ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];	Card validity end date
char CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];	Card number
char CardName[SIS_MAX_CARDNAME_LEN + 1];	Name of the card

1.2.24 PSISRecordW Structure

Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

C++

```
typedef struct tagSISRecordW {
    WCHAR Name[SIS_MAX_NAME_LEN + 1];
    WCHAR FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];
```

```

WCHAR Initial[SIS_MAX_INITIAL_LEN+ 1];
WCHAR Sex[SIS_MAX_SEX_LEN + 1];
WCHAR BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];
WCHAR SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];
WCHAR CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];
WCHAR ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];
WCHAR ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];
WCHAR CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];
WCHAR CardName[SIS_MAX_CARDNAME_LEN +1 ];
} SISRecordW, * PSISRecordW;

```

File

CardStructures.h (see page 150)

Members

Members	Description
WCHAR Name[SIS_MAX_NAME_LEN + 1];	Family name of the card owner
WCHAR FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];	First name of the card owner
WCHAR Initial[SIS_MAX_INITIAL_LEN+ 1];	Initial of the card owner
WCHAR Sex[SIS_MAX_SEX_LEN + 1];	Sex of the card owner
WCHAR BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];	Birth date of the card owner
WCHAR SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];	Social security number of the card owner
WCHAR CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];	Date of issue
WCHAR ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];	Card validity begin date
WCHAR ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];	Card validity end date
WCHAR CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];	Card number
WCHAR CardName[SIS_MAX_CARDNAME_LEN +1];	Name of the card

1.2.25 SISRecordA Structure

Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

C++

```

typedef struct tagSISRecordA {
    char Name[SIS_MAX_NAME_LEN + 1];
    char FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];
    char Initial[SIS_MAX_INITIAL_LEN+ 1];
    char Sex[SIS_MAX_SEX_LEN + 1];
    char BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];
    char SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];
    char CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];
    char ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];
    char ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];
    char CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];
    char CardName[SIS_MAX_CARDNAME_LEN +1 ];
} SISRecordA, * PSISRecordA;

```

File

CardStructures.h (see page 150)

Members

Members	Description
char Name[SIS_MAX_NAME_LEN + 1];	Family name of the card owner
char FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];	First name of the card owner

char Initial[SIS_MAX_INITIAL_LEN+ 1];	Initial of the card owner
char Sex[SIS_MAX_SEX_LEN + 1];	Sex of the card owner
char BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];	Birth date of the card owner
char SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];	Social security number of the card owner
char CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];	Date of issue
char ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];	Card validity begin date
char ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];	Card validity end date
char CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];	Card number
char CardName[SIS_MAX_CARDNAME_LEN + 1];	Name of the card

1.2.26 SISRecordW Structure

Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

C++

```
typedef struct tagSISRecordW {
    WCHAR Name[SIS_MAX_NAME_LEN + 1];
    WCHAR FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];
    WCHAR Initial[SIS_MAX_INITIAL_LEN+ 1];
    WCHAR Sex[SIS_MAX_SEX_LEN + 1];
    WCHAR BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];
    WCHAR SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];
    WCHAR CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];
    WCHAR ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];
    WCHAR ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];
    WCHAR CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];
    WCHAR CardName[SIS_MAX_CARDNAME_LEN + 1 ];
} SISRecordW, * PSISRecordW;
```

File

CardStructures.h (see page 150)

Members

Members	Description
WCHAR Name[SIS_MAX_NAME_LEN + 1];	Family name of the card owner
WCHAR FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];	First name of the card owner
WCHAR Initial[SIS_MAX_INITIAL_LEN+ 1];	Initial of the card owner
WCHAR Sex[SIS_MAX_SEX_LEN + 1];	Sex of the card owner
WCHAR BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];	Birth date of the card owner
WCHAR SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];	Social security number of the card owner
WCHAR CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];	Date of issue
WCHAR ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];	Card validity begin date
WCHAR ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];	Card validity end date
WCHAR CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];	Card number
WCHAR CardName[SIS_MAX_CARDNAME_LEN + 1];	Name of the card

1.3 Macros

The following table lists macros in this documentation.

Macros

Name	Description
EID_MAX_BIRTHDATE_LEN (see page 140)	Maximum length of the birthdate
EID_MAX_BIRTHPLACE_LEN (see page 140)	Maximum length of the birthplace
EID_MAX_CARD_NUMBER_LEN (see page 140)	Maximum length of the card number field
EID_MAX_CERT_LEN (see page 141)	Maximum length of the certificate data
EID_MAX_CHIP_NUMBER_LEN (see page 141)	Maximum length of the chip number field
EID_MAX_DATE_BEGIN_LEN (see page 141)	Maximum length of the begin date field
EID_MAX_DATE_END_LEN (see page 141)	Maximum length of the end date field
EID_MAX_DELIVERY_MUNICIPALITY_LEN (see page 142)	Maximum length of the name of the delivery municipality
EID_MAX_DOCUMENT_TYPE_LEN (see page 142)	Maximum length of the document type field
EID_MAX_FIRST_NAME1_LEN (see page 142)	Maximum length of the first name
EID_MAX_FIRST_NAME2_LEN (see page 142)	Maximum length of the first name
EID_MAX_MUNICIPALITY_LEN (see page 142)	Maximum length of the municipality name field
EID_MAX_NAME_LEN (see page 143)	Maximum length of the surname
EID_MAX_NATIONAL_NUMBER_LEN (see page 143)	Maximum length of the national number
EID_MAX_NATIONALITY_LEN (see page 143)	Maximum length of the nationality
EID_MAX_NOBLE_CONDITION_LEN (see page 143)	Maximum length of the noble condition field
EID_MAX_PICTURE_LEN (see page 144)	Maximum length of the picture data
EID_MAX_SEX_LEN (see page 144)	Maximum length of the sex field
EID_MAX_SPECIAL_STATUS_LEN (see page 144)	Maximum length of the special status field
EID_MAX_STREET_LEN (see page 144)	Maximum length of the street name field
EID_MAX_ZIP_LEN (see page 144)	Maximum length of the ZIP code field
FONT_BOLD (see page 145)	This is macro FONT_BOLD.
FONT_ITALIC (see page 145)	This is macro FONT_ITALIC.
FONT_NORMAL (see page 145)	This is macro FONT_NORMAL.
FONT_STRIKEOUT (see page 145)	This is macro FONT_STRIKEOUT.
FONT_UNDERLINE (see page 146)	This is macro FONT_UNDERLINE.
GetFileSHA256 (see page 146)	This is macro GetFileSHA256.

IID_PPV_ARG (see page 146)	IID_PPV_ARG(IType, ppType) IType is the type of pType ppType is the variable of type IType that will be filled RESULTS in: IID_IType, ppvType will create a compiler error if wrong level of indirection is used. macro for QueryInterface and related functions that require a IID and a (void **) this will insure that the cast is safe and appropriate on C
SaveCardToToXMLStreamEx (see page 146)	This is macro SaveCardToToXMLStreamEx.
SavePersonCsvToStream (see page 147)	This is macro SavePersonCsvToStream.
SIS_FIELD_MAX_BIRTHDATE_LEN (see page 147)	Maximum length of the birth date field
SIS_FIELD_MAX_CAPTUREDATE_LEN (see page 147)	Maximum length of the capture date field
SIS_FIELD_MAX_CARDNUMBER_LEN (see page 147)	Maximum length of the car number field
SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN (see page 148)	Maximum length of the social security number field
SIS_FIELD_MAX_VALIDBEGIN_LEN (see page 148)	Maximum length of the start validity date field
SIS_FIELD_MAX_VALIDEND_LEN (see page 148)	Maximum length of the end validity date field
SIS_MAX_CARDNAME_LEN (see page 148)	Maximum length of the card name field
SIS_MAX_FIRSTNAMES_LEN (see page 149)	Maximum length of the first name field
SIS_MAX_INITIAL_LEN (see page 149)	Maximum length of the initial field
SIS_MAX_NAME_LEN (see page 149)	Maximum length of the surname field
SIS_MAX_SEX_LEN (see page 149)	Maximum length of the sex field
WIDTHBYTES (see page 149)	This is macro WIDTHBYTES.

1.3.1 EID_MAX_BIRTHDATE_LEN Macro

Maximum length of the birthdate

C++

```
#define EID_MAX_BIRTHDATE_LEN 0xc
```

File

CardStructures.h (see page 150)

1.3.2 EID_MAX_BIRTHPLACE_LEN Macro

Maximum length of the birthplace

C++

```
#define EID_MAX_BIRTHPLACE_LEN 0x50
```

File

CardStructures.h (see page 150)

1.3.3 EID_MAX_CARD_NUMBER_LEN Macro

Maximum length of the card number field

C++

```
#define EID_MAX_CARD_NUMBER_LEN 0xc
```

FileCardStructures.h ([see page 150](#))

1.3.4 EID_MAX_CERT_LEN Macro

Maximum length of the certificate data

C++

```
#define EID_MAX_CERT_LEN 0x800
```

FileCardStructures.h ([see page 150](#))

1.3.5 EID_MAX_CHIP_NUMBER_LEN Macro

Maximum length of the chip number field

C++

```
#define EID_MAX_CHIP_NUMBER_LEN 0x20
```

FileCardStructures.h ([see page 150](#))

1.3.6 EID_MAX_DATE_BEGIN_LEN Macro

Maximum length of the begin date field

C++

```
#define EID_MAX_DATE_BEGIN_LEN 0xa
```

FileCardStructures.h ([see page 150](#))

1.3.7 EID_MAX_DATE_END_LEN Macro

Maximum length of the end date field

C++

```
#define EID_MAX_DATE_END_LEN 0xa
```

FileCardStructures.h ([see page 150](#))

