

Photon Fusion

2.0.2

1 Photon Fusion API Documentation	1
1.1 Main Fusion API	1
2 Photon Fusion Overview	3
2.1 Choosing the Right Mode	4
2.2 Topology Differences	5
2.2.1 Server Mode	5
2.2.1.1 Client Side Prediction	5
2.2.2 Host Mode	6
2.2.3 Shared Mode	6
2.2.4 Cost	6
3 Hierarchical Index	7
3.1 Class Hierarchy	7
4 Class Index	15
4.1 Class List	15
5 Namespace Documentation	31
5.1 Fusion Namespace Reference	31
5.1.1 Enumeration Type Documentation	44
5.1.1.1 CompareOperator	44
5.1.1.2 ConnectionType	45
5.1.1.3 DrawIfMode	45
5.1.1.4 EditorButtonVisibility	45
5.1.1.5 GameMode	45
5.1.1.6 HitboxTypes	46
5.1.1.7 HitOptions	46
5.1.1.8 NetworkObjectAcquireResult	47
5.1.1.9 NetworkObjectFlags	47
5.1.1.10 NetworkObjectHeaderFlags	47
5.1.1.11 NetworkPrefabTableGetPrefabResult	48
5.1.1.12 NetworkSceneInfoChangeSource	48
5.1.1.13 NetworkSceneInfoDefaultFlags	48
5.1.1.14 NetworkSpawnFlags	49
5.1.1.15 NetworkSpawnStatus	49
5.1.1.16 NetworkTypeIdKind	49
5.1.1.17 PageSizes	50
5.1.1.18 PriorityLevel	50
5.1.1.19 RenderSource	50
5.1.1.20 RenderTimeframe	51
5.1.1.21 RpcChannel	51
5.1.1.22 RpcHostMode	51
5.1.1.23 RpcLocallInvokeResult	51

5.1.1.24 RpcSendCullResult	52
5.1.1.25 RpcSendMessageResult	52
5.1.1.26 RpcSources	53
5.1.1.27 RpcTargets	53
5.1.1.28 RpcTargetStatus	54
5.1.1.29 ScriptHeaderBackColor	54
5.1.1.30 ScriptHeaderIcon	54
5.1.1.31 ScriptHeaderStyle	54
5.1.1.32 SessionLobby	55
5.1.1.33 ShutdownReason	55
5.1.1.34 SimulationModes	56
5.1.1.35 SimulationStages	56
5.1.1.36 Topologies	56
5.1.1.37 Units	56
5.1.1.38 UnityPlayerLoopSystemAddMode	57
5.1.2 Function Documentation	57
5.1.2.1 FusionGlobalScriptableObjectUnloadDelegate()	58
5.1.2.2 NetworkObjectSpawnDelegate()	58
5.1.2.3 RpcInvokeDelegate()	58
5.1.2.4 RpcStaticInvokeDelegate()	58
5.2 Fusion.Analyzer Namespace Reference	59
5.3 Fusion.Async Namespace Reference	59
5.4 Fusion.Encryption Namespace Reference	59
5.5 Fusion.Internal Namespace Reference	59
5.6 Fusion.LagCompensation Namespace Reference	60
5.6.1 Enumeration Type Documentation	61
5.6.1.1 HitType	61
5.6.2 Function Documentation	61
5.6.2.1 PreProcessingDelegate()	61
5.7 Fusion.Protocol Namespace Reference	62
5.7.1 Enumeration Type Documentation	62
5.7.1.1 DisconnectReason	62
5.8 Fusion.Runtime Namespace Reference	62
5.9 Fusion.Runtime.Unity Namespace Reference	62
5.10 Fusion.Sockets Namespace Reference	62
5.10.1 Enumeration Type Documentation	63
5.10.1.1 NetCommands	63
5.10.1.2 NetConnectFailedReason	63
5.10.1.3 NetDisconnectReason	64
5.10.1.4 NetPacketType	64
5.11 Fusion.Sockets.Stun Namespace Reference	64
5.11.1 Enumeration Type Documentation	64

5.11.1.1 NATType	64
5.12 Fusion.Statistics Namespace Reference	65
5.13 UnityEngine Namespace Reference	65
5.14 UnityEngine.SceneManagement Namespace Reference	65
5.15 UnityEngine.Scripting Namespace Reference	65
5.16 UnityEngine.Serialization Namespace Reference	65
6 Class Documentation	67
6.1 _128 Struct Reference	67
6.1.1 Detailed Description	67
6.1.2 Member Data Documentation	67
6.1.2.1 Data	67
6.1.2.2 SIZE	68
6.2 _16 Struct Reference	68
6.2.1 Detailed Description	68
6.2.2 Member Data Documentation	68
6.2.2.1 Data	68
6.2.2.2 SIZE	68
6.3 _2 Struct Reference	69
6.3.1 Detailed Description	69
6.3.2 Member Data Documentation	69
6.3.2.1 Data	69
6.3.2.2 SIZE	69
6.4 _256 Struct Reference	69
6.4.1 Detailed Description	70
6.4.2 Member Data Documentation	70
6.4.2.1 Data	70
6.4.2.2 SIZE	70
6.5 _32 Struct Reference	70
6.5.1 Detailed Description	71
6.5.2 Member Data Documentation	71
6.5.2.1 Data	71
6.5.2.2 SIZE	71
6.6 _4 Struct Reference	71
6.6.1 Detailed Description	72
6.6.2 Member Data Documentation	72
6.6.2.1 Data	72
6.6.2.2 SIZE	72
6.7 _512 Struct Reference	72
6.7.1 Detailed Description	72
6.7.2 Member Data Documentation	73
6.7.2.1 Data	73

6.7.2.2 SIZE	73
6.8 _64 Struct Reference	73
6.8.1 Detailed Description	73
6.8.2 Member Data Documentation	73
6.8.2.1 Data	74
6.8.2.2 SIZE	74
6.9 _8 Struct Reference	74
6.9.1 Detailed Description	74
6.9.2 Member Data Documentation	74
6.9.2.1 Data	74
6.9.2.2 SIZE	75
6.10 Allocator Struct Reference	75
6.10.1 Detailed Description	76
6.10.2 Member Data Documentation	76
6.10.2.1 BUCKET_COUNT	76
6.10.2.2 BUCKET_INVALID	76
6.10.2.3 HEAP_ALIGNMENT	76
6.10.2.4 REPLICATE_WORD_ALIGN	77
6.10.2.5 REPLICATE_WORD_SHIFT	77
6.10.2.6 REPLICATE_WORD_SIZE	77
6.11 Allocator.Config Struct Reference	77
6.11.1 Detailed Description	78
6.11.2 Constructor & Destructor Documentation	78
6.11.2.1 Config()	78
6.11.3 Member Function Documentation	78
6.11.3.1 Equals() [1/2]	79
6.11.3.2 Equals() [2/2]	79
6.11.3.3 GetHashCode()	79
6.11.3.4 ToString()	80
6.11.4 Member Data Documentation	80
6.11.4.1 BlockCount	80
6.11.4.2 BlockShift	80
6.11.4.3 DEFAULT_BLOCK_COUNT	80
6.11.4.4 DEFAULT_BLOCK_SHIFT	80
6.11.4.5 GlobalsSize	80
6.11.4.6 SIZE	81
6.11.5 Property Documentation	81
6.11.5.1 BlockByteSize	81
6.11.5.2 BlockWordCount	81
6.11.5.3 HeapSizeAllocated	81
6.11.5.4 HeapSizeUsable	81
6.12 Angle Struct Reference	81

6.12.1 Detailed Description	83
6.12.2 Member Function Documentation	83
6.12.2.1 Clamp() [1/2]	83
6.12.2.2 Clamp() [2/2]	83
6.12.2.3 Equals() [1/2]	83
6.12.2.4 Equals() [2/2]	84
6.12.2.5 GetHashCode()	84
6.12.2.6 Lerp()	84
6.12.2.7 Max()	85
6.12.2.8 Min()	85
6.12.2.9 operator Angle() [1/3]	85
6.12.2.10 operator Angle() [2/3]	85
6.12.2.11 operator Angle() [3/3]	86
6.12.2.12 operator double()	86
6.12.2.13 operator float()	86
6.12.2.14 operator"!="()	87
6.12.2.15 operator+()	87
6.12.2.16 operator-()	88
6.12.2.17 operator<()	88
6.12.2.18 operator<=()	88
6.12.2.19 operator==()	89
6.12.2.20 operator>()	89
6.12.2.21 operator>=()	90
6.12.2.22 ToString()	90
6.12.3 Member Data Documentation	90
6.12.3.1 SIZE	90
6.13 ArrayLengthAttribute Class Reference	90
6.13.1 Detailed Description	91
6.13.2 Constructor & Destructor Documentation	91
6.13.2.1 ArrayLengthAttribute() [1/2]	91
6.13.2.2 ArrayLengthAttribute() [2/2]	91
6.13.3 Property Documentation	92
6.13.3.1 MaxLength	92
6.13.3.2 MinLength	92
6.14 AssemblyNameAttribute Class Reference	92
6.14.1 Detailed Description	92
6.14.2 Property Documentation	92
6.14.2.1 RequiresUnsafeCode	93
6.15 AssetObject Class Reference	93
6.15.1 Detailed Description	93
6.16 TaskManager Class Reference	93
6.16.1 Detailed Description	93

6.16.2 Member Function Documentation	93
6.16.2.1 ContinueWhenAll()	93
6.16.2.2 Run()	94
6.16.2.3 Service()	94
6.16.2.4 Setup()	95
6.17 AuthorityMasks Class Reference	95
6.17.1 Detailed Description	95
6.17.2 Member Data Documentation	95
6.17.2.1 ALL	95
6.17.2.2 INPUT	96
6.17.2.3 NONE	96
6.17.2.4 PROXY	96
6.17.2.5 STATE	96
6.18 Behaviour Class Reference	96
6.18.1 Detailed Description	97
6.18.2 Member Function Documentation	97
6.18.2.1 AddBehaviour< T >()	97
6.18.2.2 DestroyBehaviour()	97
6.18.2.3 GetBehaviour< T >()	97
6.18.2.4 TryGetBehaviour< T >()	98
6.19 BinaryDataAttribute Class Reference	98
6.19.1 Detailed Description	98
6.20 BitSetAttribute Class Reference	98
6.20.1 Detailed Description	98
6.20.2 Constructor & Destructor Documentation	98
6.20.2.1 BitSetAttribute()	98
6.20.3 Property Documentation	99
6.20.3.1 BitCount	99
6.21 CapacityAttribute Class Reference	99
6.21.1 Detailed Description	99
6.21.2 Constructor & Destructor Documentation	99
6.21.2.1 CapacityAttribute()	99
6.21.3 Property Documentation	100
6.21.3.1 Length	100
6.22 DecoratingPropertyAttribute Class Reference	100
6.22.1 Detailed Description	100
6.22.2 Constructor & Destructor Documentation	100
6.22.2.1 DecoratingPropertyAttribute() [1/2]	101
6.22.2.2 DecoratingPropertyAttribute() [2/2]	101
6.22.3 Member Data Documentation	101
6.22.3.1 DefaultOrder	101
6.23 DefaultForPropertyAttribute Class Reference	101

6.23.1 Detailed Description	102
6.23.2 Constructor & Destructor Documentation	102
6.23.2.1 DefaultForPropertyAttribute()	102
6.23.3 Property Documentation	102
6.23.3.1PropertyName	102
6.23.3.2 WordCount	102
6.23.3.3 WordOffset	103
6.24 DisplayAsEnumAttribute Class Reference	103
6.24.1 Detailed Description	103
6.24.2 Constructor & Destructor Documentation	103
6.24.2.1 DisplayAsEnumAttribute() [1/2]	103
6.24.2.2 DisplayAsEnumAttribute() [2/2]	104
6.24.3 Property Documentation	104
6.24.3.1 EnumType	104
6.24.3.2 EnumTypeMemberName	104
6.25 DisplayNameAttribute Class Reference	104
6.25.1 Detailed Description	105
6.25.2 Constructor & Destructor Documentation	105
6.25.2.1 DisplayNameAttribute()	105
6.25.3 Member Data Documentation	105
6.25.3.1 Name	105
6.26 DolfAttributeBase Class Reference	105
6.26.1 Detailed Description	106
6.26.2 Constructor & Destructor Documentation	106
6.26.2.1 DolfAttributeBase() [1/3]	106
6.26.2.2 DolfAttributeBase() [2/3]	107
6.26.2.3 DolfAttributeBase() [3/3]	107
6.26.3 Member Data Documentation	107
6.26.3.1 _doubleValue	107
6.26.3.2 _isDouble	108
6.26.3.3 _longValue	108
6.26.3.4 Compare	108
6.26.3.5 ConditionMember	108
6.26.3.6 ErrorOnConditionMemberNotFound	108
6.27 DrawerPropertyAttribute Class Reference	108
6.27.1 Detailed Description	109
6.28 DrawIfAttribute Class Reference	109
6.28.1 Detailed Description	109
6.28.2 Constructor & Destructor Documentation	109
6.28.2.1 DrawIfAttribute()	109
6.28.3 Member Data Documentation	110
6.28.3.1 Mode	110

6.28.4 Property Documentation	110
6.28.4.1 Hide	110
6.29 DrawInlineAttribute Class Reference	110
6.29.1 Detailed Description	110
6.30 DynamicHeap Struct Reference	110
6.30.1 Detailed Description	112
6.30.2 Member Function Documentation	112
6.30.2.1 CollectGarbage()	112
6.30.2.2 CollectGarbageDelegate()	113
6.30.2.3 Free()	113
6.30.2.4 SetForcedAlive< T >()	113
6.30.3 Member Data Documentation	114
6.30.3.1 _debruijnTable	114
6.31 DynamicHeap.Ignore Class Reference	114
6.31.1 Detailed Description	114
6.32 DynamicHeapInstance Class Reference	114
6.32.1 Detailed Description	115
6.32.2 Constructor & Destructor Documentation	115
6.32.2.1 DynamicHeapInstance()	115
6.32.3 Member Function Documentation	115
6.32.3.1 Allocate()	115
6.32.3.2 AllocateArray< T >()	115
6.32.3.3 AllocateArrayPointers< T >()	116
6.32.3.4 AllocateTracked< T >()	116
6.32.3.5 AllocateTrackedArray< T >()	117
6.32.3.6 AllocateTrackedArrayPointers< T >()	118
6.32.3.7 Free()	119
6.33 EditorButtonAttribute Class Reference	119
6.33.1 Detailed Description	120
6.33.2 Constructor & Destructor Documentation	120
6.33.2.1 EditorButtonAttribute() [1/2]	120
6.33.2.2 EditorButtonAttribute() [2/2]	120
6.33.3 Member Data Documentation	122
6.33.3.1 AllowMultipleTargets	122
6.33.3.2 DirtyObject	122
6.33.3.3 Label	122
6.33.3.4 Priority	122
6.33.3.5 Visibility	122
6.34 DataEncryptor Class Reference	123
6.34.1 Detailed Description	123
6.34.2 Member Function Documentation	123
6.34.2.1 EncryptData()	123

6.35 IDataEncryption Interface Reference	124
6.35.1 Detailed Description	124
6.35.2 Member Function Documentation	124
6.35.2.1 ComputeHash()	124
6.35.2.2 DecryptData()	125
6.35.2.3 EncryptData()	125
6.35.2.4 GenerateKey()	126
6.35.2.5 Setup()	126
6.35.2.6 VerifyHash()	126
6.36 ErrorIfAttribute Class Reference	127
6.36.1 Detailed Description	127
6.36.2 Member Data Documentation	127
6.36.2.1 AsBox	127
6.36.2.2 Message	128
6.37 ExpandableEnumAttribute Class Reference	128
6.37.1 Detailed Description	128
6.37.2 Property Documentation	128
6.37.2.1 AlwaysExpanded	128
6.37.2.2 ShowFlagsButtons	128
6.37.2.3 ShowInlineHelp	129
6.38 FieldEditorButtonAttribute Class Reference	129
6.38.1 Detailed Description	129
6.38.2 Constructor & Destructor Documentation	129
6.38.2.1 FieldEditorButtonAttribute()	129
6.38.3 Member Data Documentation	130
6.38.3.1 AllowMultipleTargets	130
6.38.3.2 Label	130
6.38.3.3 TargetMethod	130
6.39 FieldsMask< T > Class Template Reference	130
6.39.1 Detailed Description	131
6.39.2 Constructor & Destructor Documentation	131
6.39.2.1 FieldsMask() [1/7]	131
6.39.2.2 FieldsMask() [2/7]	131
6.39.2.3 FieldsMask() [3/7]	132
6.39.2.4 FieldsMask() [4/7]	132
6.39.2.5 FieldsMask() [5/7]	132
6.39.2.6 FieldsMask() [6/7]	132
6.39.2.7 FieldsMask() [7/7]	132
6.39.3 Member Function Documentation	132
6.39.3.1 operator Mask256()	133
6.39.4 Member Data Documentation	133
6.39.4.1 Mask	133