1.3.8 EID_MAX_DELIVERY_MUNICIPALITY_LEN Macro

Maximum length of the name of the delivery municipality

C++

```
#define EID_MAX_DELIVERY_MUNICIPALITY_LEN 0x50
```

File

CardStructures.h (see page 150)

1.3.9 EID_MAX_DOCUMENT_TYPE_LEN Macro

Maximum length of the document type field

C++

```
#define EID_MAX_DOCUMENT_TYPE_LEN 0x2
```

File

CardStructures.h (see page 150)

1.3.10 EID_MAX_FIRST_NAME1_LEN Macro

Maximum length of the first name

C++

```
#define EID_MAX_FIRST_NAME1_LEN 0x5f
```

File

CardStructures.h (see page 150)

1.3.11 EID_MAX_FIRST_NAME2_LEN Macro

Maximum length of the first name

C++

```
#define EID_MAX_FIRST_NAME2_LEN 0x3
```

File

CardStructures.h (see page 150)

1.3.12 EID_MAX_MUNICIPALITY_LEN Macro

Maximum length of the municipality name field

C++

```
#define EID_MAX_MUNICIPALITY_LEN 0x43
```

File

CardStructures.h ([see page 150](#))

1.3.13 EID_MAX_NAME_LEN Macro

Maximum length of the surname

C++

```
#define EID_MAX_NAME_LEN 0x6e
```

File

CardStructures.h ([see page 150](#))

1.3.14 EID_MAX_NATIONAL_NUMBER_LEN Macro

Maximum length of the national number

C++

```
#define EID_MAX_NATIONAL_NUMBER_LEN 0xb
```

File

CardStructures.h ([see page 150](#))

1.3.15 EID_MAX_NATIONALITY_LEN Macro

Maximum length of the nationality

C++

```
#define EID_MAX_NATIONALITY_LEN 0x55
```

File

CardStructures.h ([see page 150](#))

1.3.16 EID_MAX_NOBLE_CONDITION_LEN Macro

Maximum length of the noble condition field

C++

```
#define EID_MAX_NOBLE_CONDITION_LEN 0x32
```

File

CardStructures.h ([see page 150](#))

1.3.17 EID_MAX_PICTURE_LEN Macro

Maximum length of the picture data

C++

```
#define EID_MAX_PICTURE_LEN 0x1000
```

File

CardStructures.h (see page 150)

1.3.18 EID_MAX_SEX_LEN Macro

Maximum length of the sex field

C++

```
#define EID_MAX_SEX_LEN 0x1
```

File

CardStructures.h (see page 150)

1.3.19 EID_MAX_SPECIAL_STATUS_LEN Macro

Maximum length of the special status field

C++

```
#define EID_MAX_SPECIAL_STATUS_LEN 0x2
```

File

CardStructures.h (see page 150)

1.3.20 EID_MAX_STREET_LEN Macro

Maximum length of the street name field

C++

```
#define EID_MAX_STREET_LEN 0x50
```

File

CardStructures.h (see page 150)

1.3.21 EID_MAX_ZIP_LEN Macro

Maximum length of the ZIP code field

C++

```
#define EID_MAX_ZIP_LEN 0x4
```

File

CardStructures.h ([see page 150](#))

1.3.22 FONT_BOLD Macro

This is macro FONT_BOLD.

C++

```
#define FONT_BOLD 0x01
```

File

Graphics.h ([see page 155](#))

1.3.23 FONT_ITALIC Macro

This is macro FONT_ITALIC.

C++

```
#define FONT_ITALIC 0x02
```

File

Graphics.h ([see page 155](#))

1.3.24 FONT_NORMAL Macro

This is macro FONT_NORMAL.

C++

```
#define FONT_NORMAL 0x00
```

File

Graphics.h ([see page 155](#))

1.3.25 FONT_STRIKEOUT Macro

This is macro FONT_STRIKEOUT.

C++

```
#define FONT_STRIKEOUT 0x08
```

File

Graphics.h ([see page 155](#))

1.3.26 FONT_UNDERLINE Macro

This is macro FONT_UNDERLINE.

C++

```
#define FONT_UNDERLINE 0x04
```

File

Graphics.h (see page 155)

1.3.27 GetFileSHA256 Macro

This is macro GetFileSHA256.

C++

```
#define GetFileSHA256 GetFileSHA256A
```

File

Encryption.h (see page 152)

1.3.28 IID_PPV_ARG Macro

IID_PPV_ARG(IType, ppType) IType is the type of pType ppType is the variable of type IType that will be filled

RESULTS in: IID_IType, ppvType will create a compiler error if wrong level of indirection is used.

macro for QueryInterface and related functions that require a IID and a (void **) this will insure that the cast is safe and appropriate on C

C++

```
#define IID_PPV_ARG(IType, ppType) &IID_##IType, (void**)(ppType)
```

File

FileOperations.h (see page 153)

1.3.29 SaveCardToToXMLStreamEx Macro

This is macro SaveCardToToXMLStreamEx.

C++

```
#define SaveCardToToXMLStreamEx SaveCardToToXMLStreamExA
```

File

Swelio.h (see page 156)

1.3.30 SavePersonCsvToStream Macro

This is macro SavePersonCsvToStream.

C++

```
#define SavePersonCsvToStream SavePersonCsvToStreamA
```

File

Swelio.h ([↗](#) see page 156)

1.3.31 SIS_FIELD_MAX_BIRTHDATE_LEN Macro

Maximum length of the birth date field

C++

```
#define SIS_FIELD_MAX_BIRTHDATE_LEN 0x8
```

File

CardStructures.h ([↗](#) see page 150)

1.3.32 SIS_FIELD_MAX_CAPTUREDATE_LEN Macro

Maximum length of the capture date field

C++

```
#define SIS_FIELD_MAX_CAPTUREDATE_LEN 0x8
```

File

CardStructures.h ([↗](#) see page 150)

1.3.33 SIS_FIELD_MAX_CARDNUMBER_LEN Macro

Maximum length of the car number field

C++

```
#define SIS_FIELD_MAX_CARDNUMBER_LEN 0xa
```

File

CardStructures.h ([↗](#) see page 150)

1.3.34

SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN Macro

Maximum length of the social security number field

C++

```
#define SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN 0xb
```

File

CardStructures.h ([see page 150](#))

1.3.35 SIS_FIELD_MAX_VALIDBEGIN_LEN Macro

Maximum length of the start validity date field

C++

```
#define SIS_FIELD_MAX_VALIDBEGIN_LEN 0x8
```

File

CardStructures.h ([see page 150](#))

1.3.36 SIS_FIELD_MAX_VALIDEND_LEN Macro

Maximum length of the end validity date field

C++

```
#define SIS_FIELD_MAX_VALIDEND_LEN 0x8
```

File

CardStructures.h ([see page 150](#))

1.3.37 SIS_MAX_CARDNAME_LEN Macro

Maximum length of the card name field

C++

```
#define SIS_MAX_CARDNAME_LEN 0x6
```

File

CardStructures.h ([see page 150](#))

1.3.38 SIS_MAX_FIRSTNAMES_LEN Macro

Maximum length of the first name field

C++

```
#define SIS_MAX_FIRSTNAMES_LEN 0x18
```

File

CardStructures.h ([see page 150](#))

1.3.39 SIS_MAX_INITIAL_LEN Macro

Maximum length of the initial field

C++

```
#define SIS_MAX_INITIAL_LEN 0x1
```

File

CardStructures.h ([see page 150](#))

1.3.40 SIS_MAX_NAME_LEN Macro

Maximum length of the surname field

C++

```
#define SIS_MAX_NAME_LEN 0x30
```

File

CardStructures.h ([see page 150](#))

1.3.41 SIS_MAX_SEX_LEN Macro

Maximum length of the sex field

C++

```
#define SIS_MAX_SEX_LEN 0x1
```

File

CardStructures.h ([see page 150](#))

1.3.42 WIDTHBYTES Macro

This is macro WIDTHBYTES.

C++

```
#define WIDTHBYTES(i) (((i) + 31) / 32 * 4)
```

File

Graphics.h ([see page 155](#))

1.4 Files

The following table lists files in this documentation.


Files

Name	Description
CardEvents.h (see page 150)	The definition of the possible card events
CardStructures.h (see page 150)	The definition of the information storage structures
Encryption.h (see page 152)	Encryption and decryption operations
FileOperations.h (see page 153)	Files and folders manipulations
Graphics.h (see page 155)	This is file Graphics.h.
NationalityConverter.h (see page 156)	Nationality to ISO code converter
quricol.h (see page 156)	QR Code generator
Swelio.h (see page 156)	Belgian electronic Id card access engine
System.h (see page 161)	Windows related routines
SystemInfo.h (see page 162)	Routines for querying information about operating system

1.4.1 CardEvents.h

The definition of the possible card events

Enumerations

	Name	Description
	tagCardEventType (see page 123)	The type of the reader event
	CardEventType (see page 128)	The type of the reader event

1.4.2 CardStructures.h








The definition of the information storage structures


Macros

Name	Description
EID_MAX_BIRTHDATE_LEN (see page 140)	Maximum length of the birthdate
EID_MAX_BIRTHPLACE_LEN (see page 140)	Maximum length of the birthplace
EID_MAX_CARD_NUMBER_LEN (see page 140)	Maximum length of the card number field
EID_MAX_CERT_LEN (see page 141)	Maximum length of the certificate data
EID_MAX_CHIP_NUMBER_LEN (see page 141)	Maximum length of the chip number field

EID_MAX_DATE_BEGIN_LEN (see page 141)	Maximum length of the begin date field
EID_MAX_DATE_END_LEN (see page 141)	Maximum length of the end date field
EID_MAX_DELIVERY_MUNICIPALITY_LEN (see page 142)	Maximum length of the name of the delivery municipality
EID_MAX_DOCUMENT_TYPE_LEN (see page 142)	Maximum length of the document type field
EID_MAX_FIRST_NAME1_LEN (see page 142)	Maximum length of the first name
EID_MAX_FIRST_NAME2_LEN (see page 142)	Maximum length of the first name
EID_MAX_MUNICIPALITY_LEN (see page 142)	Maximum length of the municipality name field
EID_MAX_NAME_LEN (see page 143)	Maximum length of the surname
EID_MAX_NATIONAL_NUMBER_LEN (see page 143)	Maximum length of the national number
EID_MAX_NATIONALITY_LEN (see page 143)	Maximum length of the nationality
EID_MAX_NOBLE_CONDITION_LEN (see page 143)	Maximum length of the noble condition field
EID_MAX_PICTURE_LEN (see page 144)	Maximum length of the picture data
EID_MAX_SEX_LEN (see page 144)	Maximum length of the sex field
EID_MAX_SPECIAL_STATUS_LEN (see page 144)	Maximum length of the special status field
EID_MAX_STREET_LEN (see page 144)	Maximum length of the street name field
EID_MAX_ZIP_LEN (see page 144)	Maximum length of the ZIP code field
SIS_FIELD_MAX_BIRTHDATE_LEN (see page 147)	Maximum length of the birth date field
SIS_FIELD_MAX_CAPTUREDATE_LEN (see page 147)	Maximum length of the capture date field
SIS_FIELD_MAX_CARDNUMBER_LEN (see page 147)	Maximum length of the card number field
SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN (see page 148)	Maximum length of the social security number field
SIS_FIELD_MAX_VALIDBEGIN_LEN (see page 148)	Maximum length of the start validity date field
SIS_FIELD_MAX_VALIDEND_LEN (see page 148)	Maximum length of the end validity date field
SIS_MAX_CARDNAME_LEN (see page 148)	Maximum length of the card name field
SIS_MAX_FIRSTNAMES_LEN (see page 149)	Maximum length of the first name field
SIS_MAX_INITIAL_LEN (see page 149)	Maximum length of the initial field
SIS_MAX_NAME_LEN (see page 149)	Maximum length of the surname field
SIS_MAX_SEX_LEN (see page 149)	Maximum length of the sex field

Structures

















	Name	Description
	tagEidAddressA (see page 123)	EID address information, stored on the card - ANSI version
	tagEidAddressW (see page 124)	EID address information, stored on the card - UNICODE version
	tagEidCertificate (see page 124)	Certificate, stored on EID card
	tagEidIdentityA (see page 125)	Identity information stored on EID card - ANSI version
	tagEidIdentityW (see page 126)	Identity information stored on EID card - UNICODE version
	tagEidPicture (see page 127)	Raw picture data from EID card
	tagSISRecordA (see page 127)	Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

	tagSISRecordW (see page 128)	Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	EidAddressA (see page 129)	EID address information, stored on the card - ANSI version
	EidAddressW (see page 129)	EID address information, stored on the card - UNICODE version
	EidCertificate (see page 130)	Certificate, stored on EID card
	EidIdentityA (see page 130)	Identity information stored on EID card - ANSI version
	EidIdentityW (see page 131)	Identity information stored on EID card - UNICODE version
	EidPicture (see page 132)	Raw picture data from EID card
	PEidAddressA (see page 132)	EID address information, stored on the card - ANSI version
	PEidAddressW (see page 133)	EID address information, stored on the card - UNICODE version
	PEidCertificate (see page 133)	Certificate, stored on EID card
	PEidIdentityA (see page 133)	Identity information stored on EID card - ANSI version
	PEidIdentityW (see page 134)	Identity information stored on EID card - UNICODE version
	PeidPicture (see page 135)	Raw picture data from EID card
	PSISRecordA (see page 136)	Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	PSISRecordW (see page 136)	Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	SISRecordA (see page 137)	Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	SISRecordW (see page 138)	Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

1.4.3 Encryption.h

Encryption and decryption operations

Functions

	Name	Description
	CardDecryptFileA (see page 14)	Decrypt file using Belgian Id card
	CardDecryptFileW (see page 15)	Decrypt file using Belgian Id card
	CardEncryptFileA (see page 15)	Encrypt file using Belgian Id card
	CardEncryptFileW (see page 15)	Encrypt file using Belgian Id card
	CheckMD5 (see page 17)	Checks the MD5 hash value of the memory buffer
	CheckSHA1 (see page 18)	Checks the SHA1 hash value of the memory buffer
	CheckSHA256 (see page 18)	Checks the SHA256 hash value of the memory buffer
	DecryptFileAESA (see page 24)	Decrypts file using AES algorithm.
	DecryptFileAESW (see page 24)	Decrypts file using AES algorithm.
	EncryptFileAESA (see page 32)	Encrypts file using AES algorithm.
	EncryptFileAESW (see page 32)	Encrypts file using AES algorithm.
	GetFileMD5A (see page 54)	Gets the MD5 hash value for the file
	GetFileMD5W (see page 55)	Gets the MD5 hash value for the file
	GetFileSHA1A (see page 56)	Gets the SHA1 hash value for the file
	GetFileSHA1W (see page 56)	Gets the SHA1 hash value for the file
	GetFileSHA256A (see page 57)	Gets the SHA256 hash value for the file

◆	GetFileSHA256W (see page 57)	Gets the SHA256 hash value for the file
◆	GetMD5 (see page 60)	Gets the MD5 hash value for the content of the memory buffer
◆	GetSHA1 (see page 64)	Gets the SHA1 hash value for the content of the memory buffer
◆	GetSHA256 (see page 64)	Gets the SHA256 hash value for the content of the memory buffer

Macros

Name	Description
GetFileSHA256 (see page 146)	This is macro GetFileSHA256.

1.4.4 FileOperations.h

Files and folders manipulations

Functions

	Name	Description
◆	AllocateBuffer (see page 11)	Allocates the buffer in memory
◆	ClearFileAttributesA (see page 19)	This function sets the file attributes to normal.
◆	ClearFileAttributesW (see page 19)	This function sets the file attributes to normal.
◆	CreateUnicodeFileA (see page 21)	Creates UNICODE file
◆	CreateUnicodeFileW (see page 21)	Creates UNICODE file
◆	DeallocateBuffer (see page 23)	Deallocates the memory buffer
◆	DeleteToRecycleBinA (see page 25)	Deletes file to the Windows Recycle Bin
◆	DeleteToRecycleBinW (see page 25)	Deletes file to WIndows Recycle Bin
◆	DirectoryExistsA (see page 26)	Determines whether a specified directory exists.
◆	DirectoryExistsW (see page 27)	Determines whether a specified directory exists.
◆	FileCloseA (see page 33)	Concludes input/output (I/O) to a file opened using the FileCreateRewrite function.
◆	FileCloseW (see page 33)	Concludes input/output (I/O) to a file opened using the FileCreateRewrite function.
◆	FileCopyA (see page 33)	The CopyFile function copies an existing file to a new file.
◆	FileCopyW (see page 34)	The CopyFile function copies an existing file to a new file.
◆	FileCreateRewriteA (see page 34)	Creates new or overwrites existing file
◆	FileCreateRewriteW (see page 34)	Creates new or overwrites existing file
◆	FileDeleteA (see page 35)	Deletes a file from disk.
◆	FileDeleteW (see page 35)	Deletes a file from disk.
◆	FileExistsA (see page 36)	Tests whether a specified file exists.
◆	FileExistsW (see page 36)	Tests whether a specified file exists.
◆	FileExtensionIsA (see page 36)	Checks the file extension
◆	FileExtensionIsW (see page 37)	Checks the file extension
◆	FileGetSizeA (see page 37)	Retrieves the size of a specified file.
◆	FileGetSizeW (see page 37)	Retrieves the size of a specified file.
◆	FileIsExeA (see page 38)	Checks if the file is a Windows executable
◆	FileIsExeW (see page 38)	Checks if the file is a Windows executable

FileIsIconA (see page 39)	Checks if the file is a Windows icon (.ico) file
FileIsIconW (see page 39)	Checks if the file is a Windows icon (.ico) file
FileIsImageA (see page 39)	Checks if the file is an image file
FileIsImageW (see page 40)	Checks if the file is an image file
FileIsLink (see page 40)	Checks to see if the file specified by file name is a Microsoft Windows shortcut (.Lnk) file (and is neither a file nor a folder).
FileOrFolderExistsA (see page 40)	Checks if the file or folder with the given name exists
FileOrFolderExistsW (see page 41)	Checks if the file or folder with the given name exists
FileRenameA (see page 41)	Renames the file
FileRenameW (see page 41)	Renames the file
FileWriteA (see page 42)	Writes string to the file
FileWriteCharA (see page 42)	Writes one character to the file
FileWriteCharW (see page 42)	Writes one character to the file
FileWriteNewLineA (see page 43)	Writes new line sequence to the file
FileWriteNewLineW (see page 43)	Writes new line sequence to the file
FileWriteW (see page 43)	Writes string to the file
FullPathA (see page 44)	Gets the full path to the file based on file name
FullPathW (see page 44)	Gets the full path to the file based on file name
GetAllFiles (see page 52)	Returns the names of files in a specified directory.
GetFilesCountA (see page 55)	Calculates the number of files in the given folder
GetFilesCountW (see page 56)	Calculates the number of files in the given folder
IsAnimatedGIFA (see page 67)	Checks if the file is an animated GIF image file
IsAnimatedGIFW (see page 67)	Checks if the file is an animated GIF image file
IsDirectoryA (see page 70)	Verifies that a path is a valid directory.
IsDirectoryW (see page 70)	Verifies that a path is a valid directory.
IsUnicodeFileA (see page 75)	Checks if the file is UNICODE file
IsUnicodeFileW (see page 76)	Checks if the file is UNICODE file
IsValidFileNameA (see page 76)	Checks if provided string is a valid file name
IsValidFileNameW (see page 76)	Checks if provided string is a valid file name
IsValidPathNameA (see page 77)	Checks if provided string is a valid file path
IsValidPathNameW (see page 77)	Checks if provided string is a valid file path
ReadBufferFromFileA (see page 87)	Reads the content of the file to the memory buffer
ReadBufferFromFileW (see page 87)	Reads the content of the file to the memory buffer
ShellCopyFileA (see page 115)	Copies file to the new location
ShellCopyFileW (see page 116)	Copies file to the new location
StripFileNameA (see page 118)	Replaces environment variable names with values
StripFileNameW (see page 118)	Replaces environment variable names with values
WriteBufferToFileA (see page 121)	Writes the memory buffer to file
WriteBufferToFileW (see page 122)	Writes the memory buffer to file

Macros

Name	Description
<code>IID_PPV_ARG</code> (see page 146)	<code>IID_PPV_ARG(IType, ppType)</code> IType is the type of pType ppType is the variable of type IType that will be filled RESULTS in: IID_IType, ppvType will create a compiler error if wrong level of indirection is used. macro for QueryInterface and related functions that require a IID and a (void **) this will insure that the cast is safe and appropriate on C

1.4.5 Graphics.h

This is file Graphics.h.