6.40 FixedArray< T > Class Template Reference	133
6.40.1 Detailed Description	134
6.40.2 Constructor & Destructor Documentation	134
6.40.2.1 FixedArray()	135
6.40.3 Member Function Documentation	135
6.40.3.1 Clear()	135
6.40.3.2 CopyFrom() [1/2]	135
6.40.3.3 CopyFrom() [2/2]	135
6.40.3.4 CopyTo() [1/2]	136
6.40.3.5 CopyTo() [2/2]	136
6.40.3.6 Create< T >()	137
6.40.3.7 Create< TActual, TAdapted >()	137
6.40.3.8 CreateFromFieldSequence< T >()	138
6.40.3.9 GetEnumerator()	138
6.40.3.10 IndexOf< T >()	138
6.40.3.11 ToArray()	139
6.40.3.12 ToStringString()	139
6.40.3.13 ToString()	139
6.40.4 Property Documentation	139
6.40.4.1 Length	139
6.40.4.2 this[int index]	139
6.41 FixedArray< T >.Enumerator Struct Reference	140
6.41.1 Detailed Description	140
6.41.2 Constructor & Destructor Documentation	140
6.41.2.1 Enumerator()	140
6.41.3 Member Function Documentation	141
6.41.3.1 Dispose()	141
6.41.3.2 MoveNext()	141
6.41.3.3 Reset()	141
6.41.4 Property Documentation	141
6.41.4.1 Current	141
6.42 FixedBufferPropertyAttribute Class Reference	142
6.42.1 Detailed Description	142
6.42.2 Constructor & Destructor Documentation	142
6.42.2.1 FixedBufferPropertyAttribute()	142
6.42.3 Property Documentation	143
6.42.3.1 Capacity	143
6.42.3.2 SurrogateType	143
6.42.3.3 Type	143
6.43 FixedStorage Class Reference	143
6.43.1 Detailed Description	143
6.43.2 Member Function Documentation	143

6.43.2.1 GetWordCount< T >()	143
6.44 FloatCompressed Struct Reference	144
6.44.1 Detailed Description	144
6.44.2 Member Function Documentation	145
6.44.2.1 Equals() [1/2]	145
6.44.2.2 Equals() [2/2]	145
6.44.2.3 GetHashCode()	145
6.44.2.4 operator float()	146
6.44.2.5 operator FloatCompressed()	146
6.44.2.6 operator"!="()	146
6.44.2.7 operator==()	147
6.44.3 Member Data Documentation	147
6.44.3.1 valueEncoded	147
6.45 FloatUtils Class Reference	147
6.45.1 Detailed Description	148
6.45.2 Member Function Documentation	148
6.45.2.1 Compress()	148
6.45.2.2 Decompress()	148
6.45.3 Member Data Documentation	149
6.45.3.1 DEFAULT_ACCURACY	149
6.46 FusionGlobalScriptableObject< T > Class Template Reference	149
6.46.1 Detailed Description	150
6.46.2 Member Function Documentation	150
6.46.2.1 OnDisable()	150
6.46.2.2 OnLoadedAsGlobal()	150
6.46.2.3 OnUnloadedAsGlobal()	150
6.46.2.4 TryGetGlobalInternal()	150
6.46.2.5 UnloadGlobalInternal()	151
6.46.3 Property Documentation	151
6.46.3.1 GlobalInternal	151
6.46.3.2 IsGlobal	151
6.46.3.3 IsGlobalLoadedInternal	152
6.47 FusionGlobalScriptableObjectAttribute Class Reference	152
6.47.1 Detailed Description	152
6.47.2 Constructor & Destructor Documentation	152
6.47.2.1 FusionGlobalScriptableObjectAttribute()	152
6.47.3 Property Documentation	153
6.47.3.1 DefaultContents	153
6.47.3.2 DefaultContentsGeneratorMethod	153
6.47.3.3 DefaultPath	153
6.48 FusionGlobalScriptableObjectLoadResult Struct Reference	153
6.48.1 Detailed Description	154

6.48.2 Constructor & Destructor Documentation	154
6.48.2.1 FusionGlobalScriptableObjectLoadResult()	154
6.48.3 Member Function Documentation	154
6.48.3.1 operator FusionGlobalScriptableObjectLoadResult()	154
6.48.4 Member Data Documentation	154
6.48.4.1 Object	154
6.48.4.2 Unloader	155
6.49 FusionGlobalScriptableObjectSourceAttribute Class Reference	155
6.49.1 Detailed Description	155
6.49.2 Member Function Documentation	156
6.49.2.1 Load()	156
6.49.3 Property Documentation	156
6.49.3.1 AllowEditMode	156
6.49.3.2 AllowFallback	156
6.49.3.3 ObjectType	156
6.49.3.4 Order	156
6.50 FusionMonoBehaviour Class Reference	157
6.50.1 Detailed Description	157
6.51 FusionScriptableObject Class Reference	157
6.51.1 Detailed Description	157
6.52 HeapConfiguration Class Reference	157
6.52.1 Detailed Description	158
6.52.2 Member Function Documentation	158
6.52.2.1 Init()	158
6.52.2.2 ToString()	158
6.52.3 Member Data Documentation	158
6.52.3.1 GlobalsSize	158
6.52.3.2 PageCount	158
6.52.3.3 PageShift	159
6.53 HideArrayElementLabelAttribute Class Reference	159
6.53.1 Detailed Description	159
6.53.2 Constructor & Destructor Documentation	159
6.53.2.1 HideArrayElementLabelAttribute()	159
6.54 Hitbox Class Reference	159
6.54.1 Detailed Description	160
6.54.2 Member Function Documentation	161
6.54.2.1 DrawGizmos()	161
6.54.2.2 OnDrawGizmos()	161
6.54.2.3 SetLayer()	161
6.54.3 Member Data Documentation	161
6.54.3.1 BoxExtents	161
6.54.3.2 CapsuleExtents	162

6.54.3.3 CapsuleRadius	162
6.54.3.4 GizmosColor	162
6.54.3.5 Offset	162
6.54.3.6 Root	162
6.54.3.7 SphereRadius	162
6.54.3.8 Type	163
6.54.4 Property Documentation	163
6.54.4.1 ColliderIndex	163
6.54.4.2 HitboxActive	163
6.54.4.3 HitboxIndex	163
6.54.4.4 Position	163
6.55 HitboxManager Class Reference	163
6.55.1 Detailed Description	165
6.55.2 Member Function Documentation	166
6.55.2.1 GetPlayerTickAndAlpha()	166
6.55.2.2 GetStatisticsSnapshot()	166
6.55.2.3 OverlapBox() [1/3]	166
6.55.2.4 OverlapBox() [2/3]	167
6.55.2.5 OverlapBox() [3/3]	168
6.55.2.6 OverlapSphere() [1/3]	169
6.55.2.7 OverlapSphere() [2/3]	169
6.55.2.8 OverlapSphere() [3/3]	170
6.55.2.9 PositionRotation() [1/2]	171
6.55.2.10 PositionRotation() [2/2]	171
6.55.2.11 Raycast() [1/3]	172
6.55.2.12 Raycast() [2/3]	172
6.55.2.13 Raycast() [3/3]	173
6.55.2.14 RaycastAll() [1/3]	174
6.55.2.15 RaycastAll() [2/3]	175
6.55.2.16 RaycastAll() [3/3]	176
6.55.3 Member Data Documentation	176
6.55.3.1 BVHDepth	177
6.55.3.2 BVHNodes	177
6.55.3.3 DrawInfo	177
6.55.3.4 TotalHitboxes	177
6.56 HitboxRoot Class Reference	177
6.56.1 Detailed Description	179
6.56.2 Member Enumeration Documentation	179
6.56.2.1 ConfigFlags	179
6.56.3 Member Function Documentation	179
6.56.3.1 DrawGizmos()	179
6.56.3.2 InitHitboxes()	180

6.56.3.3 IsHitboxActive()	180
6.56.3.4 OnDrawGizmos()	180
6.56.3.5 SetHitboxActive()	180
6.56.3.6 SetMinBoundingRadius()	181
6.56.4 Member Data Documentation	181
6.56.4.1 BroadRadius	181
6.56.4.2 Config	181
6.56.4.3 GizmosColor	182
6.56.4.4 Hitboxes	182
6.56.4.5 MAX_HITBOXES	182
6.56.4.6 Offset	182
6.56.5 Property Documentation	182
6.56.5.1 HitboxRootActive	182
6.56.5.2 InInterest	183
6.56.5.3 Manager	183
6.57 HostMigrationConfig Class Reference	183
6.57.1 Detailed Description	183
6.57.2 Member Data Documentation	183
6.57.2.1 EnableAutoUpdate	183
6.57.2.2 UpdateDelay	183
6.58 HostMigrationToken Class Reference	184
6.58.1 Detailed Description	184
6.58.2 Property Documentation	184
6.58.2.1 GameMode	184
6.59 IAfterAllTicks Interface Reference	184
6.59.1 Detailed Description	184
6.59.2 Member Function Documentation	184
6.59.2.1 AfterAllTicks()	184
6.60 IAfterClientPredictionReset Interface Reference	185
6.60.1 Detailed Description	185
6.60.2 Member Function Documentation	185
6.60.2.1 AfterClientPredictionReset()	185
6.61 IAfterHostMigration Interface Reference	185
6.61.1 Detailed Description	186
6.61.2 Member Function Documentation	186
6.61.2.1 AfterHostMigration()	186
6.62 IAfterRender Interface Reference	186
6.62.1 Detailed Description	186
6.62.2 Member Function Documentation	186
6.62.2.1 AfterRender()	186
6.63 IAfterSpawned Interface Reference	186
6.63.1 Detailed Description	187

6.63.2 Member Function Documentation	187
6.63.2.1 AfterSpawned()	187
6.64 IAfterTick Interface Reference	187
6.64.1 Detailed Description	187
6.64.2 Member Function Documentation	187
6.64.2.1 AfterTick()	188
6.65 IAfterUpdate Interface Reference	188
6.65.1 Detailed Description	188
6.65.2 Member Function Documentation	188
6.65.2.1 AfterUpdate()	188
6.66 IAsyncOperation Interface Reference	188
6.66.1 Detailed Description	189
6.66.2 Property Documentation	189
6.66.2.1 Error	189
6.66.2.2 IsDone	189
6.66.3 Event Documentation	189
6.66.3.1 Completed	189
6.67 IBeforeAllTicks Interface Reference	189
6.67.1 Detailed Description	190
6.67.2 Member Function Documentation	190
6.67.2.1 BeforeAllTicks()	190
6.68 IBeforeClientPredictionReset Interface Reference	190
6.68.1 Detailed Description	190
6.68.2 Member Function Documentation	191
6.68.2.1 BeforeClientPredictionReset()	191
6.69 IBeforeCopyPreviousState Interface Reference	191
6.69.1 Detailed Description	191
6.69.2 Member Function Documentation	191
6.69.2.1 BeforeCopyPreviousState()	191
6.70 IBeforeHitboxRegistration Interface Reference	191
6.70.1 Detailed Description	192
6.70.2 Member Function Documentation	192
6.70.2.1 BeforeHitboxRegistration()	192
6.71 IBeforeSimulation Interface Reference	192
6.71.1 Detailed Description	192
6.71.2 Member Function Documentation	192
6.71.2.1 BeforeSimulation()	192
6.72 IBeforeTick Interface Reference	193
6.72.1 Detailed Description	193
6.72.2 Member Function Documentation	193
6.72.2.1 BeforeTick()	193
6.73 IBeforeUpdate Interface Reference	193

6.73.1 Detailed Description	194
6.73.2 Member Function Documentation	194
6.73.2.1 BeforeUpdate()	194
6.74 ICoroutine Interface Reference	194
6.74.1 Detailed Description	194
6.75 IDespawned Interface Reference	194
6.75.1 Detailed Description	194
6.75.2 Member Function Documentation	194
6.75.2.1 Despawned()	194
6.76 IElementReaderWriter< T > Interface Template Reference	195
6.76.1 Detailed Description	195
6.76.2 Member Function Documentation	195
6.76.2.1 GetElementHashCode()	195
6.76.2.2 GetElementWordCount()	196
6.76.2.3 Read()	196
6.76.2.4 ReadRef()	196
6.76.2.5 Write()	197
6.77 IFixedStorage Interface Reference	197
6.77.1 Detailed Description	197
6.78 IInputAuthorityGained Interface Reference	197
6.78.1 Detailed Description	198
6.78.2 Member Function Documentation	198
6.78.2.1 InputAuthorityGained()	198
6.79 IInputAuthorityLost Interface Reference	198
6.79.1 Detailed Description	198
6.79.2 Member Function Documentation	198
6.79.2.1 InputAuthorityLost()	198
6.80 IInterestEnter Interface Reference	198
6.80.1 Detailed Description	199
6.80.2 Member Function Documentation	199
6.80.2.1 InterestEnter()	199
6.81 IInterestExit Interface Reference	199
6.81.1 Detailed Description	199
6.81.2 Member Function Documentation	199
6.81.2.1 InterestExit()	199
6.82 ILocalPrefabCreated Interface Reference	200
6.82.1 Detailed Description	200
6.82.2 Member Function Documentation	200
6.82.2.1 LocalPrefabCreated()	200
6.83 INetworkArray Interface Reference	200
6.83.1 Detailed Description	201
6.83.2 Property Documentation	201

6.83.2.1 this[int index]	201
6.84 INetworkAssetSource< T > Interface Template Reference	202
6.84.1 Detailed Description	202
6.84.2 Member Function Documentation	202
6.84.2.1 Acquire()	202
6.84.2.2 Release()	203
6.84.2.3 WaitForResult()	203
6.84.3 Property Documentation	203
6.84.3.1 Description	203
6.84.3.2 IsCompleted	203
6.85 INetworkDictionary Interface Reference	203
6.85.1 Detailed Description	204
6.85.2 Member Function Documentation	204
6.85.2.1 Add()	204
6.86 INetworkInput Interface Reference	204
6.86.1 Detailed Description	204
6.87 INetworkLinkedList Interface Reference	204
6.87.1 Detailed Description	205
6.87.2 Member Function Documentation	205
6.87.2.1 Add()	205
6.88 INetworkObjectInitializer Interface Reference	205
6.88.1 Detailed Description	205
6.88.2 Member Function Documentation	205
6.88.2.1 InitializeNetworkState()	206
6.89 INetworkObjectProvider Interface Reference	206
6.89.1 Detailed Description	206
6.89.2 Member Function Documentation	206
6.89.2.1 AcquirePrefabInstance()	206
6.89.2.2 ReleaseInstance()	207
6.90 INetworkPrefabSource Interface Reference	207
6.90.1 Detailed Description	207
6.90.2 Property Documentation	208
6.90.2.1 AssetGuid	208
6.91 INetworkRunnerCallbacks Interface Reference	208
6.91.1 Detailed Description	209
6.91.2 Member Function Documentation	209
6.91.2.1 OnConnectedToServer()	209
6.91.2.2 OnConnectFailed()	209
6.91.2.3 OnConnectRequest()	209
6.91.2.4 OnCustomAuthenticationResponse()	210
6.91.2.5 OnDisconnectedFromServer()	210
6.91.2.6 OnHostMigration()	210

6.91.2.7 OnInput()	211
6.91.2.8 OnInputMissing()	211
6.91.2.9 OnObjectEnterAOI()	211
6.91.2.10 OnObjectExitAOI()	211
6.91.2.11 OnPlayerJoined()	212
6.91.2.12 OnPlayerLeft()	212
6.91.2.13 OnReliableDataProgress()	212
6.91.2.14 OnReliableDataReceived()	213
6.91.2.15 OnSceneLoadDone()	213
6.91.2.16 OnSceneLoadStart()	213
6.91.2.17 OnSessionListUpdated()	214
6.91.2.18 OnShutdown()	214
6.91.2.19 OnUserSimulationMessage()	214
6.92 INetworkRunnerUpdater Interface Reference	214
6.92.1 Detailed Description	215
6.92.2 Member Function Documentation	215
6.92.2.1 Initialize()	215
6.92.2.2 Shutdown()	215
6.93 INetworkSceneManager Interface Reference	216
6.93.1 Detailed Description	216
6.93.2 Member Function Documentation	216
6.93.2.1 GetSceneRef() [1/2]	217
6.93.2.2 GetSceneRef() [2/2]	217
6.93.2.3 Initialize()	217
6.93.2.4 IsRunnerScene()	217
6.93.2.5 LoadScene()	217
6.93.2.6 MakeDontDestroyOnLoad()	218
6.93.2.7 MoveGameObjectToScene()	218
6.93.2.8 OnSceneInfoChanged()	218
6.93.2.9 Shutdown()	218
6.93.2.10 TryGetPhysicsScene2D()	219
6.93.2.11 TryGetPhysicsScene3D()	219
6.93.2.12 UnloadScene()	219
6.93.3 Property Documentation	219
6.93.3.1 IsBusy	219
6.93.3.2 MainRunnerScene	220
6.94 INetworkStruct Interface Reference	220
6.94.1 Detailed Description	220
6.95 INetworkTRSPTeleport Interface Reference	220
6.95.1 Detailed Description	220
6.95.2 Member Function Documentation	220
6.95.2.1 Teleport()	221

6.96 InlineHelpAttribute Class Reference	221
6.96.1 Detailed Description	221
6.96.2 Constructor & Destructor Documentation	221
6.96.2.1 InlineHelpAttribute()	221
6.96.3 Property Documentation	221
6.96.3.1 ShowTypeHelp	222
6.97 IUnitySurrogate Interface Reference	222
6.97.1 Detailed Description	222
6.97.2 Member Function Documentation	222
6.97.2.1 Read()	222
6.97.2.2 Write()	223
6.98 IUnityValueSurrogate< T > Interface Template Reference	223
6.98.1 Detailed Description	223
6.98.2 Property Documentation	223
6.98.2.1 DataProperty	224
6.99 UnityArraySurrogate< T, ReaderWriter > Class Template Reference	224
6.99.1 Detailed Description	224
6.99.2 Member Function Documentation	224
6.99.2.1 Init()	225
6.99.2.2 Read()	225
6.99.2.3 Write()	225
6.99.3 Property Documentation	226
6.99.3.1 DataProperty	226
6.100 UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter > Class Template Reference	226
6.100.1 Detailed Description	226
6.100.2 Member Function Documentation	227
6.100.2.1 Init()	227
6.100.2.2 Read()	227
6.100.2.3 Write()	228
6.100.3 Property Documentation	228
6.100.3.1 DataProperty	228
6.101 UnityLinkedListSurrogate< T, ReaderWriter > Class Template Reference	228
6.101.1 Detailed Description	228
6.101.2 Member Function Documentation	229
6.101.2.1 Init()	229
6.101.2.2 Read()	229
6.101.2.3 Write()	230
6.101.3 Property Documentation	230
6.101.3.1 DataProperty	230
6.102 UnitySurrogateBase Class Reference	230
6.102.1 Detailed Description	231