Functions

	Name	Description
⇒	<code>AlphaBlendBitmap</code> (see page 14)	This is function AlphaBlendBitmap.
⇒	<code>AlphaBlendNative</code> (see page 14)	This is function AlphaBlendNative.
⇒	<code>CloneFont</code> (see page 20)	This is function CloneFont.
⇒	<code>CopyNativeBitmap</code> (see page 20)	This is function CopyNativeBitmap.
⇒	<code>CreateNativeBitmap</code> (see page 20)	This is function CreateNativeBitmap.
⇒	<code>CreateWindowsFont</code> (see page 21)	This is function CreateWindowsFont.
⇒	<code>DestroyFont</code> (see page 26)	This is function DestroyFont.
⇒	<code>DpiY</code> (see page 27)	This is function DpiY.
⇒	<code>DrawAlphaText</code> (see page 28)	This is function DrawAlphaText.
⇒	<code>DrawAlphaTextRect</code> (see page 28)	This is function DrawAlphaTextRect.
⇒	<code>DrawNativeBitmap</code> (see page 29)	This is function DrawNativeBitmap.
⇒	<code>DrawTextDirect</code> (see page 29)	This is function DrawTextDirect.
⇒	<code>DrawTextDirectEx</code> (see page 29)	This is function DrawTextDirectEx.
⇒	<code>DrawTextGlow</code> (see page 29)	This is function DrawTextGlow.
⇒	<code>DrawTextLine</code> (see page 30)	This is function DrawTextLine.
⇒	<code>DrawTextOutline</code> (see page 30)	This is function DrawTextOutline.
⇒	<code>DrawTextRect</code> (see page 30)	This is function DrawTextRect.
⇒	<code>EmToPixels</code> (see page 31)	This is function EmToPixels.
⇒	<code>GetTextLineSize</code> (see page 66)	This is function GetTextLineSize.
⇒	<code>GetTextSize</code> (see page 66)	This is function GetTextSize.
⇒	<code>GetTextSizeEx</code> (see page 66)	This is function GetTextSizeEx.
⇒	<code>LoadBitmapJPG</code> (see page 80)	This is function LoadBitmapJPG.
⇒	<code>LoadBitmapPNG</code> (see page 80)	This is function LoadBitmapPNG.
⇒	<code>LoadPNGResource</code> (see page 82)	This is function LoadPNGResource.
⇒	<code>MakeCompatibleBitmap</code> (see page 83)	This is function MakeCompatibleBitmap.
⇒	<code>PointsToPixels</code> (see page 84)	This is function PointsToPixels.
⇒	<code>StretchNativeBitmap</code> (see page 117)	This is function StretchNativeBitmap.

Macros

Name	Description
FONT_BOLD (see page 145)	This is macro FONT_BOLD.
FONT_ITALIC (see page 145)	This is macro FONT_ITALIC.
FONT_NORMAL (see page 145)	This is macro FONT_NORMAL.
FONT_STRIKEOUT (see page 145)	This is macro FONT_STRIKEOUT.
FONT_UNDERLINE (see page 146)	This is macro FONT_UNDERLINE.
WIDTHBYTES (see page 149)	This is macro WIDTHBYTES.

1.4.6 NationalityConverter.h

Nationality to ISO code converter

Functions

	Name	Description
⇒	GetISOCodeA (see page 59)	Returns the country ISO code based on the nationality string
⇒	GetISOCodeW (see page 59)	Returns the country ISO code based on the nationality string

1.4.7 quricol.h

QR Code generator

Functions

	Name	Description
⇒	DestroyImageBuffer (see page 26)	Destroys the memory buffer
⇒	GenerateBMPA (see page 46)	Generates Windows Bitmap file with QR Code image
⇒	GenerateBMPW (see page 47)	Generates Windows Bitmap file with QR Code image
⇒	GeneratePNGA (see page 49)	Generates PNG file with QR Code image
⇒	GeneratePNGW (see page 50)	Generates PNG file with QR Code image
⇒	GetHBitmapA (see page 58)	Generates Windows Bitmap in memory with QR Code image
⇒	GetHBitmapW (see page 58)	Generates Windows Bitmap in memory with QR Code image
⇒	GetPNGA (see page 60)	Writes PNG image to the memory buffer.
⇒	GetPNGW (see page 61)	Writes PNG image to the memory buffer.

1.4.8 Swelio.h

Belgian electronic Id card access engine

Functions

	Name	Description
⇒	ActivateCard (see page 9)	Established communication between the card and the reader
⇒	ActivateCardEx (see page 10)	Established communication between the card and the reader
⇒	CardSignCadesT (see page 16)	Sign data with eID card according to CADES-T standard
⇒	CardSignCMS (see page 16)	Sign data with eID card according to CMS standard

⇒	CertSignCadesT (see page 17)	This is function CertSignCadesT.
⇒	CertSignCMS (see page 17)	This is function CertSignCMS.
⇒	CreateCardBuffer (see page 20)	Creates XML buffer
⇒	DeactivateCard (see page 22)	Terminates a connection between a smart card and a reader
⇒	DeactivateCardEx (see page 22)	Terminates a connection between a smart card and a reader
⇒	DeleteCardBuffer (see page 25)	Deletes XML buffer
⇒	DisplayCertificate (see page 27)	Displays the dialog window with certificate information
⇒	EncodeCertificate (see page 31)	Performs Base64 encoding of the certificate
⇒	EncodePhoto (see page 31)	Performs Base64 encoding of the photo
⇒	GenerateAuthenticationSignatureA (see page 44)	Generate authentication signature
⇒	GenerateAuthenticationSignatureExA (see page 45)	Generate authentication signature
⇒	GenerateAuthenticationSignatureExW (see page 45)	Generate authentication signature
⇒	GenerateAuthenticationSignatureW (see page 46)	Generate authentication signature
⇒	GenerateNonRepudiationSignatureA (see page 47)	Generate non repudiation signature
⇒	GenerateNonRepudiationSignatureExA (see page 48)	Generate non repudiation signature
⇒	GenerateNonRepudiationSignatureExW (see page 48)	Generate non repudiation signature
⇒	GenerateNonRepudiationSignatureW (see page 49)	Generate non repudiation signature
⇒	GenerateQRCodeA (see page 50)	Read eID card and save the identity information and address to PNG QR Code file
⇒	GenerateQRCodeExA (see page 51)	Read eID card and save the identity information and address to PNG QR Code file
⇒	GenerateQRCodeExW (see page 51)	Read eID card and save the identity information and address to PNG QR Code file
⇒	GenerateQRCodeW (see page 51)	Read eID card and save the identity information and address to PNG QR Code file
⇒	GetCardBufferA (see page 52)	This is function GetCardBufferA.
⇒	GetCardBufferSize (see page 52)	This function returns the size of the buffer needed to hold the information from the eID card in the XML or CSV format
⇒	GetCardBufferW (see page 53)	This is function GetCardBufferW.
⇒	GetCardSerialNumber (see page 53)	This is function GetCardSerialNumber.
⇒	GetCardVersion (see page 53)	Get the applet version number for card in the reader with specified number
⇒	GetEncodedCertificateSize (see page 54)	Returns the size of the Base64 encoded certificate
⇒	GetEncodedPhotoSize (see page 54)	Calculates buffer size for Base64 encoded photo
⇒	GetReaderIndexA (see page 61)	Returns the zero-based reader index with specified name
⇒	GetReaderIndexW (see page 61)	Returns the zero-based reader index with specified name
⇒	GetReaderNameA (see page 62)	Returns the name of the card reader
⇒	GetReaderNameLenA (see page 62)	Returns the length of the reader name
⇒	GetReaderNameLenW (see page 63)	Returns the length of the reader name
⇒	GetReaderNameW (see page 63)	Returns the name of the card reader
⇒	GetReadersCount (see page 63)	Get number of card readers connected to PC
⇒	GetSelectedReaderIndex (see page 64)	Returns the index of the active smart card reader
⇒	GetSupportSIS (see page 66)	Checks if the SIS cards are supported by the engine

IsCardActivated (see page 68)	Checks the connection between a smart card and a reader
IsCardActivatedEx (see page 68)	Checks the connection between a smart card and a reader
IsCardPresent (see page 68)	Checks if the card is present in the card reader
IsCardPresentEx (see page 69)	Checks if the card is present in the card reader
IsCardStillInserted (see page 69)	Checks if the card is still inserted in the card reader
IsCardStillInsertedEx (see page 69)	Checks if the card is still inserted in the card reader
IsEIDCard (see page 71)	Check if Belgian EID card is inserted into card reader
IsEIDCardEx (see page 71)	Check if Belgian EID card is inserted into card reader
IsEngineActive (see page 71)	Checks if the Swelio Engine is activated
IsFemaleA (see page 72)	Checks if the card owner is female
IsFemaleW (see page 72)	Checks if the card owner is female
IsMaleA (see page 72)	Checks if the card owner is male
IsMaleW (see page 73)	Checks if the card owner is male
IsSISCard (see page 74)	Check if Belgian SIS card is inserted into card reader
IsSISCardEx (see page 75)	Check if Belgian SIS card is inserted into card reader
LoadCertificateA (see page 80)	Reads the certificate from a file
LoadCertificateW (see page 81)	Reads the certificate from a file
LoadIdentityA (see page 81)	Reads the raw identity information from a file
LoadIdentityW (see page 81)	Reads the raw identity information from a file
LoadPhotoA (see page 82)	Loads photo from a file
LoadPhotoW (see page 82)	Loads photo from a file
ReadAddressA (see page 85)	Read address information from Belgian eID card
ReadAddressExA (see page 85)	Read address information from Belgian eID card
ReadAddressExW (see page 86)	Read address information from Belgian eID card
ReadAddressW (see page 86)	Read address information from Belgian eID card
ReadAuthenticationCertificate (see page 86)	Read Authentication Certificate to memory
ReadAuthenticationCertificateEx (see page 87)	Read Authentication Certificate to memory
ReadCaCertificate (see page 88)	Read Ca Certificate to memory
ReadCaCertificateEx (see page 88)	Read Ca Certificate to memory
ReadIdentityA (see page 89)	Read identity information from Belgian eID card
ReadIdentityExA (see page 89)	Read identity information from Belgian eID card
ReadIdentityExW (see page 89)	Read identity information from Belgian eID card
ReadIdentityW (see page 90)	Read identity information from Belgian eID card
ReadNonRepudiationCertificate (see page 90)	Read Non Repudiation Certificate to memory
ReadNonRepudiationCertificateEx (see page 91)	Read Non Repudiation Certificate to memory
ReadPhoto (see page 91)	Reads a photo from a card
ReadPhotoAsBitmap (see page 91)	Reads the picture from the card, converts it to bitmap and returns the bitmap handle Description: Reads the photo from the Belgian eID card and returns the bitmap handle Reading the photo from the card is a time consuming operation. Do it only when needed.
ReadPhotoAsBitmapEx (see page 92)	Reads the picture from the card, converts it to bitmap and returns the bitmap handle Description: Reads the photo from the Belgian eID card and returns the Windows bitmap handle Reading the photo from the card is a time consuming operation. Do it only when needed.
ReadPhotoEx (see page 92)	Reads a photo from a card
ReadRootCaCertificate (see page 93)	Read Root Ca Certificate to memory
ReadRootCaCertificateEx (see page 93)	Read Root Ca Certificate to memory

ReadRrnCertificate (see page 93)	Read Rrn Certificate to memory
ReadRrnCertificateEx (see page 94)	Read Rrn Certificate to memory
ReadSISCardA (see page 94)	Read Belgian SIS card.
ReadSISCardExA (see page 95)	Read Belgian SIS card.
ReadSISCardExW (see page 95)	Read Belgian SIS card.
ReadSISCardW (see page 95)	Read Belgian SIS card.
ReloadReadersList (see page 96)	Reloads the list of the available card readers
RemoveCallback (see page 96)	Remove callback procedure for card events
SaveAuthenticationCertificateA (see page 98)	Save Authentication Certificate to a file
SaveAuthenticationCertificateExW (see page 98)	Save Authentication Certificate to a file
SaveAuthenticationCertificateW (see page 99)	Save Authentication Certificate to a file
SaveCaCertificateA (see page 99)	Save Ca Certificate to a file
SaveCaCertificateExW (see page 99)	Save Ca Certificate to a file
SaveCaCertificateW (see page 100)	Save Ca Certificate to a file
SaveCardToToXMLStreamExA (see page 100)	Read eID card and save the information to XML buffer
SaveCardToToXMLStreamExW (see page 100)	Read eID card and save the information to XML buffer
SaveCardToXmlA (see page 101)	Read eID card and save the information to XML file
SaveCardToXmlExA (see page 101)	Read eID card and save the information to XML file
SaveCardToXmlExW (see page 102)	Read eID card and save the information to XML file
SaveCardToXmlW (see page 102)	Read eID card and save the information to XML file
SavIdentityA (see page 103)	Saves indentity information to a file
SavIdentityW (see page 103)	Saves indentity information to a file
SaveNonRepudiationCertificateA (see page 103)	Save Non Repudiation Certificate to a file
SaveNonRepudiationCertificateExW (see page 104)	Save Non Repudiation Certificate to a file
SaveNonRepudiationCertificateW (see page 104)	Save Non Repudiation Certificate to a file
SavePersonCsvToStreamA (see page 104)	This is function SavePersonCsvToStreamA.
SavePersonCsvToStreamW (see page 105)	This is function SavePersonCsvToStreamW.
SavePersonToCsvA (see page 105)	Read eID card and save the identity information and address to CSV file
SavePersonToCsvExA (see page 105)	Read eID card and save the identity information and address to CSV file
SavePersonToCsvExW (see page 106)	Read eID card and save the identity information and address to CSV file
SavePersonToCsvW (see page 106)	Read eID card and save the identity information and address to CSV file
SavePhotoA (see page 106)	Save photo to a file
SavePhotoAsBitmapA (see page 107)	Save the picture from the card to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsBitmapExA (see page 107)	Reads the picture from the card and saves it to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.

SavePhotoAsBitmapExW (see page 108)	Reads the picture from the card and saves it to Windows Bitmap file Description: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsBitmapW (see page 108)	Save the picture from the card to Windows Bitmap file Description: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsJpegA (see page 108)	Save the picture from the card to JPG file Description: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsJpegExA (see page 109)	Save the picture from the card to JPG file Description: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsJpegExW (see page 109)	Save the picture from the card to JPG file Description: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoAsJpegW (see page 109)	Save the picture from the card to JPG file Description: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
SavePhotoW (see page 110)	Saves photo to a file
SaveRootCaCertificateA (see page 110)	Save Root Ca Certificate to a file
SaveRootCaCertificateExW (see page 111)	Save Root Ca Certificate to a file
SaveRootCaCertificateW (see page 111)	Save Root Ca Certificate to a file
SaveRrnCertificateA (see page 111)	Save RRN Certificate to a file
SaveRrnCertificateExW (see page 112)	Save RRN Certificate to a file
SaveRrnCertificateW (see page 112)	Save RRN Certificate to a file
SelectReader (see page 112)	When more than 1 reader connected, select the reader with specified number
SelectReaderByNameA (see page 113)	Select active smart card reader by providing the reader name
SelectReaderByNameW (see page 113)	Select active smart card reader by providing the reader name
SendAPDU (see page 114)	This is function SendAPDU.
SetCallback (see page 114)	Activates callback procedure for card status change event
SetMWCompatibility (see page 114)	Set the compatibility mode with the old version of the official EID MiddleWare
SetSupportSIS (see page 115)	Activates or deactivates SIS card support by engine
StartEngine (see page 117)	Activates the Swelio Engine.
StopEngine (see page 117)	Deactivates the Swelio Engine
VerifyPinA (see page 119)	Verify PIN code
VerifyPinExA (see page 120)	Verify PIN code
VerifyPinExW (see page 120)	Verify PIN code
VerifyPinW (see page 120)	Verify PIN code
VerifySignature (see page 121)	Verifies the signature from the specified hash value.

Macros

Name	Description
SaveCardToToXMLStreamEx (see page 146)	This is macro SaveCardToToXMLStreamEx.
SavePersonCsvToStream (see page 147)	This is macro SavePersonCsvToStream.

1.4.9 System.h

Windows related routines

Functions

	Name	Description
◆	AddRemoveMessageFilter (see page 10)	Adds or removes a message from the User Interface Privilege Isolation (UIPI) message filter.
◆	AllocateDefaultHWNDAs (see page 11)	This function creates the invisible tool window
◆	AllocateDefaultHWNDA (see page 11)	This function creates the invisible tool window
◆	AllocateHWNDA (see page 12)	This function creates the invisible tool window using the provided window procedure
◆	AllocateHWNDA (see page 12)	This function creates the invisible tool window using the provided window procedure
◆	AllocateLayeredWindowA (see page 12)	This function creates the layered window using the provided window class name
◆	AllocateLayeredWindowW (see page 13)	This function creates the layered window using the provided window class name
◆	AllocateWindowClassA (see page 13)	This function creates the standard window using the provided window class name
◆	AllocateWindowClassW (see page 13)	This function creates the standard window using the provided window class name
◆	BringWindowToFront (see page 14)	This function brings the specified window to the top of the z-order.
◆	ClearUnusedMemory (see page 19)	Clears unused memory and minimized the application memory usage
◆	DeallocateHWNDA (see page 23)	This function destroys the specified window.
◆	DeallocateHWNDA (see page 23)	This function destroys the specified window.
◆	DrawLayeredWindow (see page 28)	Repaints the surface of the layered window
◆	EmptyRecycleBin (see page 30)	Empties the recycle bin
◆	fpreset (see page 44)	This is function fpreset.
◆	GetStartupA (see page 65)	Checks if the application is registered to run when Windows starts
◆	GetStartupW (see page 65)	Checks if the application is registered to run when Windows starts
◆	HibernateWindows (see page 67)	Hibernates Windows
◆	LayeredWndProcA (see page 79)	The default window procedure for the layered window
◆	LayeredWndProcW (see page 79)	The default window procedure for the layered window
◆	MakeSoundFromFileA (see page 83)	Plays the wave sound from the file
◆	MakeSoundFromFileW (see page 83)	Plays the wave sound from the file
◆	MakeSoundFromResourceA (see page 84)	Plays the wave sound from the resource