6.102.2 Member Function Documentation	231
6.102.2.1 Init()	231
6.102.2.2 Read()	231
6.102.2.3 Write()	231
6.103 UnityValueSurrogate< T, TReaderWriter > Class Template Reference	232
6.103.1 Detailed Description	232
6.103.2 Member Function Documentation	233
6.103.2.1 Init()	233
6.103.2.2 Read()	233
6.103.2.3 Write()	233
6.103.3 Property Documentation	234
6.103.3.1 DataProperty	234
6.104 InterpolatedErrorCorrectionSettings Class Reference	234
6.104.1 Detailed Description	234
6.104.2 Member Data Documentation	234
6.104.2.1 MaxRate	235
6.104.2.2 MinRate	235
6.104.2.3 PosBlendEnd	235
6.104.2.4 PosBlendStart	235
6.104.2.5 PosMinCorrection	236
6.104.2.6 PosTeleportDistance	236
6.104.2.7 RotBlendEnd	236
6.104.2.8 RotBlendStart	236
6.104.2.9 RotTeleportRadians	237
6.105 IPlayerJoined Interface Reference	237
6.105.1 Detailed Description	237
6.105.2 Member Function Documentation	237
6.105.2.1 PlayerJoined()	237
6.106 IPlayerLeft Interface Reference	237
6.106.1 Detailed Description	238
6.106.2 Member Function Documentation	238
6.106.2.1 PlayerLeft()	238
6.107 IRemotePrefabCreated Interface Reference	238
6.107.1 Detailed Description	238
6.107.2 Member Function Documentation	238
6.107.2.1 RemotePrefabCreated()	239
6.108 ISceneLoadDone Interface Reference	239
6.108.1 Detailed Description	239
6.108.2 Member Function Documentation	239
6.108.2.1 SceneLoadDone()	239
6.109 ISceneLoadStart Interface Reference	239
6.109.1 Detailed Description	240

6.109.2 Member Function Documentation	240
6.109.2.1 SceneLoadStart()	240
6.110 ISimulationEnter Interface Reference	240
6.110.1 Detailed Description	240
6.110.2 Member Function Documentation	240
6.110.2.1 SimulationEnter()	240
6.111 ISimulationExit Interface Reference	241
6.111.1 Detailed Description	241
6.111.2 Member Function Documentation	241
6.111.2.1 SimulationExit()	241
6.112 ISpawned Interface Reference	241
6.112.1 Detailed Description	241
6.112.2 Member Function Documentation	242
6.112.2.1 Spawning()	242
6.113 IStateAuthorityChanged Interface Reference	242
6.113.1 Detailed Description	242
6.113.2 Member Function Documentation	242
6.113.2.1 StateAuthorityChanged()	242
6.114 LagCompensatedHit Struct Reference	242
6.114.1 Detailed Description	243
6.114.2 Member Function Documentation	243
6.114.2.1 operator LagCompensatedHit() [1/2]	243
6.114.2.2 operator LagCompensatedHit() [2/2]	244
6.114.3 Member Data Documentation	244
6.114.3.1 Collider	244
6.114.3.2 Collider2D	244
6.114.3.3 Distance	245
6.114.3.4 GameObject	245
6.114.3.5 Hitbox	245
6.114.3.6 HitboxColliderPosition	245
6.114.3.7 HitboxColliderRotation	245
6.114.3.8 Normal	245
6.114.3.9 Point	246
6.114.3.10 Type	246
6.115 AABB Struct Reference	246
6.115.1 Detailed Description	246
6.115.2 Constructor & Destructor Documentation	246
6.115.2.1 AABB() [1/2]	246
6.115.2.2 AABB() [2/2]	247
6.115.3 Member Data Documentation	247
6.115.3.1 Center	247
6.115.3.2 Extents	247

6.115.3.3 Max	247
6.115.3.4 Min	248
6.116 BoxOverlapQuery Class Reference	248
6.116.1 Detailed Description	248
6.116.2 Constructor & Destructor Documentation	248
6.116.2.1 BoxOverlapQuery() [1/2]	248
6.116.2.2 BoxOverlapQuery() [2/2]	249
6.116.3 Member Function Documentation	249
6.116.3.1 Check()	249
6.116.4 Member Data Documentation	249
6.116.4.1 Center	250
6.116.4.2 Extents	250
6.116.4.3 Rotation	250
6.117 BoxOverlapQueryParams Struct Reference	250
6.117.1 Detailed Description	251
6.117.2 Constructor & Destructor Documentation	251
6.117.2.1 BoxOverlapQueryParams()	251
6.117.3 Member Data Documentation	251
6.117.3.1 Center	251
6.117.3.2 Extents	251
6.117.3.3 QueryParams	252
6.117.3.4 Rotation	252
6.117.3.5 StaticHitsCapacity	252
6.118 BVHDraw Class Reference	252
6.118.1 Detailed Description	252
6.118.2 Member Function Documentation	252
6.118.2.1 GetEnumerator()	252
6.119 BVHNodeDrawInfo Class Reference	253
6.119.1 Detailed Description	253
6.119.2 Property Documentation	253
6.119.2.1 Bounds	253
6.119.2.2 Depth	253
6.119.2.3 MaxDepth	253
6.120 ColliderDrawInfo Class Reference	254
6.120.1 Detailed Description	254
6.120.2 Property Documentation	254
6.120.2.1 BoxExtents	254
6.120.2.2 CapsuleBottomCenter	254
6.120.2.3 CapsuleExtents	255
6.120.2.4 CapsuleTopCenter	255
6.120.2.5 LocalToWorldMatrix	255
6.120.2.6 Offset	255

6.120.2.7 Radius	255
6.120.2.8 Type	255
6.121 HitboxColliderContainerDraw Class Reference	256
6.121.1 Detailed Description	256
6.121.2 Member Function Documentation	256
6.121.2.1 GetEnumerator()	256
6.122 LagCompensatedExt Class Reference	256
6.122.1 Detailed Description	256
6.122.2 Member Function Documentation	256
6.122.2.1 SortDistance()	256
6.122.2.2 SortReference()	257
6.123 LagCompensationDraw Class Reference	257
6.123.1 Detailed Description	258
6.123.2 Member Function Documentation	258
6.123.2.1 GizmosDrawWireCapsule()	258
6.123.3 Member Data Documentation	258
6.123.3.1 BVHDraw	258
6.123.3.2 SnapshotHistoryDraw	258
6.124 LagCompensationUtils.ContactData Struct Reference	258
6.124.1 Detailed Description	259
6.124.2 Member Data Documentation	259
6.124.2.1 Normal	259
6.124.2.2 Penetration	259
6.124.2.3 Point	259
6.125 PositionRotationQueryParams Struct Reference	259
6.125.1 Detailed Description	260
6.125.2 Constructor & Destructor Documentation	260
6.125.2.1 PositionRotationQueryParams()	260
6.125.3 Member Data Documentation	260
6.125.3.1 Hitbox	260
6.125.3.2 QueryParams	261
6.126 Query Class Reference	261
6.126.1 Detailed Description	261
6.126.2 Constructor & Destructor Documentation	262
6.126.2.1 Query()	262
6.126.3 Member Function Documentation	263
6.126.3.1 Check()	263
6.126.4 Member Data Documentation	263
6.126.4.1 Alpha	263
6.126.4.2 LayerMask	263
6.126.4.3 Options	264
6.126.4.4 Player	264

6.126.4.5 PreProcessingDelegate	264
6.126.4.6 Tick	264
6.126.4.7 TickTo	264
6.126.4.8 TriggerInteraction	264
6.126.4.9 UserArgs	265
6.127 QueryParams Struct Reference	265
6.127.1 Detailed Description	265
6.127.2 Member Data Documentation	265
6.127.2.1 Alpha	265
6.127.2.2 LayerMask	266
6.127.2.3 Options	266
6.127.2.4 Player	266
6.127.2.5 PreProcessingDelegate	266
6.127.2.6 Tick	266
6.127.2.7 TickTo	266
6.127.2.8 TriggerInteraction	267
6.127.2.9 UserArgs	267
6.128 RaycastAllQuery Class Reference	267
6.128.1 Detailed Description	267
6.128.2 Constructor & Destructor Documentation	267
6.128.2.1 RaycastAllQuery() [1/2]	267
6.128.2.2 RaycastAllQuery() [2/2]	268
6.129 RaycastQuery Class Reference	268
6.129.1 Detailed Description	269
6.129.2 Constructor & Destructor Documentation	269
6.129.2.1 RaycastQuery()	269
6.129.3 Member Function Documentation	269
6.129.3.1 Check()	269
6.129.4 Member Data Documentation	270
6.129.4.1 Direction	270
6.129.4.2 Length	270
6.129.4.3 Origin	270
6.130 RaycastQueryParams Struct Reference	270
6.130.1 Detailed Description	271
6.130.2 Constructor & Destructor Documentation	271
6.130.2.1 RaycastQueryParams()	271
6.130.3 Member Data Documentation	271
6.130.3.1 Direction	271
6.130.3.2 Length	271
6.130.3.3 Origin	272
6.130.3.4 QueryParams	272
6.130.3.5 StaticHitsCapacity	272

6.131 SnapshotHistoryDraw Class Reference	272
6.131.1 Detailed Description	272
6.131.2 Member Function Documentation	272
6.131.2.1 GetEnumerator()	273
6.132 SphereOverlapQuery Class Reference	273
6.132.1 Detailed Description	273
6.132.2 Constructor & Destructor Documentation	273
6.132.2.1 SphereOverlapQuery() [1/2]	273
6.132.2.2 SphereOverlapQuery() [2/2]	274
6.132.3 Member Function Documentation	274
6.132.3.1 Check()	274
6.132.4 Member Data Documentation	274
6.132.4.1 Center	275
6.132.4.2 Radius	275
6.133 SphereOverlapQueryParams Struct Reference	275
6.133.1 Detailed Description	275
6.133.2 Constructor & Destructor Documentation	275
6.133.2.1 SphereOverlapQueryParams()	275
6.133.3 Member Data Documentation	276
6.133.3.1 Center	276
6.133.3.2 QueryParams	276
6.133.3.3 Radius	276
6.133.3.4 StaticHitsCapacity	276
6.134 LagCompensationSettings Class Reference	276
6.134.1 Detailed Description	277
6.134.2 Member Data Documentation	277
6.134.2.1 CachedStaticCollidersSize	277
6.134.2.2 Enabled	277
6.134.2.3 HitboxBufferLengthInMs	277
6.134.2.4 HitboxDefaultCapacity	278
6.134.3 Property Documentation	278
6.134.3.1 ExpansionFactor	278
6.134.3.2 Optimize	278
6.135 LayerAttribute Class Reference	278
6.135.1 Detailed Description	278
6.136 LayerMatrixAttribute Class Reference	278
6.136.1 Detailed Description	278
6.137 LobbyInfo Class Reference	279
6.137.1 Detailed Description	279
6.137.2 Property Documentation	279
6.137.2.1 IsValid	279
6.137.2.2 Name	279

6.137.2.3 Region	279
6.138 LogSimpleUnity Class Reference	280
6.138.1 Detailed Description	280
6.139 Mask256 Struct Reference	280
6.139.1 Detailed Description	281
6.139.2 Constructor & Destructor Documentation	281
6.139.2.1 Mask256()	281
6.139.3 Member Function Documentation	281
6.139.3.1 Clear()	281
6.139.3.2 Equals() [1/2]	282
6.139.3.3 Equals() [2/2]	282
6.139.3.4 GetBit()	282
6.139.3.5 GetHashCode()	282
6.139.3.6 IsNothing()	282
6.139.3.7 operator long()	283
6.139.3.8 operator Mask256()	283
6.139.3.9 operator&()	283
6.139.3.10 operator" "()	283
6.139.3.11 operator~()	283
6.139.3.12 SetBit()	284
6.139.3.13 ToString()	284
6.139.4 Property Documentation	284
6.139.4.1 this[int i]	284
6.140 MaxStringByteCountAttribute Class Reference	284
6.140.1 Detailed Description	285
6.140.2 Constructor & Destructor Documentation	285
6.140.2.1 MaxStringByteCountAttribute()	285
6.140.3 Property Documentation	285
6.140.3.1 ByteCount	285
6.140.3.2 Encoding	285
6.141 NestedComponentUtilities Class Reference	285
6.141.1 Detailed Description	286
6.141.2 Member Function Documentation	287
6.141.2.1 EnsureRootComponentExists< T, TStopOn >()	287
6.141.2.2 FindObjectsOfTypeInOrder< T >() [1/2]	287
6.141.2.3 FindObjectsOfTypeInOrder< T >() [2/2]	288
6.141.2.4 FindObjectsOfTypeInOrder< T, TCast >() [1/2]	289
6.141.2.5 FindObjectsOfTypeInOrder< T, TCast >() [2/2]	289
6.141.2.6 GetNestedComponentInChildren< T, TStopOn >()	290
6.141.2.7 GetNestedComponentInParent< T, TStopOn >()	291
6.141.2.8 GetNestedComponentInParents< T, TStopOn >()	291
6.141.2.9 GetNestedComponentsInChildren< T >()	292

6.141.2.10 GetNestedComponentsInChildren< T, TSearch, TStop >()	292
6.141.2.11 GetNestedComponentsInChildren< T, TStopOn >()	292
6.141.2.12 GetNestedComponentsInParents< T >()	293
6.141.2.13 GetNestedComponentsInParents< T, TStop >()	293
6.141.2.14 GetParentComponent< T >()	293
6.142 NetworkArray< T > Struct Template Reference	293
6.142.1 Detailed Description	295
6.142.2 Constructor & Destructor Documentation	295
6.142.2.1 NetworkArray()	295
6.142.3 Member Function Documentation	295
6.142.3.1 Clear()	296
6.142.3.2 CopyFrom() [1/2]	296
6.142.3.3 CopyFrom() [2/2]	296
6.142.3.4 CopyTo() [1/3]	297
6.142.3.5 CopyTo() [2/3]	297
6.142.3.6 CopyTo() [3/3]	297
6.142.3.7 Get()	298
6.142.3.8 GetEnumerator()	298
6.142.3.9 operator NetworkArrayReadOnly< T >()	298
6.142.3.10 Set()	298
6.142.3.11 ToArray()	299
6.142.3.12 ToString()	299
6.142.3.13 ToReadOnly()	299
6.142.3.14 ToString()	299
6.142.4 Property Documentation	299
6.142.4.1 Length	299
6.142.4.2 this[int index]	299
6.143 NetworkArray< T >.Enumerator Struct Reference	300
6.143.1 Detailed Description	300
6.143.2 Constructor & Destructor Documentation	300
6.143.2.1 Enumerator()	300
6.143.3 Member Function Documentation	301
6.143.3.1 Dispose()	301
6.143.3.2 MoveNext()	301
6.143.3.3 Reset()	301
6.143.4 Property Documentation	301
6.143.4.1 Current [1/2]	301
6.143.4.2 Current [2/2]	302
6.144 NetworkArrayExtensions Class Reference	302
6.144.1 Detailed Description	302
6.144.2 Member Function Documentation	302
6.144.2.1 GetRef< T >()	302

6.144.2.2 IndexOf< T >()	303
6.145 NetworkArrayReadOnly< T > Struct Template Reference	303
6.145.1 Detailed Description	303
6.145.2 Property Documentation	304
6.145.2.1 Length	304
6.145.2.2 this[int index]	304
6.146 NetworkAssemblyIgnoreAttribute Class Reference	304
6.146.1 Detailed Description	304
6.147 NetworkAssemblyWeavedAttribute Class Reference	304
6.147.1 Detailed Description	304
6.148 NetworkBehaviour Class Reference	304
6.148.1 Detailed Description	308
6.148.2 Member Function Documentation	308
6.148.2.1 CopyBackingFieldsToState()	308
6.148.2.2 CopyStateFrom()	308
6.148.2.3 CopyStateToBackingFields()	308
6.148.2.4 Despawned()	309
6.148.2.5 FixedUpdateNetwork()	309
6.148.2.6 GetArrayReader< T >() [1/2]	309
6.148.2.7 GetArrayReader< T >() [2/2]	310
6.148.2.8 GetBehaviourReader< T >() [1/2]	310
6.148.2.9 GetBehaviourReader< T >() [2/2]	311
6.148.2.10 GetBehaviourReader< TBehaviour, TProperty >()	311
6.148.2.11 GetChangeDetector()	312
6.148.2.12 GetDictionaryReader< K, V >() [1/2]	312
6.148.2.13 GetDictionaryReader< K, V >() [2/2]	313
6.148.2.14 GetInput< T >() [1/2]	313
6.148.2.15 GetInput< T >() [2/2]	314
6.148.2.16 GetLinkListReader< T >() [1/2]	314
6.148.2.17 GetLinkListReader< T >() [2/2]	314
6.148.2.18 GetLocalAuthorityMask()	315
6.148.2.19 GetPropertyReader< T >() [1/2]	315
6.148.2.20 GetPropertyReader< T >() [2/2]	316
6.148.2.21 GetPropertyReader< TBehaviour, TProperty >()	316
6.148.2.22 MakeInitializer< K, V >()	317
6.148.2.23 MakeInitializer< T >()	317
6.148.2.24 MakePtr< T >() [1/2]	317
6.148.2.25 MakePtr< T >() [2/2]	318
6.148.2.26 MakeRef< T >() [1/2]	318
6.148.2.27 MakeRef< T >() [2/2]	319
6.148.2.28 NetworkDeserialize()	319
6.148.2.29 NetworkSerialize() [1/2]	320