◆	MakeSoundFromResourceW (see page 84)	Plays the wave sound from the resource
◆	RemoveStartupA (see page 97)	Removes the application from the list of the automatically started applications
◆	RemoveStartupW (see page 97)	Removes the application from the list of the automatically started applications
◆	RestoreWindowSubclassA (see page 97)	Restores window standard procedure
◆	RestoreWindowSubclassW (see page 98)	Restores window standard procedure
◆	SetStartupA (see page 114)	Register application to run when Windows starts
◆	SetStartupW (see page 115)	Register application to run when Windows starts
◆	ShutdownWindows (see page 116)	Logs off the interactive user, shuts down the system.
◆	SuspendWindows (see page 118)	Suspends Windows
◆	TurnMonitorOff (see page 119)	Turns the monitor off
◆	TurnMonitorOn (see page 119)	Turns the monitor on
◆	UpdateWindowPosition (see page 119)	Updated the window position

1.4.10 SystemInfo.h

Routines for querying information about operating system

Functions

	Name	Description
◆	CurrentIPAddressA (see page 21)	Returns the IP address
◆	CurrentIPAddressW (see page 22)	Returns the IP address
◆	IsCitrixSession (see page 70)	Checks if application is running in Citrix session
◆	IsConnectedToInternet (see page 70)	Checks if PC is connected to Internet
◆	IsMediaCenter (see page 73)	Checks if the Media Center version of Windows is installed
◆	IsMetroActive (see page 73)	Checks if metro interface is active
◆	IsMultiTouchReady (see page 74)	Checks if the system is multi touch ready
◆	IsNativeWin64 (see page 74)	Checks if the application is native 64 bit executable
◆	IsRemoteSession (see page 74)	Checks if application is running in RDP session
◆	IsTabletPC (see page 75)	Checks if the application is running on the Tablet PC
◆	IsWindows10 (see page 78)	Checks if PC is running Windows 10 or better
◆	IsWindows7 (see page 78)	Checks if PC is running Windows 7 or better
◆	IsWindows8 (see page 78)	Checks if PC is Running Windows 8 or better
◆	IsWindowsVista (see page 78)	Checks if PC is running Windows Vista or better
◆	IsWindowsXP (see page 79)	Checks if PC is running Windows XP
◆	IsWindowsXPSP2 (see page 79)	Checks if PC is running Windows XP with Service Pack 2 installed
◆	IsWow64 (see page 79)	Checks if the 32 bit application runs on 64 bit Windows
◆	PortAvailable (see page 85)	Checks if the port with specified number is available
◆	RecycleBinEmpty (see page 96)	Returns TRUE if Windows Recycle Bin is empty

Index

A

ActivateCard 9
ActivateCard function 9
ActivateCardEx 10
ActivateCardEx function 10
AddRemoveMessageFilter 10
AddRemoveMessageFilter function 10
AllocateBuffer 11
AllocateBuffer function 11
AllocateDefaultHWND 11
AllocateDefaultHWND function 11
AllocateDefaultHWNDW 11
AllocateDefaultHWNDW function 11
AllocateHWND 12
AllocateHWND function 12
AllocateHWNDW 12
AllocateHWNDW function 12
AllocateLayeredWindowA 12
AllocateLayeredWindowA function 12
AllocateLayeredWindowW 13
AllocateLayeredWindowW function 13
AllocateWindowClassA 13
AllocateWindowClassA function 13
AllocateWindowClassW 13
AllocateWindowClassW function 13
AlphaBlendBitmap 14
AlphaBlendBitmap function 14
AlphaBlendNative 14
AlphaBlendNative function 14

B

BringWindowToFront 14
BringWindowToFront function 14

C

CardDecryptFileA 14
CardDecryptFileA function 14
CardDecryptFileW 15
CardDecryptFileW function 15

CardEncryptFileA 15
CardEncryptFileA function 15
CardEncryptFileW 15
CardEncryptFileW function 15
CardEvents.h 150
CardEventType 128
CardEventType enumeration 128
CardSignCadesT 16
CardSignCadesT function 16
CardSignCMS 16
CardSignCMS function 16
CardStructures.h 150
CertSignCadesT 17
CertSignCadesT function 17
CertSignCMS 17
CertSignCMS function 17
CheckMD5 17
CheckMD5 function 17
CheckSHA1 18
CheckSHA1 function 18
CheckSHA256 18
CheckSHA256 function 18
ClearFileAttributesA 19
ClearFileAttributesA function 19
ClearFileAttributesW 19
ClearFileAttributesW function 19
ClearUnusedMemory 19
ClearUnusedMemory function 19
CloneFont 20
CloneFont function 20
CopyNativeBitmap 20
CopyNativeBitmap function 20
CreateCardBuffer 20
CreateCardBuffer function 20
CreateNativeBitmap 20
CreateNativeBitmap function 20
CreateUnicodeFileA 21
CreateUnicodeFileA function 21
CreateUnicodeFileW 21
CreateUnicodeFileW function 21
CreateWindowsFont 21
CreateWindowsFont function 21

CurrentIPAddressA 21
CurrentIPAddressA function 21
CurrentIPAddressW 22
CurrentIPAddressW function 22

D

DeactivateCard 22
DeactivateCard function 22
DeactivateCardEx 22
DeactivateCardEx function 22
DeallocateBuffer 23
DeallocateBuffer function 23
DeallocateHWND 23
DeallocateHWND function 23
DeallocateHWNDA 23
DeallocateHWNDA function 23
DeallocateHWNDAW 23
DeallocateHWNDAW function 23
DecryptFileAESA 24
DecryptFileAESA function 24
DecryptFileAESW 24
DecryptFileAESW function 24
DeleteCardBuffer 25
DeleteCardBuffer function 25
DeleteToRecycleBinA 25
DeleteToRecycleBinA function 25
DeleteToRecycleBinW 25
DeleteToRecycleBinW function 25
DestroyFont 26
DestroyFont function 26
DestroyImageBuffer 26
DestroyImageBuffer function 26
DirectoryExistsA 26
DirectoryExistsA function 26
DirectoryExistsW 27
DirectoryExistsW function 27
DisplayCertificate 27
DisplayCertificate function 27
DpiY 27
DpiY function 27
DrawAlphaText 28
DrawAlphaText function 28
DrawAlphaTextRect 28
DrawAlphaTextRect function 28

DrawLayeredWindow 28
DrawLayeredWindow function 28
DrawNativeBitmap 29
DrawNativeBitmap function 29
DrawTextDirect 29
DrawTextDirect function 29
DrawTextDirectEx 29
DrawTextDirectEx function 29
DrawTextGlow 29
DrawTextGlow function 29
DrawTextLine 30
DrawTextLine function 30
DrawTextOutline 30
DrawTextOutline function 30
DrawTextRect 30
DrawTextRect function 30

E

EID_MAX_BIRTHDATE_LEN 140
EID_MAX_BIRTHDATE_LEN macro 140
EID_MAX_BIRTHPLACE_LEN 140
EID_MAX_BIRTHPLACE_LEN macro 140
EID_MAX_CARD_NUMBER_LEN 140
EID_MAX_CARD_NUMBER_LEN macro 140
EID_MAX_CERT_LEN 141
EID_MAX_CERT_LEN macro 141
EID_MAX_CHIP_NUMBER_LEN 141
EID_MAX_CHIP_NUMBER_LEN macro 141
EID_MAX_DATE_BEGIN_LEN 141
EID_MAX_DATE_BEGIN_LEN macro 141
EID_MAX_DATE_END_LEN 141
EID_MAX_DATE_END_LEN macro 141
EID_MAX_DELIVERY_MUNICIPALITY_LEN 142
EID_MAX_DELIVERY_MUNICIPALITY_LEN macro 142
EID_MAX_DOCUMENT_TYPE_LEN 142
EID_MAX_DOCUMENT_TYPE_LEN macro 142
EID_MAX_FIRST_NAME1_LEN 142
EID_MAX_FIRST_NAME1_LEN macro 142
EID_MAX_FIRST_NAME2_LEN 142
EID_MAX_FIRST_NAME2_LEN macro 142
EID_MAX_MUNICIPALITY_LEN 142
EID_MAX_MUNICIPALITY_LEN macro 142

EID_MAX_NAME_LEN 143
EID_MAX_NAME_LEN macro 143
EID_MAX_NATIONAL_NUMBER_LEN 143
EID_MAX_NATIONAL_NUMBER_LEN macro 143
EID_MAX_NATIONALITY_LEN 143
EID_MAX_NATIONALITY_LEN macro 143
EID_MAX_NOBLE_CONDITION_LEN 143
EID_MAX_NOBLE_CONDITION_LEN macro 143
EID_MAX_PICTURE_LEN 144
EID_MAX_PICTURE_LEN macro 144
EID_MAX_SEX_LEN 144
EID_MAX_SEX_LEN macro 144
EID_MAX_SPECIAL_STATUS_LEN 144
EID_MAX_SPECIAL_STATUS_LEN macro 144
EID_MAX_STREET_LEN 144
EID_MAX_STREET_LEN macro 144
EID_MAX_ZIP_LEN 144
EID_MAX_ZIP_LEN macro 144
EidAddressA 129
EidAddressA structure 129
EidAddressW 129
EidAddressW structure 129
EidCertificate 130
EidCertificate structure 130
EidIdentityA 130
EidIdentityA structure 130
EidIdentityW 131
EidIdentityW structure 131
EidPicture 132
EidPicture structure 132
EmptyRecycleBin 30
EmptyRecycleBin function 30
EmToPixels 31
EmToPixels function 31
EncodeCertificate 31
EncodeCertificate function 31
EncodePhoto 31
EncodePhoto function 31
EncryptFileAESA 32
EncryptFileAESA function 32
EncryptFileAESW 32
EncryptFileAESW function 32

Encryption.h 152
ewtCardInsert enumeration member 123
ewtCardRemove enumeration member 123
ewtReadersChange enumeration member 123
ewtUnknownEvent enumeration member 123

F

FileCloseA 33
FileCloseA function 33
FileCloseW 33
FileCloseW function 33
FileCopyA 33
FileCopyA function 33
FileCopyW 34
FileCopyW function 34
FileCreateRewriteA 34
FileCreateRewriteA function 34
FileCreateRewriteW 34
FileCreateRewriteW function 34
FileDeleteA 35
FileDeleteA function 35
FileDeleteW 35
FileDeleteW function 35
FileExistsA 36
FileExistsA function 36
FileExistsW 36
FileExistsW function 36
FileExtensionIsA 36
FileExtensionIsA function 36
FileExtensionIsW 37
FileExtensionIsW function 37
FileGetSizeA 37
FileGetSizeA function 37
FileGetSizeW 37
FileGetSizeW function 37
FileIsExeA 38
FileIsExeA function 38
FileIsExeW 38
FileIsExeW function 38
FileIsIconA 39
FileIsIconA function 39
FileIsIconW 39

FileIsIconW function 39
FileIsImageA 39
FileIsImageA function 39
FileIsImageW 40
FileIsImageW function 40
FileIsLink 40
FileIsLink function 40
FileOperations.h 153
FileOrFolderExistsA 40
FileOrFolderExistsA function 40
FileOrFolderExistsW 41
FileOrFolderExistsW function 41
FileRenameA 41
FileRenameA function 41
FileRenameW 41
FileRenameW function 41
Files 150
FileWriteA 42
FileWriteA function 42
FileWriteCharA 42
FileWriteCharA function 42
FileWriteCharW 42
FileWriteCharW function 42
FileWriteNewLineA 43
FileWriteNewLineA function 43
FileWriteNewLineW 43
FileWriteNewLineW function 43
FileWriteW 43
FileWriteW function 43
FONT_BOLD 145
FONT_BOLD macro 145
FONT_ITALIC 145
FONT_ITALIC macro 145
FONT_NORMAL 145
FONT_NORMAL macro 145
FONT_STRIKEOUT 145
FONT_STRIKEOUT macro 145
FONT_UNDERLINE 146
FONT_UNDERLINE macro 146
fpreset 44
fpreset function 44
FullPathA 44

FullPathA function 44
FullPathW 44
FullPathW function 44
Functions 1

G

GenerateAuthenticationSignatureA 44
GenerateAuthenticationSignatureA function 44
GenerateAuthenticationSignatureExA 45
GenerateAuthenticationSignatureExA function 45
GenerateAuthenticationSignatureExW 45
GenerateAuthenticationSignatureExW function 45
GenerateAuthenticationSignatureW 46
GenerateAuthenticationSignatureW function 46
GenerateBMPA 46
GenerateBMPA function 46
GenerateBMPW 47
GenerateBMPW function 47
GenerateNonRepudiationSignatureA 47
GenerateNonRepudiationSignatureA function 47
GenerateNonRepudiationSignatureExA 48
GenerateNonRepudiationSignatureExA function 48
GenerateNonRepudiationSignatureExW 48
GenerateNonRepudiationSignatureExW function 48
GenerateNonRepudiationSignatureW 49
GenerateNonRepudiationSignatureW function 49
GeneratePNGA 49
GeneratePNGA function 49
GeneratePNGW 50
GeneratePNGW function 50
GenerateQRCodeA 50
GenerateQRCodeA function 50
GenerateQRCodeExA 51
GenerateQRCodeExA function 51
GenerateQRCodeExW 51
GenerateQRCodeExW function 51
GenerateQRCodeW 51
GenerateQRCodeW function 51
GetAllFiles 52
GetAllFiles function 52
GetCardBufferA 52
GetCardBufferA function 52

GetCardBufferSize 52	GetPNGW 61
GetCardBufferSize function 52	GetPNGW function 61
GetCardBufferW 53	GetReaderIndexA 61
GetCardBufferW function 53	GetReaderIndexA function 61
GetCardSerialNumber 53	GetReaderIndexW 61
GetCardSerialNumber function 53	GetReaderIndexW function 61
GetCardVersion 53	GetReaderNameA 62
GetCardVersion function 53	GetReaderNameA function 62
GetEncodedCertificateSize 54	GetReaderNameLenA 62
GetEncodedCertificateSize function 54	GetReaderNameLenA function 62
GetEncodedPhotoSize 54	GetReaderNameLenW 63
GetEncodedPhotoSize function 54	GetReaderNameLenW function 63
GetFileMD5A 54	GetReaderNameW 63
GetFileMD5A function 54	GetReaderNameW function 63
GetFileMD5W 55	GetReadersCount 63
GetFileMD5W function 55	GetReadersCount function 63
GetFilesCountA 55	GetSelectedReaderIndex 64
GetFilesCountA function 55	GetSelectedReaderIndex function 64
GetFilesCountW 56	GetSHA1 64
GetFilesCountW function 56	GetSHA1 function 64
GetFileSHA1A 56	GetSHA256 64
GetFileSHA1A function 56	GetSHA256 function 64
GetFileSHA1W 56	GetStartupA 65
GetFileSHA1W function 56	GetStartupA function 65
GetFileSHA256 146	GetStartupW 65
GetFileSHA256 macro 146	GetStartupW function 65
GetFileSHA256A 57	GetSupportSIS 66
GetFileSHA256A function 57	GetSupportSIS function 66
GetFileSHA256W 57	GetTextLineSize 66
GetFileSHA256W function 57	GetTextLineSize function 66
GetHBitmapA 58	GetTextSize 66
GetHBitmapA function 58	GetTextSize function 66
GetHBitmapW 58	GetTextSizeEx 66
GetHBitmapW function 58	GetTextSizeEx function 66
GetISOCODEA 59	Graphics.h 155
GetISOCODEA function 59	
GetISOCODEW 59	H
GetISOCODEW function 59	HibernateWindows 67
GetMD5 60	HibernateWindows function 67
GetMD5 function 60	
GetPNGA 60	I
GetPNGA function 60	IID_PPV_ARG 146

IID_PPV_ARG macro 146	IsMetroActive function 73
IsAnimatedGIFA 67	IsMultiTouchReady 74
IsAnimatedGIFA function 67	IsMultiTouchReady function 74
IsAnimatedGIFW 67	IsNativeWin64 74
IsAnimatedGIFW function 67	IsNativeWin64 function 74
IsCardActivated 68	IsRemoteSession 74
IsCardActivated function 68	IsRemoteSession function 74
IsCardActivatedEx 68	IsSISCard 74
IsCardActivatedEx function 68	IsSISCard function 74
IsCardPresent 68	IsSISCardEx 75
IsCardPresent function 68	IsSISCardEx function 75
IsCardPresentEx 69	IsTabletPC 75
IsCardPresentEx function 69	IsTabletPC function 75
IsCardStillInserted 69	IsUnicodeFileA 75
IsCardStillInserted function 69	IsUnicodeFileA function 75
IsCardStillInsertedEx 69	IsUnicodeFileW 76
IsCardStillInsertedEx function 69	IsUnicodeFileW function 76
IsCitrixSession 70	IsValidFileNameA 76
IsCitrixSession function 70	IsValidFileNameA function 76
IsConnectedToInternet 70	IsValidFileNameW 76
IsConnectedToInternet function 70	IsValidFileNameW function 76
IsDirectoryA 70	IsValidPathNameA 77
IsDirectoryA function 70	IsValidPathNameA function 77
IsDirectoryW 70	IsValidPathNameW 77
IsDirectoryW function 70	IsValidPathNameW function 77
IsEIDCard 71	IsWindows10 78
IsEIDCard function 71	IsWindows10 function 78
IsEIDCardEx 71	IsWindows7 78
IsEIDCardEx function 71	IsWindows7 function 78
IsEngineActive 71	IsWindows8 78
IsEngineActive function 71	IsWindows8 function 78
IsFemaleA 72	IsWindowsVista 78
IsFemaleA function 72	IsWindowsVista function 78
IsFemaleW 72	IsWindowsXP 79
IsFemaleW function 72	IsWindowsXP function 79
IsMaleA 72	IsWindowsXPSP2 79
IsMaleA function 72	IsWindowsXPSP2 function 79
IsMaleW 73	IsWow64 79
IsMaleW function 73	IsWow64 function 79
IsMediaCenter 73	
IsMediaCenter function 73	
IsMetroActive 73	
	L
	LayeredWndProcA 79

LayeredWndProcA function 79
 LayeredWndProcW 79
 LayeredWndProcW function 79
 LoadBitmapJPG 80
 LoadBitmapJPG function 80
 LoadBitmapPNG 80
 LoadBitmapPNG function 80
 LoadCertificateA 80
 LoadCertificateA function 80
 LoadCertificateW 81
 LoadCertificateW function 81
 LoadIdentityA 81
 LoadIdentityA function 81
 LoadIdentityW 81
 LoadIdentityW function 81
 LoadPhotoA 82
 LoadPhotoA function 82
 LoadPhotoW 82
 LoadPhotoW function 82
 LoadPNGResource 82
 LoadPNGResource function 82

M

Macros 139
 MakeCompatibleBitmap 83
 MakeCompatibleBitmap function 83
 MakeSoundFromFileA 83
 MakeSoundFromFileA function 83
 MakeSoundFromFileW 83
 MakeSoundFromFileW function 83
 MakeSoundFromResourceA 84
 MakeSoundFromResourceA function 84
 MakeSoundFromResourceW 84
 MakeSoundFromResourceW function 84