6.148.2.30 NetworkSerialize() [2/2]	320
6.148.2.31 NetworkUnwrap()	321
6.148.2.32 NetworkWrap()	321
6.148.2.33 operator NetworkBehaviourId()	321
6.148.2.34 ReinterpretState< T >()	323
6.148.2.35 ReplicateTo() [1/2]	323
6.148.2.36 ReplicateTo() [2/2]	324
6.148.2.37 ReplicateToAll()	324
6.148.2.38 ResetState()	324
6.148.2.39 Spawning()	325
6.148.2.40 TryGetSnapshotsBuffers()	325
6.148.3 Member Data Documentation	325
6.148.3.1 offset	325
6.148.4 Property Documentation	325
6.148.4.1 ChangedTick	326
6.148.4.2 DynamicWordCount	326
6.148.4.3 HasInputAuthority	326
6.148.4.4 HasStateAuthority	326
6.148.4.5 Id	326
6.148.4.6 IsProxy	326
6.148.4.7 StateBuffer	327
6.148.4.8 StateBufferIsValid	327
6.149 NetworkBehaviour.ArrayReader< T > Struct Template Reference	327
6.149.1 Detailed Description	327
6.149.2 Member Function Documentation	327
6.149.2.1 Read()	327
6.150 NetworkBehaviour.BehaviourReader< T > Struct Template Reference	328
6.150.1 Detailed Description	328
6.150.2 Member Function Documentation	328
6.150.2.1 Read()	328
6.150.3 Member Data Documentation	329
6.150.3.1 T	329
6.151 NetworkBehaviour.ChangeDetector Class Reference	329
6.151.1 Detailed Description	330
6.151.2 Member Enumeration Documentation	330
6.151.2.1 Source	330
6.151.3 Member Function Documentation	330
6.151.3.1 DetectChanges() [1/2]	330
6.151.3.2 DetectChanges() [2/2]	331
6.151.3.3 Init()	331
6.152 NetworkBehaviour.ChangeDetector.Enumerable Struct Reference	332
6.152.1 Detailed Description	332

6.152.2 Member Function Documentation	332
6.152.2.1 Changed()	332
6.152.2.2 GetEnumerator()	332
6.153 NetworkBehaviour.ChangeDetector.Enumerator Struct Reference	333
6.153.1 Detailed Description	333
6.153.2 Member Function Documentation	333
6.153.2.1 MoveNext()	333
6.153.2.2 Reset()	333
6.153.3 Property Documentation	334
6.153.3.1 Current	334
6.154 NetworkBehaviour.DictionaryReader< K, V > Struct Template Reference	334
6.154.1 Detailed Description	334
6.154.2 Member Function Documentation	334
6.154.2.1 Read()	334
6.155 NetworkBehaviour.LinkedListReader< T > Struct Template Reference	335
6.155.1 Detailed Description	335
6.155.2 Member Function Documentation	335
6.155.2.1 Read()	335
6.156 NetworkBehaviour.PropertyReader< T > Struct Template Reference	336
6.156.1 Detailed Description	336
6.156.2 Constructor & Destructor Documentation	336
6.156.2.1 PropertyReader()	336
6.156.3 Member Function Documentation	337
6.156.3.1 Read()	337
6.156.4 Member Data Documentation	337
6.156.4.1 T	337
6.157 NetworkBehaviourBuffer Struct Reference	337
6.157.1 Detailed Description	338
6.157.2 Member Function Documentation	338
6.157.2.1 operator bool()	338
6.157.2.2 Read() [1/5]	339
6.157.2.3 Read() [2/5]	339
6.157.2.4 Read() [3/5]	339
6.157.2.5 Read() [4/5]	341
6.157.2.6 Read() [5/5]	341
6.157.2.7 Read< T >() [1/2]	341
6.157.2.8 Read< T >() [2/2]	343
6.157.2.9 ReinterpretState< T >()	343
6.157.3 Property Documentation	344
6.157.3.1 Length	344
6.157.3.2 this[int index]	344
6.157.3.3 Tick	344

6.157.3.4 Valid	345
6.158 NetworkBehaviourBufferInterpolator Struct Reference	345
6.158.1 Detailed Description	346
6.158.2 Constructor & Destructor Documentation	346
6.158.2.1 NetworkBehaviourBufferInterpolator()	346
6.158.3 Member Function Documentation	347
6.158.3.1 Angle() [1/2]	347
6.158.3.2 Angle() [2/2]	347
6.158.3.3 Bool() [1/2]	347
6.158.3.4 Bool() [2/2]	348
6.158.3.5 Float() [1/2]	348
6.158.3.6 Float() [2/2]	348
6.158.3.7 Int() [1/2]	350
6.158.3.8 Int() [2/2]	350
6.158.3.9 operator bool()	350
6.158.3.10 Quaternion() [1/2]	351
6.158.3.11 Quaternion() [2/2]	351
6.158.3.12 Select< T >() [1/2]	352
6.158.3.13 Select< T >() [2/2]	352
6.158.3.14 Vector2() [1/2]	353
6.158.3.15 Vector2() [2/2]	353
6.158.3.16 Vector3() [1/2]	353
6.158.3.17 Vector3() [2/2]	354
6.158.3.18 Vector4() [1/2]	354
6.158.3.19 Vector4() [2/2]	354
6.158.4 Member Data Documentation	355
6.158.4.1 Alpha	355
6.158.4.2 Behaviour	355
6.158.4.3 From	355
6.158.4.4 To	355
6.158.4.5 Valid	355
6.159 NetworkBehaviourId Struct Reference	356
6.159.1 Detailed Description	357
6.159.2 Member Function Documentation	357
6.159.2.1 Equals() [1/2]	357
6.159.2.2 Equals() [2/2]	357
6.159.2.3 GetHashCode()	357
6.159.2.4 operator"!=="()	358
6.159.2.5 operator==()	358
6.159.2.6 ToString()	359
6.159.3 Member Data Documentation	359
6.159.3.1 Behaviour	359

6.159.3.2 Object	359
6.159.3.3 SIZE	359
6.159.4 Property Documentation	359
6.159.4.1 IsValid	359
6.159.4.2 None	360
6.160 NetworkBehaviourUtils Class Reference	360
6.160.1 Detailed Description	361
6.160.2 Member Function Documentation	362
6.160.2.1 CloneArray< T >()	362
6.160.2.2 CopyFromNetworkArray< T >()	362
6.160.2.3 CopyFromNetworkDictionary< D, K, V >()	363
6.160.2.4 CopyFromNetworkList< T >()	363
6.160.2.5 GetMetaData()	364
6.160.2.6 GetRpcStaticIndexOrThrow()	364
6.160.2.7 GetStaticWordCount()	365
6.160.2.8 GetWordCount()	365
6.160.2.9 HasStaticWordCount()	366
6.160.2.10 InitializeNetworkArray< T >()	366
6.160.2.11 InitializeNetworkDictionary< D, K, V >()	367
6.160.2.12 InitializeNetworkList< T >()	367
6.160.2.13 InternalOnDestroy()	368
6.160.2.14 InternalOnDisable()	368
6.160.2.15 InternalOnEnable()	368
6.160.2.16 MakeSerializableDictionary< K, V >()	369
6.160.2.17 NotifyLocalSimulationNotAllowedToSendRpc()	369
6.160.2.18 NotifyLocalTargetedRpcCulled()	370
6.160.2.19 NotifyNetworkUnwrapFailed< T >()	370
6.160.2.20 NotifyNetworkWrapFailed< T >() [1/2]	370
6.160.2.21 NotifyNetworkWrapFailed< T >() [2/2]	371
6.160.2.22 NotifyRpcPayloadSizeExceeded()	371
6.160.2.23 NotifyRpcTargetUnreachable()	371
6.160.2.24 RegisterMetaData()	372
6.160.2.25 RegisterRpcInvokeDelegates()	372
6.160.2.26 ShouldRegisterRpcInvokeDelegates()	372
6.160.2.27 ThrowIfBehaviourNotInitialized()	373
6.160.2.28 TryGetRpcInvokeDelegateArray()	373
6.160.2.29 TryGetRpcStaticInvokeDelegate()	374
6.160.3 Member Data Documentation	374
6.160.3.1 InvokeRpc	374
6.161 NetworkBehaviourUtils.ArrayInitializer< T > Struct Template Reference	374
6.161.1 Detailed Description	374
6.161.2 Member Function Documentation	375

6.161.2.1 operator NetworkArray< T >()	375
6.161.2.2 operator NetworkLinkedList< T >()	375
6.162 NetworkBehaviourUtils.DictionaryInitializer< K, V > Struct Template Reference	376
6.162.1 Detailed Description	376
6.162.2 Member Function Documentation	376
6.162.2.1 operator NetworkDictionary< K, V >()	376
6.163 NetworkBehaviourUtils.MetaData Struct Reference	377
6.163.1 Detailed Description	377
6.164 NetworkBehaviourWeavedAttribute Class Reference	377
6.164.1 Detailed Description	377
6.164.2 Constructor & Destructor Documentation	377
6.164.2.1 NetworkBehaviourWeavedAttribute()	377
6.164.3 Property Documentation	378
6.164.3.1 WordCount	378
6.165 NetworkBool Struct Reference	378
6.165.1 Detailed Description	379
6.165.2 Constructor & Destructor Documentation	379
6.165.2.1 NetworkBool()	379
6.165.3 Member Function Documentation	379
6.165.3.1 Equals() [1/2]	379
6.165.3.2 Equals() [2/2]	379
6.165.3.3 GetHashCode()	380
6.165.3.4 operator bool()	380
6.165.3.5 operator NetworkBool()	380
6.165.3.6 ToString()	381
6.166 NetworkButtons Struct Reference	381
6.166.1 Detailed Description	382
6.166.2 Constructor & Destructor Documentation	382
6.166.2.1 NetworkButtons()	382
6.166.3 Member Function Documentation	382
6.166.3.1 Equals() [1/2]	383
6.166.3.2 Equals() [2/2]	383
6.166.3.3 GetHashCode()	383
6.166.3.4 GetPressed()	383
6.166.3.5 GetReleased()	384
6.166.3.6 IsSet()	384
6.166.3.7 IsSet< T >()	384
6.166.3.8 Set()	386
6.166.3.9 Set< T >()	386
6.166.3.10 SetAllDown()	387
6.166.3.11 SetAllUp()	387
6.166.3.12 SetDown()	387

6.166.3.13 SetDown< T >()	387
6.166.3.14 SetUp()	388
6.166.3.15 SetUp< T >()	388
6.166.3.16 WasPressed()	389
6.166.3.17 WasPressed< T >()	389
6.166.3.18 WasReleased()	390
6.166.3.19 WasReleased< T >()	390
6.166.4 Member Data Documentation	390
6.166.4.1 NetworkButtons	391
6.166.5 Property Documentation	391
6.166.5.1 Bits	391
6.167 NetworkConfiguration Class Reference	391
6.167.1 Detailed Description	392
6.167.2 Member Enumeration Documentation	392
6.167.2.1 ReliableDataTransfers	392
6.167.3 Member Function Documentation	392
6.167.3.1 Init()	392
6.167.4 Member Data Documentation	393
6.167.4.1 ConnectionShutdownTime	393
6.167.4.2 ConnectionTimeout	393
6.167.4.3 ReliableDataTransferModes	393
6.167.5 Property Documentation	393
6.167.5.1 ConnectAttempts	393
6.167.5.2 ConnectInterval	394
6.167.5.3 ConnectionDefaultRtt	394
6.167.5.4 ConnectionPingInterval	394
6.167.5.5 SocketRecvBufferSize	394
6.167.5.6 SocketSendBufferSize	394
6.168 NetworkDelegates Class Reference	394
6.168.1 Detailed Description	395
6.169 NetworkDeserializeMethodAttribute Class Reference	395
6.169.1 Detailed Description	395
6.170 NetworkDictionary< K, V > Struct Template Reference	395
6.170.1 Detailed Description	397
6.170.2 Constructor & Destructor Documentation	398
6.170.2.1 NetworkDictionary()	398
6.170.3 Member Function Documentation	398
6.170.3.1 Add() [1/2]	398
6.170.3.2 Add() [2/2]	398
6.170.3.3 Clear()	399
6.170.3.4 ContainsKey()	399
6.170.3.5 ContainsValue()	399

6.170.3.6 Get()	399
6.170.3.7 GetEnumerator()	399
6.170.3.8 operator NetworkDictionaryReadOnly< K, V >()	400
6.170.3.9 Remove() [1/2]	400
6.170.3.10 Remove() [2/2]	400
6.170.3.11 Set()	401
6.170.3.12 ToReadOnly()	401
6.170.3.13 TryGet()	401
6.170.4 Member Data Documentation	401
6.170.4.1 META_WORD_COUNT	402
6.170.5 Property Documentation	402
6.170.5.1 Capacity	402
6.170.5.2 Count	402
6.170.5.3 this[K key]	402
6.171 NetworkDictionary< K, V >.Enumerator Struct Reference	402
6.171.1 Detailed Description	403
6.171.2 Member Function Documentation	403
6.171.2.1 Dispose()	403
6.171.2.2 MoveNext()	403
6.171.2.3 Reset()	403
6.171.3 Property Documentation	403
6.171.3.1 Current	403
6.172 NetworkDictionaryReadOnly< K, V > Struct Template Reference	404
6.172.1 Detailed Description	405
6.172.2 Member Function Documentation	406
6.172.2.1 Get()	406
6.172.2.2 TryGet()	406
6.172.3 Property Documentation	406
6.172.3.1 Capacity	406
6.172.3.2 Count	407
6.173 NetworkedAttribute Class Reference	407
6.173.1 Detailed Description	407
6.173.2 Constructor & Destructor Documentation	407
6.173.2.1 NetworkedAttribute()	407
6.173.3 Property Documentation	407
6.173.3.1 Default	408
6.174 NetworkedWeavedAttribute Class Reference	408
6.174.1 Detailed Description	408
6.174.2 Constructor & Destructor Documentation	408
6.174.2.1 NetworkedWeavedAttribute()	408
6.174.3 Property Documentation	409
6.174.3.1 WordCount	409

6.174.3.2 WordOffset	409
6.175 NetworkEvents Class Reference	409
6.175.1 Detailed Description	410
6.176 NetworkEvents.ConnectFailedEvent Class Reference	410
6.176.1 Detailed Description	411
6.177 NetworkEvents.ConnectRequestEvent Class Reference	411
6.177.1 Detailed Description	411
6.178 NetworkEvents.CustomAuthenticationResponse Class Reference	411
6.178.1 Detailed Description	411
6.179 NetworkEvents.DisconnectFromServerEvent Class Reference	411
6.179.1 Detailed Description	411
6.180 NetworkEvents.HostMigrationEvent Class Reference	411
6.180.1 Detailed Description	412
6.181 NetworkEvents.InputEvent Class Reference	412
6.181.1 Detailed Description	412
6.182 NetworkEvents.InputPlayerEvent Class Reference	412
6.182.1 Detailed Description	412
6.183 NetworkEvents.ObjectEvent Class Reference	412
6.183.1 Detailed Description	412
6.184 NetworkEvents.ObjectPlayerEvent Class Reference	412
6.184.1 Detailed Description	413
6.185 NetworkEvents.PlayerEvent Class Reference	413
6.185.1 Detailed Description	413
6.186 NetworkEvents.ReliableDataEvent Class Reference	413
6.186.1 Detailed Description	413
6.187 NetworkEvents.ReliableProgressEvent Class Reference	413
6.187.1 Detailed Description	413
6.188 NetworkEvents.RunnerEvent Class Reference	413
6.188.1 Detailed Description	414
6.189 NetworkEvents.SessionListUpdateEvent Class Reference	414
6.189.1 Detailed Description	414
6.190 NetworkEvents.ShutdownEvent Class Reference	414
6.190.1 Detailed Description	414
6.191 NetworkEvents.SimulationMessageEvent Class Reference	414
6.191.1 Detailed Description	414
6.192 NetworkId Struct Reference	414
6.192.1 Detailed Description	416
6.192.2 Member Function Documentation	416
6.192.2.1 CompareTo()	416
6.192.2.2 Equals() [1/2]	416
6.192.2.3 Equals() [2/2]	417
6.192.2.4 GetHashCode()	417

6.192.2.5 operator bool()	417
6.192.2.6 operator"!=="	417
6.192.2.7 operator==()	418
6.192.2.8 Read()	418
6.192.2.9 ToNamePrefixString()	418
6.192.2.10 ToString()	418
6.192.2.11 Write() [1/2]	418
6.192.2.12 Write() [2/2]	419
6.192.3 Member Data Documentation	419
6.192.3.1 ALIGNMENT	419
6.192.3.2 BLOCK_SIZE	419
6.192.3.3 Raw	419
6.192.3.4 SIZE	420
6.192.4 Property Documentation	420
6.192.4.1 Comparer	420
6.192.4.2 IsReserved	420
6.192.4.3 IsValid	420
6.193 NetworkId.EqualityComparer Class Reference	420
6.193.1 Detailed Description	421
6.193.2 Member Function Documentation	421
6.193.2.1 Equals()	421
6.193.2.2 GetHashCode()	421
6.194 NetworkInput Struct Reference	421
6.194.1 Detailed Description	422
6.194.2 Member Function Documentation	422
6.194.2.1 Convert< T >()	422
6.194.2.2 Get< T >()	422
6.194.2.3 Is< T >()	423
6.194.2.4 Set< T >()	423
6.194.2.5 TryGet< T >()	423
6.194.2.6 TrySet< T >()	423
6.194.3 Property Documentation	424
6.194.3.1 Data	424
6.194.3.2 IsValid	424
6.194.3.3 Type	424
6.194.3.4 WordCount	424
6.195 NetworkInputUtils Class Reference	424
6.195.1 Detailed Description	425
6.195.2 Member Function Documentation	425
6.195.2.1 GetMaxWordCount()	425
6.195.2.2 GetType()	425
6.195.2.3 GetTypeKey()	425