N

NationalityConverter.h 156

P

PEidAddressA 132
 PEidAddressA structure 132
 PEidAddressW 133

PEidAddressW structure 133
 PEidCertificate 133
 PEidCertificate structure 133
 PEidIdentityA 133
 PEidIdentityA structure 133
 PEidIdentityW 134
 PEidIdentityW structure 134
 PeidPicture 135
 PeidPicture structure 135
 PointsToPixels 84
 PointsToPixels function 84
 PortAvailable 85
 PortAvailable function 85
 PSISRecordA 136
 PSISRecordA structure 136
 PSISRecordW 136
 PSISRecordW structure 136

Q

quricol.h 156

R

ReadAddressA 85
 ReadAddressA function 85
 ReadAddressExA 85
 ReadAddressExA function 85
 ReadAddressExW 86
 ReadAddressExW function 86
 ReadAddressW 86
 ReadAddressW function 86
 ReadAuthenticationCertificate 86
 ReadAuthenticationCertificate function 86
 ReadAuthenticationCertificateEx 87
 ReadAuthenticationCertificateEx function 87
 ReadBufferFromFileA 87
 ReadBufferFromFileA function 87
 ReadBufferFromFileW 87
 ReadBufferFromFileW function 87
 ReadCaCertificate 88
 ReadCaCertificate function 88
 ReadCaCertificateEx 88
 ReadCaCertificateEx function 88

ReadIdentityA 89
ReadIdentityA function 89
ReadIdentityExA 89
ReadIdentityExA function 89
ReadIdentityExW 89
ReadIdentityExW function 89
ReadIdentityW 90
ReadIdentityW function 90
ReadNonRepudiationCertificate 90
ReadNonRepudiationCertificate function 90
ReadNonRepudiationCertificateEx 91
ReadNonRepudiationCertificateEx function 91
ReadPhoto 91
ReadPhoto function 91
ReadPhotoAsBitmap 91
ReadPhotoAsBitmap function 91
ReadPhotoAsBitmapEx 92
ReadPhotoAsBitmapEx function 92
ReadPhotoEx 92
ReadPhotoEx function 92
ReadRootCaCertificate 93
ReadRootCaCertificate function 93
ReadRootCaCertificateEx 93
ReadRootCaCertificateEx function 93
ReadRrnCertificate 93
ReadRrnCertificate function 93
ReadRrnCertificateEx 94
ReadRrnCertificateEx function 94
ReadSISCardA 94
ReadSISCardA function 94
ReadSISCardExA 95
ReadSISCardExA function 95
ReadSISCardExW 95
ReadSISCardExW function 95
ReadSISCardW 95
ReadSISCardW function 95
RecycleBinEmpty 96
RecycleBinEmpty function 96
ReloadReadersList 96
ReloadReadersList function 96
RemoveCallback 96
RemoveCallback function 96

RemoveStartupA 97
RemoveStartupA function 97
RemoveStartupW 97
RemoveStartupW function 97
RestoreWindowSubclassA 97
RestoreWindowSubclassA function 97
RestoreWindowSubclassW 98
RestoreWindowSubclassW function 98

S

SaveAuthenticationCertificateA 98
SaveAuthenticationCertificateA function 98
SaveAuthenticationCertificateExW 98
SaveAuthenticationCertificateExW function 98
SaveAuthenticationCertificateW 99
SaveAuthenticationCertificateW function 99
SaveCaCertificateA 99
SaveCaCertificateA function 99
SaveCaCertificateExW 99
SaveCaCertificateExW function 99
SaveCaCertificateW 100
SaveCaCertificateW function 100
SaveCardToToXMLStreamEx 146
SaveCardToToXMLStreamEx macro 146
SaveCardToToXMLStreamExA 100
SaveCardToToXMLStreamExA function 100
SaveCardToToXMLStreamExW 100
SaveCardToToXMLStreamExW function 100
SaveCardToXmlA 101
SaveCardToXmlA function 101
SaveCardToXmlExA 101
SaveCardToXmlExA function 101
SaveCardToXmlExW 102
SaveCardToXmlExW function 102
SaveCardToXmlW 102
SaveCardToXmlW function 102
SaveIdentityA 103
SaveIdentityA function 103
SaveIdentityW 103
SaveIdentityW function 103
SaveNonRepudiationCertificateA 103
SaveNonRepudiationCertificateA function 103

SaveNonRepudiationCertificateExW 104	SaveRootCaCertificateW 111
SaveNonRepudiationCertificateExW function 104	SaveRootCaCertificateW function 111
SaveNonRepudiationCertificateW 104	SaveRrnCertificateA 111
SaveNonRepudiationCertificateW function 104	SaveRrnCertificateA function 111
SavePersonCsvToStream 147	SaveRrnCertificateExW 112
SavePersonCsvToStream macro 147	SaveRrnCertificateExW function 112
SavePersonCsvToStreamA 104	SaveRrnCertificateW 112
SavePersonCsvToStreamA function 104	SaveRrnCertificateW function 112
SavePersonCsvToStreamW 105	SelectReader 112
SavePersonCsvToStreamW function 105	SelectReader function 112
SavePersonToCsvA 105	SelectReaderByNameA 113
SavePersonToCsvA function 105	SelectReaderByNameA function 113
SavePersonToCsvExA 105	SelectReaderByNameW 113
SavePersonToCsvExA function 105	SelectReaderByNameW function 113
SavePersonToCsvExW 106	SendAPDU 114
SavePersonToCsvExW function 106	SendAPDU function 114
SavePersonToCsvW 106	SetCallback 114
SavePersonToCsvW function 106	SetCallback function 114
SavePhotoA 106	SetMWCompatibility 114
SavePhotoA function 106	SetMWCompatibility function 114
SavePhotoAsBitmapA 107	SetStartupA 114
SavePhotoAsBitmapA function 107	SetStartupA function 114
SavePhotoAsBitmapExA 107	SetStartupW 115
SavePhotoAsBitmapExA function 107	SetStartupW function 115
SavePhotoAsBitmapExW 108	SetSupportSIS 115
SavePhotoAsBitmapExW function 108	SetSupportSIS function 115
SavePhotoAsBitmapW 108	ShellCopyFileA 115
SavePhotoAsBitmapW function 108	ShellCopyFileA function 115
SavePhotoAsJpegA 108	ShellCopyFileW 116
SavePhotoAsJpegA function 108	ShellCopyFileW function 116
SavePhotoAsJpegExA 109	ShutdownWindows 116
SavePhotoAsJpegExA function 109	ShutdownWindows function 116
SavePhotoAsJpegExW 109	SIS_FIELD_MAX_BIRTHDATE_LEN 147
SavePhotoAsJpegExW function 109	SIS_FIELD_MAX_BIRTHDATE_LEN macro 147
SavePhotoAsJpegW 109	SIS_FIELD_MAX_CAPTUREDATE_LEN 147
SavePhotoAsJpegW function 109	SIS_FIELD_MAX_CAPTUREDATE_LEN macro 147
SavePhotoW 110	SIS_FIELD_MAX_CARDNUMBER_LEN 147
SavePhotoW function 110	SIS_FIELD_MAX_CARDNUMBER_LEN macro 147
SaveRootCaCertificateA 110	SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN 148
SaveRootCaCertificateA function 110	SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN macro 148
SaveRootCaCertificateExW 111	SIS_FIELD_MAX_VALIDBEGIN_LEN 148
SaveRootCaCertificateExW function 111	SIS_FIELD_MAX_VALIDBEGIN_LEN macro 148

SIS_FIELD_MAX_VALIDEND_LEN 148
SIS_FIELD_MAX_VALIDEND_LEN macro 148
SIS_MAX_CARDNAME_LEN 148
SIS_MAX_CARDNAME_LEN macro 148
SIS_MAX_FIRSTNAMES_LEN 149
SIS_MAX_FIRSTNAMES_LEN macro 149
SIS_MAX_INITIAL_LEN 149
SIS_MAX_INITIAL_LEN macro 149
SIS_MAX_NAME_LEN 149
SIS_MAX_NAME_LEN macro 149
SIS_MAX_SEX_LEN 149
SIS_MAX_SEX_LEN macro 149
SISRecordA 137
SISRecordA structure 137
SISRecordW 138
SISRecordW structure 138
StartEngine 117
StartEngine function 117
StopEngine 117
StopEngine function 117
StretchNativeBitmap 117
StretchNativeBitmap function 117
StripFileNameA 118
StripFileNameA function 118
StripFileNameW 118
StripFileNameW function 118
Structs, Records, Enums 122
SuspendWindows 118
SuspendWindows function 118
Swelio.h 156
System.h 161
SystemInfo.h 162

T

tagCardEventType 123
tagCardEventType enumeration 123
tagEidAddressA 123
tagEidAddressA structure 123
tagEidAddressW 124
tagEidAddressW structure 124
tagEidCertificate 124
tagEidCertificate structure 124

tagEidIdentityA 125
tagEidIdentityA structure 125
tagEidIdentityW 126
tagEidIdentityW structure 126
tagEidPicture 127
tagEidPicture structure 127
tagSISRecordA 127
tagSISRecordA structure 127
tagSISRecordW 128
tagSISRecordW structure 128
TurnMonitorOff 119
TurnMonitorOff function 119
TurnMonitorOn 119
TurnMonitorOn function 119

U

UpdateWindowPosition 119
UpdateWindowPosition function 119

V

VerifyPinA 119
VerifyPinA function 119
VerifyPinExA 120
VerifyPinExA function 120
VerifyPinExW 120
VerifyPinExW function 120
VerifyPinW 120
VerifyPinW function 120
VerifySignature 121
VerifySignature function 121

W

WIDTHBYTES 149
WIDTHBYTES macro 149
WriteBufferToFileA 121
WriteBufferToFileA function 121
WriteBufferToFileW 122
WriteBufferToFileW function 122