6.195.2.4 GetWordCount()	426
6.196 NetworkInputWeavedAttribute Class Reference	426
6.196.1 Detailed Description	426
6.196.2 Constructor & Destructor Documentation	426
6.196.2.1 NetworkInputWeavedAttribute()	426
6.196.3 Property Documentation	427
6.196.3.1 WordCount	427
6.197 NetworkLinkedList< T > Struct Template Reference	427
6.197.1 Detailed Description	429
6.197.2 Constructor & Destructor Documentation	429
6.197.2.1 NetworkLinkedList()	429
6.197.3 Member Function Documentation	429
6.197.3.1 Add() [1/2]	429
6.197.3.2 Add() [2/2]	430
6.197.3.3 Clear()	430
6.197.3.4 Contains() [1/2]	430
6.197.3.5 Contains() [2/2]	430
6.197.3.6 Get()	431
6.197.3.7 GetEnumerator()	431
6.197.3.8 IndexOf() [1/2]	431
6.197.3.9 IndexOf() [2/2]	431
6.197.3.10 Remap()	431
6.197.3.11 Remove() [1/2]	432
6.197.3.12 Remove() [2/2]	432
6.197.3.13 Set()	432
6.197.4 Member Data Documentation	432
6.197.4.1 ELEMENT_WORDS	432
6.197.4.2 META_WORDS	433
6.197.5 Property Documentation	433
6.197.5.1 Capacity	433
6.197.5.2 Count	433
6.197.5.3 this[int index]	433
6.198 NetworkLinkedList< T >.Enumerator Struct Reference	433
6.198.1 Detailed Description	434
6.198.2 Member Function Documentation	434
6.198.2.1 Dispose()	434
6.198.2.2 MoveNext()	434
6.198.2.3 Reset()	434
6.198.3 Property Documentation	434
6.198.3.1 Current	434
6.199 NetworkLinkedListReadOnly< T > Struct Template Reference	435
6.199.1 Detailed Description	436

6.199.2 Member Function Documentation	436
6.199.2.1 Contains() [1/2]	436
6.199.2.2 Contains() [2/2]	436
6.199.2.3 Get()	436
6.199.2.4 IndexOf() [1/2]	437
6.199.2.5 IndexOf() [2/2]	437
6.199.3 Member Data Documentation	437
6.199.3.1 ELEMENT_WORDS	437
6.199.3.2 META_WORDS	437
6.199.4 Property Documentation	438
6.199.4.1 Capacity	438
6.199.4.2 Count	438
6.199.4.3 this[int index]	438
6.200 NetworkLoadSceneParameters Struct Reference	438
6.200.1 Detailed Description	439
6.200.2 Member Function Documentation	439
6.200.2.1 Equals() [1/2]	439
6.200.2.2 Equals() [2/2]	440
6.200.2.3 GetHashCode()	440
6.200.2.4 operator"!=()"	440
6.200.2.5 operator==()	441
6.200.2.6 ToString()	441
6.200.3 Member Data Documentation	441
6.200.3.1 LoadId	441
6.200.4 Property Documentation	441
6.200.4.1 IsActiveOnLoad	441
6.200.4.2 IsLocalPhysics2D	442
6.200.4.3 IsLocalPhysics3D	442
6.200.4.4 IsSingleLoad	442
6.200.4.5 LoadSceneMode	442
6.200.4.6 LoadSceneParameters	442
6.200.4.7 LocalPhysicsMode	442
6.201 NetworkMecanimAnimator Class Reference	443
6.201.1 Detailed Description	443
6.201.2 Member Function Documentation	443
6.201.2.1 SetTrigger() [1/2]	444
6.201.2.2 SetTrigger() [2/2]	444
6.201.3 Member Data Documentation	444
6.201.3.1 Animator	444
6.201.3.2 ApplyTiming	445
6.201.4 Property Documentation	445
6.201.4.1 DynamicWordCount	445

6.202 NetworkObject Class Reference	445
6.202.1 Detailed Description	447
6.202.2 Member Function Documentation	447
6.202.2.1 AssignInputAuthority()	448
6.202.2.2 Awake()	448
6.202.2.3 CopyStateFrom() [1/2]	448
6.202.2.4 CopyStateFrom() [2/2]	448
6.202.2.5 GetLocalAuthorityMask()	448
6.202.2.6 GetWordCount()	449
6.202.2.7 NetworkUnwrap()	449
6.202.2.8 NetworkWrap() [1/2]	449
6.202.2.9 NetworkWrap() [2/2]	450
6.202.2.10 OnDestroy()	450
6.202.2.11 operator NetworkId()	450
6.202.2.12 PriorityLevelDelegate()	451
6.202.2.13 ReleaseStateAuthority()	451
6.202.2.14 RemoveInputAuthority()	451
6.202.2.15 ReplicateToDelegate()	451
6.202.2.16 RequestStateAuthority()	452
6.202.2.17 SetPlayerAlwaysInterested()	452
6.202.3 Member Data Documentation	452
6.202.3.1 Flags	452
6.202.3.2 IsResume	452
6.202.3.3 NestedObjects	453
6.202.3.4 NetworkedBehaviours	453
6.202.3.5 NetworkTypeId	453
6.202.3.6 PriorityCallback	453
6.202.3.7 ReplicateTo	453
6.202.3.8 SortKey	453
6.202.4 Property Documentation	454
6.202.4.1 HasInputAuthority	454
6.202.4.2 HasStateAuthority	454
6.202.4.3 Id	454
6.202.4.4 InputAuthority	454
6.202.4.5 IsInSimulation	454
6.202.4.6 IsProxy	455
6.202.4.7 IsSpawnable	455
6.202.4.8 IsValid	455
6.202.4.9 LastReceiveTick	455
6.202.4.10 Name	455
6.202.4.11 RenderSource	455
6.202.4.12 RenderTime	456

6.202.4.13 RenderTimeframe	456
6.202.4.14 Runner	456
6.202.4.15 StateAuthority	456
6.203 NetworkObjectFlagsExtensions Class Reference	456
6.203.1 Detailed Description	457
6.203.2 Member Function Documentation	457
6.203.2.1 GetVersion()	457
6.203.2.2 IsIgnored()	457
6.203.2.3 IsVersionCurrent()	457
6.203.2.4 SetCurrentVersion()	458
6.203.2.5 SetIgnored()	458
6.204 NetworkObjectGuid Struct Reference	458
6.204.1 Detailed Description	460
6.204.2 Constructor & Destructor Documentation	460
6.204.2.1 NetworkObjectGuid() [1/4]	460
6.204.2.2 NetworkObjectGuid() [2/4]	460
6.204.2.3 NetworkObjectGuid() [3/4]	462
6.204.2.4 NetworkObjectGuid() [4/4]	462
6.204.3 Member Function Documentation	462
6.204.3.1 CompareTo()	462
6.204.3.2 Equals() [1/2]	463
6.204.3.3 Equals() [2/2]	463
6.204.3.4 GetHashCode()	463
6.204.3.5 operator Guid()	464
6.204.3.6 operator NetworkObjectGuid()	464
6.204.3.7 operator NetworkPrefabRef()	464
6.204.3.8 operator"!="()	465
6.204.3.9 operator==()	465
6.204.3.10 Parse()	465
6.204.3.11 ToString() [1/2]	466
6.204.3.12 ToString() [2/2]	466
6.204.3.13 ToUnityGuidString()	466
6.204.3.14 TryParse()	466
6.204.4 Member Data Documentation	466
6.204.4.1 ALIGNMENT	467
6.204.4.2 RawGuidValue	467
6.204.4.3 SIZE	467
6.204.5 Property Documentation	467
6.204.5.1 Empty	467
6.204.5.2 IsValid	467
6.205 NetworkObjectGuid.EqualityComparer Class Reference	467
6.205.1 Detailed Description	468

6.205.2 Member Function Documentation	468
6.205.2.1 Equals()	468
6.205.2.2 GetHashCode()	468
6.206 NetworkObjectHeader Struct Reference	468
6.206.1 Detailed Description	470
6.206.2 Member Function Documentation	470
6.206.2.1 Equals() [1/2]	470
6.206.2.2 Equals() [2/2]	470
6.206.2.3 GetBehaviourChangedTickArray()	470
6.206.2.4 GetDataPointer()	471
6.206.2.5 GetDataWordCount()	471
6.206.2.6 GetHashCode()	471
6.206.2.7 GetMainNetworkTRSPData()	471
6.206.2.8 HasMainNetworkTRSP()	472
6.206.2.9 operator"!="()	472
6.206.2.10 operator==()	472
6.206.2.11 ToString()	473
6.206.3 Member Data Documentation	473
6.206.3.1 _reserved	473
6.206.3.2 BehaviourCount	473
6.206.3.3 Flags	473
6.206.3.4 Id	473
6.206.3.5 InputAuthority	473
6.206.3.6 NestingKey	474
6.206.3.7 NestingRoot	474
6.206.3.8 PLAYER_DATA_WORD	474
6.206.3.9 SIZE	474
6.206.3.10 StateAuthority	474
6.206.3.11 Type	474
6.206.3.12 WordCount	475
6.206.3.13 WORDS	475
6.206.4 Property Documentation	475
6.206.4.1 ByteCount	475
6.207 NetworkObjectHeaderPtr Struct Reference	475
6.207.1 Detailed Description	475
6.207.2 Member Data Documentation	476
6.207.2.1 Ptr	476
6.207.3 Property Documentation	476
6.207.3.1 Id	476
6.207.3.2 Type	476
6.208 NetworkObjectInitializerUnity Class Reference	476
6.208.1 Detailed Description	476

6.208.2 Member Function Documentation	476
6.208.2.1 InitializeNetworkState()	476
6.209 NetworkObjectMeta Class Reference	477
6.209.1 Detailed Description	477
6.209.2 Property Documentation	477
6.209.2.1 Id	477
6.209.2.2 InputAuthority	477
6.209.2.3 StateAuthority	478
6.209.2.4 Type	478
6.210 NetworkObjectNestingKey Struct Reference	478
6.210.1 Detailed Description	479
6.210.2 Constructor & Destructor Documentation	479
6.210.2.1 NetworkObjectNestingKey()	479
6.210.3 Member Function Documentation	479
6.210.3.1 Equals() [1/2]	479
6.210.3.2 Equals() [2/2]	480
6.210.3.3 GetHashCode()	480
6.210.3.4 ToString()	480
6.210.4 Member Data Documentation	481
6.210.4.1 ALIGNMENT	481
6.210.4.2 SIZE	481
6.210.4.3 Value	481
6.210.5 Property Documentation	481
6.210.5.1 IsNone	481
6.210.5.2 IsValid	482
6.211 NetworkObjectNestingKey.EqualityComparer Class Reference	482
6.211.1 Detailed Description	482
6.211.2 Member Function Documentation	482
6.211.2.1 Equals()	482
6.211.2.2 GetHashCode()	482
6.212 NetworkObjectPrefabData Class Reference	483
6.212.1 Detailed Description	483
6.212.2 Member Data Documentation	483
6.212.2.1 Guid	483
6.213 NetworkObjectProviderDummy Class Reference	483
6.213.1 Detailed Description	484
6.214 NetworkObjectReleaseContext Struct Reference	484
6.214.1 Detailed Description	484
6.214.2 Constructor & Destructor Documentation	484
6.214.2.1 NetworkObjectReleaseContext()	484
6.214.3 Member Function Documentation	485
6.214.3.1 ToString()	485

6.214.4 Member Data Documentation	485
6.214.4.1 IsBeingDestroyed	485
6.214.4.2 IsNestedObject	485
6.214.4.3 Object	485
6.214.4.4 Typeld	486
6.215 NetworkObjectSortKeyComparer Class Reference	486
6.215.1 Detailed Description	486
6.215.2 Member Function Documentation	486
6.215.2.1 Compare()	486
6.215.3 Member Data Documentation	487
6.215.3.1 Instance	487
6.216 NetworkObjectSpawnException Class Reference	487
6.216.1 Detailed Description	487
6.216.2 Constructor & Destructor Documentation	487
6.216.2.1 NetworkObjectSpawnException()	487
6.216.3 Property Documentation	488
6.216.3.1 Message	488
6.216.3.2 Status	488
6.216.3.3 Typeld	488
6.217 NetworkObjectTypeld Struct Reference	488
6.217.1 Detailed Description	490
6.217.2 Member Function Documentation	490
6.217.2.1 Equals() [1/2]	490
6.217.2.2 Equals() [2/2]	491
6.217.2.3 FromCustom()	491
6.217.2.4 FromPrefabId()	491
6.217.2.5 FromSceneRefAndObjectIndex()	492
6.217.2.6 FromStruct()	492
6.217.2.7 GetHashCode()	493
6.217.2.8 operator NetworkObjectTypeld()	493
6.217.2.9 operator"!=()"	493
6.217.2.10 operator==()	493
6.217.2.11 ToString()	494
6.217.3 Member Data Documentation	494
6.217.3.1 _value0	494
6.217.3.2 _value1	494
6.217.3.3 ALIGNMENT	494
6.217.3.4 MAX_SCENE_OBJECT_INDEX	494
6.217.3.5 SIZE	494
6.217.4 Property Documentation	494
6.217.4.1 AsCustom	494
6.217.4.2 AsInternalStructId	495

6.217.4.3 AsPrefabId	495
6.217.4.4 AsSceneObjectId	495
6.217.4.5 Comparer	496
6.217.4.6 IsCustom	496
6.217.4.7 IsNone	496
6.217.4.8 IsPrefab	496
6.217.4.9 IsSceneObject	496
6.217.4.10 IsStruct	496
6.217.4.11 IsValid	497
6.217.4.12 Kind	497
6.217.4.13 PlayerData	497
6.218 NetworkObjectTypeId.EqualityComparer Class Reference	497
6.218.1 Detailed Description	497
6.218.2 Member Function Documentation	497
6.218.2.1 Equals()	498
6.218.2.2 GetHashCode()	498
6.219 NetworkPhysicsInfo Struct Reference	499
6.219.1 Detailed Description	499
6.219.2 Member Data Documentation	499
6.219.2.1 SIZE	499
6.219.2.2 TimeScale	499
6.219.2.3 WORD_COUNT	500
6.220 NetworkPrefabAcquireContext Struct Reference	500
6.220.1 Detailed Description	500
6.220.2 Constructor & Destructor Documentation	500
6.220.2.1 NetworkPrefabAcquireContext()	500
6.220.3 Member Data Documentation	501
6.220.3.1 DontDestroyOnLoad	501
6.220.3.2 IsSynchronous	501
6.220.3.3 Meta	501
6.220.3.4 PrefabId	501
6.220.4 Property Documentation	501
6.220.4.1 Data	501
6.220.4.2 HasHeader	502
6.221 NetworkPrefabAttribute Class Reference	502
6.221.1 Detailed Description	502
6.222 NetworkPrefabId Struct Reference	502
6.222.1 Detailed Description	504
6.222.2 Member Function Documentation	504
6.222.2.1 CompareTo() [1/2]	504
6.222.2.2 CompareTo() [2/2]	504
6.222.2.3 Equals() [1/2]	504

6.222.2.4 Equals() [2/2]	504
6.222.2.5 FromIndex()	505
6.222.2.6 FromRaw()	505
6.222.2.7 GetHashCode()	505
6.222.2.8 operator"!="()	505
6.222.2.9 operator==()	505
6.222.2.10 ToString() [1/2]	506
6.222.2.11 ToString() [2/2]	506
6.222.3 Member Data Documentation	506
6.222.3.1 ALIGNMENT	506
6.222.3.2 MAX_INDEX	506
6.222.3.3 RawValue	506
6.222.3.4 SIZE	506
6.222.4 Property Documentation	507
6.222.4.1 AsIndex	507
6.222.4.2 IsNone	507
6.222.4.3 IsValid	507
6.223 NetworkPrefabId.EqualityComparer Class Reference	507
6.223.1 Detailed Description	507
6.223.2 Member Function Documentation	508
6.223.2.1 Equals()	508
6.223.2.2 GetHashCode()	508
6.224 NetworkPrefabInfo Struct Reference	508
6.224.1 Detailed Description	508
6.224.2 Member Data Documentation	509
6.224.2.1 Header	509
6.224.2.2 IsSynchronous	509
6.224.2.3 Prefab	509
6.224.3 Property Documentation	509
6.224.3.1 Data	509
6.224.3.2 HasHeader	509
6.225 NetworkPrefabRef Struct Reference	510
6.225.1 Detailed Description	511
6.225.2 Constructor & Destructor Documentation	511
6.225.2.1 NetworkPrefabRef() [1/4]	511
6.225.2.2 NetworkPrefabRef() [2/4]	511
6.225.2.3 NetworkPrefabRef() [3/4]	512
6.225.2.4 NetworkPrefabRef() [4/4]	512
6.225.3 Member Function Documentation	512
6.225.3.1 CompareTo()	512
6.225.3.2 Equals() [1/2]	513
6.225.3.3 Equals() [2/2]	513

6.225.3.4 GetHashCode()	513
6.225.3.5 operator Guid()	514
6.225.3.6 operator NetworkObjectGuid()	514
6.225.3.7 operator NetworkPrefabRef()	514
6.225.3.8 operator"!=()	515
6.225.3.9 operator==()	515
6.225.3.10 Parse()	515
6.225.3.11 ToString() [1/2]	516
6.225.3.12 ToString() [2/2]	516
6.225.3.13 ToUnityGuidString()	516
6.225.3.14 TryParse()	516
6.225.4 Member Data Documentation	516
6.225.4.1 ALIGNMENT	517
6.225.4.2 RawGuidValue	517
6.225.4.3 SIZE	517
6.225.5 Property Documentation	517
6.225.5.1 Empty	517
6.225.5.2 IsValid	517
6.226 NetworkPrefabRef.EqualityComparer Class Reference	517
6.226.1 Detailed Description	518
6.226.2 Member Function Documentation	518
6.226.2.1 Equals()	518
6.226.2.2 GetHashCode()	518
6.227 NetworkPrefabTable Class Reference	518
6.227.1 Detailed Description	520
6.227.2 Member Function Documentation	520
6.227.2.1 AddInstance()	520
6.227.2.2 AddSource()	520
6.227.2.3 Clear()	520
6.227.2.4 Contains()	521
6.227.2.5 GetEntries()	521
6.227.2.6 GetGuid()	521
6.227.2.7 GetId()	522
6.227.2.8 GetInstancesCount()	522
6.227.2.9 GetSource() [1/2]	522
6.227.2.10 GetSource() [2/2]	523
6.227.2.11 IsAcquired()	523
6.227.2.12 Load()	523
6.227.2.13 RemoveInstance()	524
6.227.2.14 TryAddSource()	524
6.227.2.15 Unload()	525
6.227.2.16 UnloadAll()	525

6.227.2.17 UnloadUnreferenced()	525
6.227.3 Member Data Documentation	526
6.227.3.1 Options	526
6.227.4 Property Documentation	526
6.227.4.1 Prefabs	526
6.227.4.2 Version	526
6.228 NetworkPrefabTableOptions Struct Reference	526
6.228.1 Detailed Description	527
6.228.2 Member Data Documentation	527
6.228.2.1 Default	527
6.228.2.2 UnloadPrefabOnReleasingLastInstance	527
6.228.2.3 UnloadUnusedPrefabsOnShutdown	527
6.229 NetworkProjectConfig Class Reference	527
6.229.1 Detailed Description	529
6.229.2 Member Enumeration Documentation	529
6.229.2.1 PeerModes	529
6.229.2.2 ReplicationFeatures	530
6.229.3 Member Function Documentation	530
6.229.3.1 Deserialize()	530
6.229.3.2 GetExecutionOrder()	531
6.229.3.3 Serialize()	531
6.229.3.4 ToString()	531
6.229.3.5 UnloadGlobal()	531
6.229.4 Member Data Documentation	532
6.229.4.1 AssembliesToWeave	532
6.229.4.2 BuildTypes	532
6.229.4.3 CheckNetworkedPropertiesBeingEmpty	532
6.229.4.4 CheckRpcAttributeUsage	532
6.229.4.5 CurrentTypeId	532
6.229.4.6 CurrentVersion	533
6.229.4.7 DefaultResourceName	533
6.229.4.8 EncryptionConfig	533
6.229.4.9 EnqueueIncompleteSynchronousSpawns	533
6.229.4.10 Heap	533
6.229.4.11 HideNetworkObjectInactivityGuard	533
6.229.4.12 HostMigration	534
6.229.4.13 InvokeRenderInBatchMode	534
6.229.4.14 LagCompensation	534
6.229.4.15 Network	534
6.229.4.16 NetworkConditions	534
6.229.4.17 NetworkIdlsObjectName	534
6.229.4.18 NullChecksForNetworkedProperties	535

6.229.4.19 PeerMode	535
6.229.4.20 PrefabTable	535
6.229.4.21 Simulation	535
6.229.4.22 TimeSynchronizationOverride	535
6.229.4.23 TypeId	535
6.229.4.24 UseSerializableDictionary	536
6.229.4.25 Version	536
6.229.5 Property Documentation	536
6.229.5.1 Global	536
6.230 NetworkProjectConfigAsset Class Reference	536
6.230.1 Detailed Description	537
6.230.2 Member Function Documentation	537
6.230.2.1 OnDisable()	537
6.230.2.2 TryGetGlobal()	537
6.230.2.3 UnloadGlobal()	538
6.230.3 Member Data Documentation	538
6.230.3.1 BehaviourMeta	538
6.230.3.2 Config	538
6.230.3.3 PrefabOptions	538
6.230.3.4 Prefabs	539
6.230.4 Property Documentation	539
6.230.4.1 Global	539
6.230.4.2 IsGlobalLoaded	539
6.231 NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta Struct Reference	539
6.231.1 Detailed Description	539
6.231.2 Member Data Documentation	540
6.231.2.1 ExecutionOrder	540
6.231.2.2 Type	540
6.232 NetworkRNG Struct Reference	540
6.232.1 Detailed Description	541
6.232.2 Constructor & Destructor Documentation	541
6.232.2.1 NetworkRNG()	541
6.232.3 Member Function Documentation	542
6.232.3.1 Next()	542
6.232.3.2 NextExclusive()	542
6.232.3.3 NextInt32()	542
6.232.3.4 NextSingle()	542
6.232.3.5 NextSingleExclusive()	543
6.232.3.6 NextUInt32()	543
6.232.3.7 RangeExclusive() [1/2]	543
6.232.3.8 RangeExclusive() [2/2]	543
6.232.3.9 RangeInclusive() [1/4]	544

6.232.3.10 RangeInclusive() [2/4]	544
6.232.3.11 RangeInclusive() [3/4]	544
6.232.3.12 RangeInclusive() [4/4]	544
6.232.3.13 ToString()	544
6.232.4 Member Data Documentation	545
6.232.4.1 MAX	545
6.232.4.2 SIZE	545
6.232.5 Property Documentation	545
6.232.5.1 Peek	545
6.233 NetworkRpcStaticWeavedInvokerAttribute Class Reference	545
6.233.1 Detailed Description	546
6.233.2 Constructor & Destructor Documentation	546
6.233.2.1 NetworkRpcStaticWeavedInvokerAttribute()	546
6.233.3 Property Documentation	546
6.233.3.1 Key	546
6.234 NetworkRpcWeavedInvokerAttribute Class Reference	546
6.234.1 Detailed Description	547
6.234.2 Constructor & Destructor Documentation	547
6.234.2.1 NetworkRpcWeavedInvokerAttribute()	547
6.234.3 Property Documentation	547
6.234.3.1 Key	547
6.234.3.2 Sources	548
6.234.3.3 Targets	548
6.235 NetworkRunner Class Reference	548
6.235.1 Detailed Description	557
6.235.2 Member Enumeration Documentation	557
6.235.2.1 BuildTypes	557
6.235.2.2 States	557
6.235.3 Member Function Documentation	558
6.235.3.1 AddCallbacks()	558
6.235.3.2 AddGlobal()	558
6.235.3.3 AddPlayerAreaOfInterest()	558
6.235.3.4 Attach() [1/2]	558
6.235.3.5 Attach() [2/2]	559
6.235.3.6 ClearPlayerAreaOfInterest()	559
6.235.3.7 Despawn()	559
6.235.3.8 DestroySingleton< T >()	560
6.235.3.9 Disconnect()	560
6.235.3.10 EnsureRunnerScenelsActive()	560
6.235.3.11 Exists() [1/2]	561
6.235.3.12 Exists() [2/2]	561
6.235.3.13 FindObject()	561

6.235.3.14 GetAllBehaviours()	562
6.235.3.15 GetAllBehaviours< T >() [1/2]	562
6.235.3.16 GetAllBehaviours< T >() [2/2]	562
6.235.3.17 GetAllNetworkObjects()	563
6.235.3.18 GetAreaOfInterestGizmoData()	563
6.235.3.19 GetAvailableRegions()	563
6.235.3.20 GetInputForPlayer< T >()	564
6.235.3.21 GetInstancesEnumerator()	564
6.235.3.22 GetInterfaceListHead()	564
6.235.3.23 GetInterfaceListNext()	565
6.235.3.24 GetInterfaceListPrev()	565
6.235.3.25 GetInterfaceListsCount()	565
6.235.3.26 GetMemorySnapshot()	566
6.235.3.27 GetObjectsInAreaOfInterestForPlayer()	566
6.235.3.28 GetPhysicsScene()	566
6.235.3.29 GetPhysicsScene2D()	567
6.235.3.30 GetPlayerActorId()	567
6.235.3.31 GetPlayerConnectionToken()	567
6.235.3.32 GetPlayerConnectionType()	567
6.235.3.33 GetPlayerObject()	568
6.235.3.34 GetPlayerRtt()	568
6.235.3.35 GetPlayerUserId()	568
6.235.3.36 GetRawInputForPlayer()	569
6.235.3.37 GetResumeSnapshotNetworkObjects()	569
6.235.3.38 GetResumeSnapshotNetworkSceneObjects()	569
6.235.3.39 GetRpcTargetStatus()	570
6.235.3.40 GetRunnerForObject()	570
6.235.3.41 GetRunnerForScene()	570
6.235.3.42 GetSingleton< T >()	570
6.235.3.43 HasSingleton< T >()	571
6.235.3.44 InstantiateInRunnerScene() [1/2]	571
6.235.3.45 InstantiateInRunnerScene() [2/2]	571
6.235.3.46 InstantiateInRunnerScene< T >() [1/2]	571
6.235.3.47 InstantiateInRunnerScene< T >() [2/2]	572
6.235.3.48 InvokeSceneLoadDone()	572
6.235.3.49 InvokeSceneLoadStart()	572
6.235.3.50 IsInterestedIn()	572
6.235.3.51 IsPlayerValid()	572
6.235.3.52 JoinSessionLobby()	573
6.235.3.53 LoadScene() [1/3]	573
6.235.3.54 LoadScene() [2/3]	574
6.235.3.55 LoadScene() [3/3]	574

6.235.3.56 MakeDontDestroyOnLoad()	575
6.235.3.57 MoveGameObjectToSameScene()	575
6.235.3.58 MoveGameObjectToScene()	575
6.235.3.59 MoveToRunnerScene()	576
6.235.3.60 MoveToRunnerScene< T >()	576
6.235.3.61 ObjectDelegate()	576
6.235.3.62 OnBeforeSpawned()	577
6.235.3.63 PushHostMigrationSnapshot()	577
6.235.3.64 RegisterSceneObjects()	577
6.235.3.65 RemoveCallbacks()	578
6.235.3.66 RemoveGlobal()	578
6.235.3.67 RenderInternal()	578
6.235.3.68 SendReliableDataToPlayer()	578
6.235.3.69 SendReliableDataToServer()	579
6.235.3.70 SendRpc() [1/2]	579
6.235.3.71 SendRpc() [2/2]	579
6.235.3.72 SetAreaOfInterestCellSize()	579
6.235.3.73 SetAreaOfInterestGrid()	581
6.235.3.74 SetBehaviourReplicateTo()	581
6.235.3.75 SetBehaviourReplicateToAll()	582
6.235.3.76 SetIsSimulated()	582
6.235.3.77 SetMasterClient()	582
6.235.3.78 SetPlayerAlwaysInterested()	583
6.235.3.79 SetPlayerObject()	583
6.235.3.80 SetSimulateMultiPeerPhysics()	583
6.235.3.81 Shutdown()	584
6.235.3.82 SinglePlayerContinue()	584
6.235.3.83 SinglePlayerPause() [1/2]	584
6.235.3.84 SinglePlayerPause() [2/2]	584
6.235.3.85 Spawn() [1/5]	584
6.235.3.86 Spawn() [2/5]	585
6.235.3.87 Spawn() [3/5]	585
6.235.3.88 Spawn() [4/5]	586
6.235.3.89 Spawn() [5/5]	587
6.235.3.90 Spawn< T >()	587
6.235.3.91 SpawnAsync() [1/5]	588
6.235.3.92 SpawnAsync() [2/5]	589
6.235.3.93 SpawnAsync() [3/5]	589
6.235.3.94 SpawnAsync() [4/5]	590
6.235.3.95 SpawnAsync() [5/5]	590
6.235.3.96 SpawnAsync< T >()	591
6.235.3.97 StartGame()	592

6.235.3.98 TryFindBehaviour()	592
6.235.3.99 TryFindBehaviour< T >()	593
6.235.3.100 TryFindObject()	593
6.235.3.101 TryGetBehaviourStatistics()	594
6.235.3.102 TryGetFusionStatistics()	594
6.235.3.103 TryGetInputForPlayer< T >()	595
6.235.3.104 TryGetNetworkedBehaviourFromNetworkedObjectRef< T >()	595
6.235.3.105 TryGetNetworkedBehaviourId()	595
6.235.3.106 TryGetObjectRefFromNetworkedBehaviour()	596
6.235.3.107 TryGetPhysicsInfo()	596
6.235.3.108 TryGetPlayerObject()	597
6.235.3.109 TryGetSceneInfo()	597
6.235.3.110 TrySetPhysicsInfo()	597
6.235.3.111 TrySpawn() [1/5]	598
6.235.3.112 TrySpawn() [2/5]	598
6.235.3.113 TrySpawn() [3/5]	600
6.235.3.114 TrySpawn() [4/5]	601
6.235.3.115 TrySpawn() [5/5]	601
6.235.3.116 TrySpawn< T >()	602
6.235.3.117 UnloadScene()	602
6.235.3.118 UpdateInternal()	604
6.235.4 Property Documentation	604
6.235.4.1 ActivePlayers	604
6.235.4.2 AuthenticationValues	604
6.235.4.3 BuildType	604
6.235.4.4 CanSpawn	605
6.235.4.5 Config	605
6.235.4.6 CurrentConnectionType	605
6.235.4.7 DeltaTime	605
6.235.4.8 GameMode	605
6.235.4.9 Instances	605
6.235.4.10 IsClient	606
6.235.4.11 IsCloudReady	606
6.235.4.12 IsConnectedToServer	606
6.235.4.13 IsFirstTick	606
6.235.4.14 IsForward	606
6.235.4.15 IsLastTick	606
6.235.4.16 IsPlayer	607
6.235.4.17 IsResimulation	607
6.235.4.18 IsResume	607
6.235.4.19 IsRunning	607
6.235.4.20 IsSceneAuthority	607

6.235.4.21 IsSceneManagerBusy	607
6.235.4.22 IsServer	608
6.235.4.23 IsSharedModeMasterClient	608
6.235.4.24 IsShutdown	608
6.235.4.25 IsSinglePlayer	608
6.235.4.26 IsStarting	608
6.235.4.27 LagCompensation	608
6.235.4.28 LatestServerTick	609
6.235.4.29 LobbyInfo	609
6.235.4.30 LocalAlpha	609
6.235.4.31 LocalPlayer	609
6.235.4.32 LocalRenderTime	609
6.235.4.33 Mode	609
6.235.4.34 NATType	610
6.235.4.35 ObjectProvider	610
6.235.4.36 Prefabs	610
6.235.4.37 ProvideInput	610
6.235.4.38 RemoteRenderTime	610
6.235.4.39 SceneManager	610
6.235.4.40 SessionInfo	611
6.235.4.41 SimulationTime	611
6.235.4.42 SimulationUnityScene	611
6.235.4.43 Stage	611
6.235.4.44 State	611
6.235.4.45 Tick	611
6.235.4.46 TickRate	612
6.235.4.47 TicksExecuted	612
6.235.4.48 Topology	612
6.235.4.49 UserId	612
6.235.5 Event Documentation	612
6.235.5.1 ObjectAcquired	612
6.236 NetworkRunnerCallbackArgs Class Reference	612
6.236.1 Detailed Description	613
6.237 NetworkRunnerCallbackArgs.ConnectRequest Class Reference	613
6.237.1 Detailed Description	613
6.237.2 Member Function Documentation	613
6.237.2.1 Accept()	613
6.237.2.2 Refuse()	614
6.237.2.3 Waiting()	614
6.237.3 Property Documentation	614
6.237.3.1 RemoteAddress	614
6.238 NetworkRunnerUpdaterDefault Class Reference	614

6.238.1 Detailed Description	615
6.238.2 Member Function Documentation	615
6.238.2.1 RegisterInPlayerLoop()	615
6.238.2.2 UnregisterFromPlayerLoop()	615
6.238.3 Member Data Documentation	616
6.238.3.1 RenderSettings	616
6.238.3.2 UpdateSettings	616
6.239 NetworkRunnerUpdaterDefault.NetworkRunnerRender Struct Reference	616
6.239.1 Detailed Description	616
6.240 NetworkRunnerUpdaterDefault.NetworkRunnerUpdate Struct Reference	616
6.240.1 Detailed Description	616
6.241 NetworkRunnerUpdaterDefaultInvokeSettings Struct Reference	617
6.241.1 Detailed Description	617
6.241.2 Member Function Documentation	617
6.241.2.1 Equals() [1/2]	617
6.241.2.2 Equals() [2/2]	618
6.241.2.3 GetHashCode()	618
6.241.2.4 operator"!()"	618
6.241.2.5 operator=="()	619
6.241.2.6 ToString()	619
6.241.3 Member Data Documentation	619
6.241.3.1 AddMode	620
6.241.3.2 ReferencePlayerLoopSystem	620
6.242 NetworkSceneAsyncOp Struct Reference	620
6.242.1 Detailed Description	621
6.242.2 Member Function Documentation	621
6.242.2.1 AddOnCompleted()	621
6.242.2.2 FromAsyncOperation()	621
6.242.2.3 FromCompleted()	622
6.242.2.4 FromCoroutine()	622
6.242.2.5 FromError()	623
6.242.2.6 FromTask()	623
6.242.2.7 GetAwaiter()	624
6.242.3 Member Data Documentation	624
6.242.3.1 SceneRef	624
6.242.4 Property Documentation	624
6.242.4.1 Error	624
6.242.4.2 IsDone	624
6.242.4.3 IsValid	624
6.243 NetworkSceneAsyncOp.Awaiter Struct Reference	625
6.243.1 Detailed Description	625
6.243.2 Constructor & Destructor Documentation	625

6.243.2.1 Awaiter()	625
6.243.3 Member Function Documentation	626
6.243.3.1 GetResult()	626
6.243.3.2 OnCompleted()	626
6.243.4 Property Documentation	626
6.243.4.1 IsCompleted	626
6.244 NetworkSceneInfo Struct Reference	626
6.244.1 Detailed Description	627
6.244.2 Member Function Documentation	628
6.244.2.1 AddSceneRef()	628
6.244.2.2 Equals() [1/2]	628
6.244.2.3 Equals() [2/2]	628
6.244.2.4 GetHashCode()	629
6.244.2.5 IndexOf()	629
6.244.2.6 operator NetworkSceneInfo()	629
6.244.2.7 RemoveSceneRef()	630
6.244.2.8 ToString()	630
6.244.3 Member Data Documentation	630
6.244.3.1 MaxScenes	630
6.244.3.2 SIZE	631
6.244.3.3 WORD_COUNT	631
6.244.4 Property Documentation	631
6.244.4.1 SceneCount	631
6.244.4.2 SceneParams	631
6.244.4.3 Scenes	631
6.244.4.4 Version	631
6.245 NetworkSceneLoadId Struct Reference	632
6.245.1 Detailed Description	632
6.245.2 Constructor & Destructor Documentation	632
6.245.2.1 NetworkSceneLoadId()	632
6.245.3 Member Function Documentation	632
6.245.3.1 Equals() [1/2]	633
6.245.3.2 Equals() [2/2]	633
6.245.3.3 GetHashCode()	633
6.245.4 Member Data Documentation	633
6.245.4.1 Value	633
6.246 NetworkSceneObjectId Struct Reference	634
6.246.1 Detailed Description	634
6.246.2 Member Function Documentation	634
6.246.2.1 Equals() [1/2]	634
6.246.2.2 Equals() [2/2]	635
6.246.2.3 GetHashCode()	635

6.246.2.4 <code>ToString()</code>	635
6.246.3 Member Data Documentation	635
6.246.3.1 <code>ObjectId</code>	636
6.246.3.2 <code>Scene</code>	636
6.246.3.3 <code>SceneLoadId</code>	636
6.246.4 Property Documentation	636
6.246.4.1 <code>IsValid</code>	636
6.247 NetworkSerializeMethodAttribute Class Reference	636
6.247.1 Detailed Description	637
6.247.2 Property Documentation	637
6.247.2.1 <code>MaxSize</code>	637
6.248 NetworkSimulationConfiguration Class Reference	637
6.248.1 Detailed Description	638
6.248.2 Member Function Documentation	638
6.248.2.1 <code>Clone()</code>	638
6.248.2.2 <code>Create()</code>	638
6.248.3 Member Data Documentation	638
6.248.3.1 <code>AdditionalJitter</code>	639
6.248.3.2 <code>AdditionalLoss</code>	639
6.248.3.3 <code>DelayMax</code>	639
6.248.3.4 <code>DelayMin</code>	639
6.248.3.5 <code>DelayPeriod</code>	639
6.248.3.6 <code>DelayShape</code>	639
6.248.3.7 <code>DelayThreshold</code>	640
6.248.3.8 <code>Enabled</code>	640
6.248.3.9 <code>LossChanceMax</code>	640
6.248.3.10 <code>LossChanceMin</code>	640
6.248.3.11 <code>LossChancePeriod</code>	640
6.248.3.12 <code>LossChanceShape</code>	640
6.248.3.13 <code>LossChanceThreshold</code>	641
6.249 NetworkSpawnOp Struct Reference	641
6.249.1 Detailed Description	641
6.249.2 Member Function Documentation	642
6.249.2.1 <code>GetAwaiter()</code>	642
6.249.3 Member Data Documentation	642
6.249.3.1 <code>Runner</code>	642
6.249.4 Property Documentation	642
6.249.4.1 <code>IsFailed</code>	642
6.249.4.2 <code>IsQueued</code>	642
6.249.4.3 <code>IsSpawned</code>	642
6.249.4.4 <code>Object</code>	643
6.249.4.5 <code>Status</code>	643

6.250 NetworkSpawnOp.Awaiter Struct Reference	643
6.250.1 Detailed Description	643
6.250.2 Constructor & Destructor Documentation	643
6.250.2.1 Awaiter()	643
6.250.3 Member Function Documentation	644
6.250.3.1 GetResult()	644
6.250.3.2 OnCompleted()	644
6.250.4 Property Documentation	644
6.250.4.1 IsCompleted	645
6.251 NetworkString< TSize > Class Template Reference	645
6.251.1 Detailed Description	647
6.251.2 Constructor & Destructor Documentation	648
6.251.2.1 NetworkString()	648
6.251.3 Member Function Documentation	648
6.251.3.1 Assign()	648
6.251.3.2 Compare() [1/3]	648
6.251.3.3 Compare() [2/3]	649
6.251.3.4 Compare() [3/3]	649
6.251.3.5 Compare< TOtherSize >() [1/2]	649
6.251.3.6 Compare< TOtherSize >() [2/2]	650
6.251.3.7 Contains() [1/3]	651
6.251.3.8 Contains() [2/3]	651
6.251.3.9 Contains() [3/3]	651
6.251.3.10 Contains< TOtherSize >() [1/2]	652
6.251.3.11 Contains< TOtherSize >() [2/2]	652
6.251.3.12 EndsWith()	653
6.251.3.13 EndsWith< TOtherSize >()	653
6.251.3.14 Equals() [1/4]	654
6.251.3.15 Equals() [2/4]	654
6.251.3.16 Equals() [3/4]	655
6.251.3.17 Equals() [4/4]	655
6.251.3.18 Equals< TOtherSize >() [1/2]	655
6.251.3.19 Equals< TOtherSize >() [2/2]	656
6.251.3.20 Get()	656
6.251.3.21 GetCapacity< TSize >()	657
6.251.3.22 GetCharCount()	657
6.251.3.23 GetEnumerator()	657
6.251.3.24 GetHashCode()	658
6.251.3.25 IndexOf() [1/6]	658
6.251.3.26 IndexOf() [2/6]	658
6.251.3.27 IndexOf() [3/6]	659
6.251.3.28 IndexOf() [4/6]	659

6.251.3.29 IndexOf() [5/6]	660
6.251.3.30 IndexOf() [6/6]	660
6.251.3.31 IndexOf< TOtherSize >() [1/4]	661
6.251.3.32 IndexOf< TOtherSize >() [2/4]	661
6.251.3.33 IndexOf< TOtherSize >() [3/4]	662
6.251.3.34 IndexOf< TOtherSize >() [4/4]	663
6.251.3.35 operator NetworkString< TSize >()	664
6.251.3.36 operator string()	664
6.251.3.37 operator"!=() [1/3]	664
6.251.3.38 operator"!=() [2/3]	665
6.251.3.39 operator"!=() [3/3]	665
6.251.3.40 operator==() [1/3]	665
6.251.3.41 operator==() [2/3]	666
6.251.3.42 operator==() [3/3]	666
6.251.3.43 Set()	667
6.251.3.44 StartsWith()	667
6.251.3.45 StartsWith< TOtherSize >()	667
6.251.3.46 Substring() [1/2]	668
6.251.3.47 Substring() [2/2]	668
6.251.3.48 ToLower()	669
6.251.3.49 ToString()	669
6.251.3.50 ToUpper()	669
6.251.4 Property Documentation	670
6.251.4.1 Capacity	670
6.251.4.2 Length	670
6.251.4.3 this[int index]	670
6.251.4.4 Value	670
6.252 NetworkStructUtils Class Reference	670
6.252.1 Detailed Description	671
6.252.2 Member Function Documentation	671
6.252.2.1 GetWordCount< T >()	671
6.253 NetworkStructWeavedAttribute Class Reference	671
6.253.1 Detailed Description	672
6.253.2 Constructor & Destructor Documentation	672
6.253.2.1 NetworkStructWeavedAttribute()	672
6.253.3 Property Documentation	672
6.253.3.1 WordCount	672
6.254 NetworkTransform Class Reference	672
6.254.1 Detailed Description	673
6.254.2 Member Function Documentation	673
6.254.2.1 SetAreaOfInterestOverride()	673
6.254.2.2 Teleport()	674

6.254.3 Member Data Documentation	674
6.254.3.1 DisableSharedModelInterpolation	674
6.254.3.2 SyncParent	674
6.254.3.3 SyncScale	674
6.254.4 Property Documentation	675
6.254.4.1 AutoUpdateAreaOfInterestOverride	675
6.255 NetworkTRSP Class Reference	675
6.255.1 Detailed Description	676
6.255.2 Member Function Documentation	676
6.255.2.1 Render()	676
6.255.2.2 ResolveAOIOVERRIDE()	676
6.255.2.3 SetAreaOfInterestOverride()	677
6.255.2.4 SetParentTransform()	677
6.255.2.5 Teleport()	677
6.255.3 Property Documentation	677
6.255.3.1 Data	678
6.255.3.2 IsMainTRSP	678
6.255.3.3 State	678
6.256 NetworkTRSPData Struct Reference	678
6.256.1 Detailed Description	679
6.256.2 Member Data Documentation	679
6.256.2.1 AreaOfInterestOverride	679
6.256.2.2 Parent	679
6.256.2.3 Position	679
6.256.2.4 POSITION_OFFSET	680
6.256.2.5 Rotation	680
6.256.2.6 Scale	680
6.256.2.7 SIZE	680
6.256.2.8 TeleportKey	680
6.256.2.9 WORDS	680
6.256.3 Property Documentation	681
6.256.3.1 NonNetworkedParent	681
6.257 NormalizedRectAttribute Class Reference	681
6.257.1 Detailed Description	681
6.257.2 Constructor & Destructor Documentation	681
6.257.2.1 NormalizedRectAttribute()	681
6.257.3 Member Data Documentation	682
6.257.3.1 AspectRatio	682
6.257.3.2 InvertY	682
6.258 OnChangedRenderAttribute Class Reference	682
6.258.1 Detailed Description	682
6.258.2 Constructor & Destructor Documentation	683

6.258.2.1 OnChangedRenderAttribute()	683
6.258.3 Property Documentation	683
6.258.3.1 MethodName	683
6.259 PlayerRef Struct Reference	683
6.259.1 Detailed Description	685
6.259.2 Member Function Documentation	685
6.259.2.1 Equals() [1/2]	685
6.259.2.2 Equals() [2/2]	685
6.259.2.3 FromEncoded()	686
6.259.2.4 FromIndex()	686
6.259.2.5 GetHashCode()	686
6.259.2.6 operator"!="()	686
6.259.2.7 operator==()	687
6.259.2.8 Read()	687
6.259.2.9 ToString()	688
6.259.2.10 Write()	688
6.259.2.11 Write< T >()	688
6.259.3 Member Data Documentation	688
6.259.3.1 MASTER_CLIENT_RAW	689
6.259.3.2 SIZE	689
6.259.4 Property Documentation	689
6.259.4.1 AsIndex	689
6.259.4.2 Comparer	689
6.259.4.3 IsMasterClient	689
6.259.4.4 IsNone	690
6.259.4.5 IsRealPlayer	690
6.259.4.6 MasterClient	690
6.259.4.7 None	690
6.259.4.8 PlayerId	690
6.259.4.9 RawEncoded	690
6.260 PreserveInPluginAttribute Class Reference	691
6.260.1 Detailed Description	691
6.260.2 Constructor & Destructor Documentation	691
6.260.2.1 PreserveInPluginAttribute()	691
6.261 IMessage Interface Reference	691
6.261.1 Detailed Description	691
6.262 Versioning Class Reference	691
6.262.1 Detailed Description	692
6.262.2 Property Documentation	692
6.262.2.1 GetCurrentVersion	692
6.262.2.2 GetProductVersion	692
6.263 Ptr Struct Reference	692

6.263.1 Detailed Description	693
6.263.2 Member Function Documentation	693
6.263.2.1 Equals() [1/2]	693
6.263.2.2 Equals() [2/2]	694
6.263.2.3 GetHashCode()	694
6.263.2.4 operator bool()	694
6.263.2.5 operator"!=()	695
6.263.2.6 operator+()	695
6.263.2.7 operator-()	696
6.263.2.8 operator==()	696
6.263.2.9 ToString()	696
6.263.3 Member Data Documentation	697
6.263.3.1 Address	697
6.263.3.2 SIZE	697
6.263.4 Property Documentation	697
6.263.4.1 Null	697
6.264 Ptr.EqualityComparer Class Reference	697
6.264.1 Detailed Description	697
6.264.2 Member Function Documentation	698
6.264.2.1 Equals()	698
6.264.2.2 GetHashCode()	699
6.265 QuaternionCompressed Struct Reference	699
6.265.1 Detailed Description	700
6.265.2 Member Function Documentation	700
6.265.2.1 Equals() [1/2]	700
6.265.2.2 Equals() [2/2]	701
6.265.2.3 GetHashCode()	701
6.265.2.4 operator Quaternion()	701
6.265.2.5 operator QuaternionCompressed()	702
6.265.2.6 operator"!=()	702
6.265.2.7 operator==()	702
6.265.3 Member Data Documentation	703
6.265.3.1 wEncoded	703
6.265.3.2 xEncoded	703
6.265.3.3 yEncoded	703
6.265.3.4 zEncoded	703
6.265.4 Property Documentation	703
6.265.4.1 W	704
6.265.4.2 X	704
6.265.4.3 Y	704
6.265.4.4 Z	704
6.266 RangeExAttribute Class Reference	704

6.266.1 Detailed Description	705
6.266.2 Constructor & Destructor Documentation	705
6.266.2.1 RangeExAttribute()	705
6.266.3 Member Data Documentation	705
6.266.3.1 ClampMax	705
6.266.3.2 ClampMin	705
6.266.3.3 UseSlider	706
6.266.4 Property Documentation	706
6.266.4.1 Max	706
6.266.4.2 Min	706
6.267 ReadOnlyAttribute Class Reference	706
6.267.1 Detailed Description	706
6.267.2 Property Documentation	707
6.267.2.1 InEditMode	707
6.267.2.2 InPlayMode	707
6.268 ReadWriteUtils Class Reference	707
6.268.1 Detailed Description	708
6.268.2 Member Function Documentation	708
6.268.2.1 ReadFloat()	708
6.268.2.2 ReadNetworkBehaviourRef()	708
6.268.2.3 ReadQuaternion()	709
6.268.2.4 ReadVector2()	709
6.268.2.5 ReadVector3()	709
6.268.2.6 ReadVector4()	710
6.268.2.7 WriteEmptyNetworkBehaviourRef()	710
6.268.2.8 WriteFloat()	710
6.268.2.9 WriteNetworkBehaviourRef()	711
6.268.2.10 WriteNullBehaviourRef()	711
6.268.2.11 WriteQuaternion()	711
6.268.2.12 WriteVector2()	712
6.268.2.13 WriteVector3()	712
6.268.2.14 WriteVector4()	712
6.268.3 Member Data Documentation	713
6.268.3.1 ACCURACY	713
6.269 ReadWriteUtilsForWeaver Class Reference	713
6.269.1 Detailed Description	713
6.269.2 Member Function Documentation	714
6.269.2.1 GetByteArrayHashCode()	714
6.269.2.2 GetByteCountUtf8NoHash()	714
6.269.2.3 GetStringHashCode()	714
6.269.2.4 GetWordCountString()	715
6.269.2.5 ReadBoolean()	715

6.269.2.6 ReadStringUtf32NoHash()	715
6.269.2.7 ReadStringUtf32WithHash()	716
6.269.2.8 ReadStringUtf8NoHash()	716
6.269.2.9 VerifyRawNetworkUnwrap< T >()	717
6.269.2.10 VerifyRawNetworkWrap< T >()	717
6.269.2.11 WriteBoolean()	718
6.269.2.12 WriteStringUtf32NoHash()	718
6.269.2.13 WriteStringUtf32WithHash()	718
6.269.2.14 WriteStringUtf8NoHash()	719
6.270 ReflectionUtils Class Reference	719
6.270.1 Detailed Description	720
6.270.2 Member Function Documentation	720
6.270.2.1 GetAllNetworkBehaviourTypes()	720
6.270.2.2 GetAllSimulationBehaviourTypes()	720
6.270.2.3 GetAllWeavedAssemblies()	720
6.270.2.4 GetAllWeavedNetworkBehaviourTypes()	721
6.270.2.5 GetAllWeavedSimulationBehaviourTypes()	721
6.270.2.6 GetAllWeaverGeneratedTypes()	721
6.270.2.7 GetCustomAttributeOrThrow< T >()	721
6.270.2.8 GetWeavedAttributeOrThrow()	722
6.271 RenderAttribute Class Reference	722
6.271.1 Detailed Description	723
6.271.2 Constructor & Destructor Documentation	723
6.271.2.1 RenderAttribute() [1/2]	723
6.271.2.2 RenderAttribute() [2/2]	723
6.271.3 Property Documentation	723
6.271.3.1 Method	724
6.271.3.2 Source	724
6.271.3.3 Timeframe	724
6.272 RenderTimeline Struct Reference	724
6.272.1 Detailed Description	724
6.272.2 Member Function Documentation	724
6.272.2.1 GetRenderBuffers()	724
6.273 RenderWeavedAttribute Class Reference	725
6.273.1 Detailed Description	725
6.273.2 Constructor & Destructor Documentation	725
6.273.2.1 RenderWeavedAttribute()	725
6.274 ResolveNetworkPrefabSourceAttribute Class Reference	725
6.274.1 Detailed Description	725
6.275 RpcAttribute Class Reference	726
6.275.1 Detailed Description	726
6.275.2 Constructor & Destructor Documentation	727

6.275.2.1 RpcAttribute() [1/2]	727
6.275.2.2 RpcAttribute() [2/2]	727
6.275.3 Member Data Documentation	727
6.275.3.1 MaxPayloadSize	727
6.275.4 Property Documentation	727
6.275.4.1 Channel	727
6.275.4.2 HostMode	728
6.275.4.3 InvokeLocal	728
6.275.4.4 Sources	728
6.275.4.5 Targets	728
6.275.4.6 TickAligned	728
6.276 RpcHeader Struct Reference	728
6.276.1 Detailed Description	729
6.276.2 Member Function Documentation	729
6.276.2.1 Create() [1/2]	729
6.276.2.2 Create() [2/2]	730
6.276.2.3 Read()	730
6.276.2.4 ReadSize()	731
6.276.2.5 ToString()	731
6.276.2.6 Write()	731
6.276.3 Member Data Documentation	731
6.276.3.1 Behaviour	732
6.276.3.2 Method	732
6.276.3.3 Object	732
6.276.3.4 SIZE	732
6.277 RpclInfo Struct Reference	732
6.277.1 Detailed Description	733
6.277.2 Member Function Documentation	733
6.277.2.1 FromLocal()	733
6.277.2.2 FromMessage()	733
6.277.2.3 ToString()	734
6.277.3 Member Data Documentation	734
6.277.3.1 Channel	734
6.277.3.2 IsInvokeLocal	734
6.277.3.3 Source	734
6.277.3.4 Tick	735
6.278 RpclInvokeData Struct Reference	735
6.278.1 Detailed Description	735
6.278.2 Member Function Documentation	735
6.278.2.1 ToString()	735
6.278.3 Member Data Documentation	736
6.278.3.1 Delegate	736

6.278.3.2 Key	736
6.278.3.3 Sources	736
6.278.3.4 Targets	736
6.279 RpcInvokeInfo Struct Reference	736
6.279.1 Detailed Description	737
6.279.2 Member Function Documentation	737
6.279.2.1 ToString()	737
6.279.3 Member Data Documentation	737
6.279.3.1 LocalInvokeResult	737
6.279.3.2 SendCullResult	737
6.279.3.3 SendResult	738
6.280 RpcSendResult Struct Reference	738
6.280.1 Detailed Description	738
6.280.2 Member Function Documentation	738
6.280.2.1 ToString()	738
6.280.3 Member Data Documentation	738
6.280.3.1 MessageSize	739
6.280.3.2 Result	739
6.281 RpcTargetAttribute Class Reference	739
6.281.1 Detailed Description	739
6.281.2 Constructor & Destructor Documentation	739
6.281.2.1 RpcTargetAttribute()	739
6.282 SceneLoadDoneArgs Struct Reference	740
6.282.1 Detailed Description	740
6.282.2 Constructor & Destructor Documentation	740
6.282.2.1 SceneLoadDoneArgs()	740
6.282.3 Member Data Documentation	741
6.282.3.1 RootGameObjects	741
6.282.3.2 Scene	741
6.282.3.3 SceneObjects	741
6.282.3.4 SceneRef	741
6.283 ScenePathAttribute Class Reference	741
6.283.1 Detailed Description	741
6.284 SceneRef Struct Reference	742
6.284.1 Detailed Description	743
6.284.2 Member Function Documentation	743
6.284.2.1 Equals() [1/2]	743
6.284.2.2 Equals() [2/2]	743
6.284.2.3 FromIndex()	744
6.284.2.4 FromPath()	744
6.284.2.5 FromRaw()	745
6.284.2.6 GetHashCode()	745

6.284.2.7 IsPath()	745
6.284.2.8 operator"!="()	746
6.284.2.9 operator==()	746
6.284.2.10 ToString() [1/2]	746
6.284.2.11 ToString() [2/2]	747
6.284.3 Member Data Documentation	747
6.284.3.1 FLAG_ADDRESSABLE	747
6.284.3.2 RawValue	747
6.284.3.3 SIZE	747
6.284.4 Property Documentation	748
6.284.4.1 AsIndex	748
6.284.4.2 AsPathHash	748
6.284.4.3 IsIndex	748
6.284.4.4 IsValid	748
6.284.4.5 None	748
6.285 ScriptHelpAttribute Class Reference	749
6.285.1 Detailed Description	749
6.285.2 Property Documentation	749
6.285.2.1 BackColor	749
6.285.2.2 Hide	749
6.285.2.3 Style	749
6.285.2.4 Url	750
6.286 SerializableDictionary< TKey, TValue > Class Template Reference	750
6.286.1 Detailed Description	751
6.286.2 Member Function Documentation	751
6.286.2.1 Add()	751
6.286.2.2 Clear()	751
6.286.2.3 ContainsKey()	752
6.286.2.4 Create< TKey, TValue >()	752
6.286.2.5 GetEnumerator()	752
6.286.2.6 Remove()	753
6.286.2.7 Reset()	753
6.286.2.8 Store()	753
6.286.2.9 TryGetValue()	753
6.286.2.10 Wrap()	754
6.286.3 Member Data Documentation	754
6.286.3.1 EntryKeyPropertyPath	754
6.286.3.2 ItemsPropertyPath	754
6.286.4 Property Documentation	754
6.286.4.1 Count	755
6.286.4.2 IsReadOnly	755
6.286.4.3 Keys	755

6.286.4.4 this[TKey key]	755
6.286.4.5 Values	755
6.287 SerializableType< BaseType > Struct Template Reference	756
6.287.1 Detailed Description	757
6.287.2 Constructor & Destructor Documentation	757
6.287.2.1 SerializableType() [1/2]	757
6.287.2.2 SerializableType() [2/2]	757
6.287.3 Member Function Documentation	758
6.287.3.1 AsShort()	758
6.287.3.2 Equals() [1/2]	758
6.287.3.3 Equals() [2/2]	758
6.287.3.4 GetHashCode()	758
6.287.3.5 GetShortAssemblyQualifiedName()	758
6.287.3.6 operator SerializableType()	759
6.287.3.7 operator Type()	759
6.287.4 Member Data Documentation	759
6.287.4.1 AssemblyQualifiedName	759
6.287.5 Property Documentation	759
6.287.5.1 IsValid	759
6.287.5.2 Value	759
6.288 SerializableTypeAttribute Class Reference	760
6.288.1 Detailed Description	760
6.288.2 Property Documentation	760
6.288.2.1 BaseType	760
6.288.2.2 UseFullAssemblyQualifiedName	760
6.288.2.3 WarnIfNoPreserveAttribute	760
6.289 SerializeReferenceTypePickerAttribute Class Reference	761
6.289.1 Detailed Description	761
6.289.2 Constructor & Destructor Documentation	761
6.289.2.1 SerializeReferenceTypePickerAttribute()	761
6.289.3 Member Data Documentation	762
6.289.3.1 GroupTypesByNamespace	762
6.289.3.2 ShowFullName	762
6.289.4 Property Documentation	762
6.289.4.1 Types	762
6.290 SessionInfo Class Reference	762
6.290.1 Detailed Description	763
6.290.2 Member Function Documentation	763
6.290.2.1 operator bool()	763
6.290.2.2 ToString()	763
6.290.2.3 UpdateCustomProperties()	764
6.290.3 Property Documentation	764

6.290.3.1 IsOpen	764
6.290.3.2 IsValid	764
6.290.3.3 IsVisible	764
6.290.3.4 MaxPlayers	765
6.290.3.5 Name	765
6.290.3.6 PlayerCount	765
6.290.3.7 Properties	765
6.290.3.8 Region	765
6.291 Simulation Class Reference	765
6.291.1 Detailed Description	768
6.291.2 Member Function Documentation	769
6.291.2.1 AfterSimulation()	769
6.291.2.2 AfterUpdate()	769
6.291.2.3 BeforeFirstTick()	769
6.291.2.4 BeforeSimulation()	769
6.291.2.5 BeforeUpdate()	769
6.291.2.6 GetAreaOfInterestGizmoData()	769
6.291.2.7 GetInputAuthority()	770
6.291.2.8 GetInputForPlayer()	770
6.291.2.9 GetObjectsAndPlayersInAreaOfInterestCell()	770
6.291.2.10 GetObjectsInAreaOfInterestForPlayer()	771
6.291.2.11 GetStateAuthority()	771
6.291.2.12 HasAnyActiveConnections()	771
6.291.2.13 IsInputAuthority()	771
6.291.2.14 IsInterestedIn()	772
6.291.2.15 IsLocalSimulationInputAuthority()	772
6.291.2.16 IsLocalSimulationStateAuthority() [1/2]	773
6.291.2.17 IsLocalSimulationStateAuthority() [2/2]	773
6.291.2.18 IsLocalSimulationStateOrInputSource()	773
6.291.2.19 IsStateAuthority() [1/2]	774
6.291.2.20 IsStateAuthority() [2/2]	774
6.291.2.21 NetworkConnected()	774
6.291.2.22 NetworkDisconnected()	775
6.291.2.23 NetworkReceiveDone()	775
6.291.2.24 NoSimulation()	775
6.291.2.25 TryGetHostPlayer()	775
6.291.2.26 Update()	776
6.291.3 Property Documentation	776
6.291.3.1 ActivePlayers	776
6.291.3.2 Config	776
6.291.3.3 DeltaTime	776
6.291.3.4 InputCount	777

6.291.3.5 IsClient	777
6.291.3.6 IsFirstTick	777
6.291.3.7 IsForward	777
6.291.3.8 IsLastTick	777
6.291.3.9 IsLocalPlayerFirstExecution	778
6.291.3.10 IsMasterClient	778
6.291.3.11 IsPlayer	778
6.291.3.12 IsResimulation	778
6.291.3.13 IsRunning	778
6.291.3.14 IsServer	778
6.291.3.15 IsShutdown	779
6.291.3.16 IsSinglePlayer	779
6.291.3.17 LatestServerTick	779
6.291.3.18 LocalAddress	779
6.291.3.19 LocalAlpha	779
6.291.3.20 LocalPlayer	779
6.291.3.21 Mode	780
6.291.3.22 NetConfigPointer	780
6.291.3.23 ObjectCount	780
6.291.3.24 Objects	780
6.291.3.25 ProjectConfig	780
6.291.3.26 RemoteAlpha	780
6.291.3.27 RemoteTick	781
6.291.3.28 RemoteTickPrevious	781
6.291.3.29 SendDelta	781
6.291.3.30 SendRate	781
6.291.3.31 Stage	781
6.291.3.32 Tick	781
6.291.3.33 TickDeltaDouble	782
6.291.3.34 TickDeltaFloat	782
6.291.3.35 TickPrevious	782
6.291.3.36 TickRate	782
6.291.3.37 TickStride	782
6.291.3.38 Time	782
6.291.3.39 Topology	783
6.292 Simulation.AreaOfInterest Struct Reference	783
6.292.1 Detailed Description	783
6.292.2 Member Function Documentation	783
6.292.2.1 GetCellSize()	784
6.292.2.2 SphereToCells()	784
6.292.2.3 ToCell() [1/2]	784
6.292.2.4 ToCell() [2/2]	785

6.292.2.5 ToCellCenter()	785
6.292.3 Member Data Documentation	785
6.292.3.1 CELL_SIZE	785
6.292.3.2 x	786
6.293 SimulationBehaviour Class Reference	786
6.293.1 Detailed Description	787
6.293.2 Member Function Documentation	787
6.293.2.1 FixedUpdateNetwork()	787
6.293.2.2 Render()	787
6.293.3 Property Documentation	788
6.293.3.1 CanReceiveRenderCallback	788
6.293.3.2 CanReceiveSimulationCallback	788
6.293.3.3 Object	788
6.293.3.4 Runner	788
6.294 SimulationBehaviourAttribute Class Reference	788
6.294.1 Detailed Description	789
6.294.2 Property Documentation	789
6.294.2.1 Modes	789
6.294.2.2 Stages	789
6.294.2.3 Topologies	789
6.295 SimulationBehaviourListScope Struct Reference	790
6.295.1 Detailed Description	790
6.295.2 Member Function Documentation	790
6.295.2.1 Dispose()	790
6.296 SimulationConfig Class Reference	790
6.296.1 Detailed Description	791
6.296.2 Member Enumeration Documentation	791
6.296.2.1 DataConsistency	791
6.296.2.2 InputTransferModes	792
6.296.2.3 SimulationTimeMode	792
6.296.3 Member Data Documentation	792
6.296.3.1 HostMigration	792
6.296.3.2 InputDataWordCount	793
6.296.3.3 InputTransferMode	793
6.296.3.4 ObjectDataConsistency	793
6.296.3.5 PlayerCount	793
6.296.3.6 ReplicationFeatures	793
6.296.3.7 SimulationUpdateTimeMode	793
6.296.3.8 TickRateSelection	794
6.296.3.9 Topology	794
6.296.4 Property Documentation	794
6.296.4.1 AreaOfInterestEnabled	794

6.296.4.2 InputTotalWordCount	794
6.296.4.3 SchedulingEnabled	794
6.296.4.4 SchedulingWithoutAOI	794
6.297 SimulationInput Class Reference	795
6.297.1 Detailed Description	795
6.297.2 Member Function Documentation	795
6.297.2.1 Clear()	795
6.297.2.2 CopyFrom()	796
6.297.3 Property Documentation	796
6.297.3.1 Data	796
6.297.3.2 Header	796
6.297.3.3 Player	796
6.297.3.4 Sent	796
6.298 SimulationInput.Buffer Class Reference	797
6.298.1 Detailed Description	797
6.298.2 Constructor & Destructor Documentation	797
6.298.2.1 Buffer()	797
6.298.3 Member Function Documentation	798
6.298.3.1 Add()	798
6.298.3.2 Clear()	798
6.298.3.3 Contains()	798
6.298.3.4 CopySortedTo()	799
6.298.3.5 Get()	799
6.298.3.6 GetInsertTime()	799
6.298.3.7 GetLastUsedInputHeader()	800
6.298.3.8 Remove()	800
6.298.4 Property Documentation	800
6.298.4.1 Count	800
6.298.4.2 Full	801
6.299 SimulationInputHeader Struct Reference	801
6.299.1 Detailed Description	801
6.299.2 Member Data Documentation	801
6.299.2.1 InterpAlpha	801
6.299.2.2 InterpFrom	802
6.299.2.3 InterpTo	802
6.299.2.4 SIZE	802
6.299.2.5 Tick	802
6.299.2.6 WORD_COUNT	802
6.300 SimulationMessage Struct Reference	802
6.300.1 Detailed Description	805
6.300.2 Member Function Documentation	805
6.300.2.1 Allocate()	805

6.300.2.2 CanAllocateUserPayload()	805
6.300.2.3 Clone()	805
6.300.2.4 GetData()	806
6.300.2.5 GetFlag()	806
6.300.2.6 IsTargeted()	806
6.300.2.7 ReadInt()	807
6.300.2.8 ReadNetworkedObjectRef()	807
6.300.2.9 ReadVector3()	807
6.300.2.10 ReferenceCountAdd()	808
6.300.2.11 ReferenceCountSub()	808
6.300.2.12 SetDummy()	808
6.300.2.13 SetNotTickAligned()	808
6.300.2.14 SetStatic()	808
6.300.2.15 SetTarget()	808
6.300.2.16 SetUnreliable()	809
6.300.2.17 ToString() [1/2]	809
6.300.2.18 ToString() [2/2]	809
6.300.2.19 WriteInt()	809
6.300.2.20 WriteNetworkedObjectRef()	810
6.300.2.21 WriteVector3()	810
6.300.3 Member Data Documentation	810
6.300.3.1 Capacity	810
6.300.3.2 FLAG_DUMMY	810
6.300.3.3 FLAG_INTERNAL	811
6.300.3.4 FLAG_NOT_TICK_ALIGNED	811
6.300.3.5 FLAG_REMOTE	811
6.300.3.6 FLAG_STATIC	811
6.300.3.7 FLAG_TARGET_PLAYER	811
6.300.3.8 FLAG_TARGET_SERVER	811
6.300.3.9 FLAG_UNRELIABLE	812
6.300.3.10 FLAG_USER_FLAGS_START	812
6.300.3.11 FLAG_USER_MESSAGE	812
6.300.3.12 Flags	812
6.300.3.13 FLAGS_RESERVED	812
6.300.3.14 FLAGS_RESERVED_BITS	812
6.300.3.15 MAX_PAYLOAD_SIZE	813
6.300.3.16 Offset	813
6.300.3.17 References	813
6.300.3.18 SIZE	813
6.300.3.19 Source	813
6.300.3.20 Target	813
6.300.3.21 Tick	814

6.300.4 Property Documentation	814
6.300.4.1 IsUnreliable	814
6.301 SimulationMessagePtr Struct Reference	814
6.301.1 Detailed Description	814
6.301.2 Member Data Documentation	814
6.301.2.1 Message	814
6.302 SimulationRuntimeConfig Struct Reference	814
6.302.1 Detailed Description	815
6.302.2 Member Data Documentation	815
6.302.2.1 HostPlayer	815
6.302.2.2 MasterClient	815
6.302.2.3 PlayerMaxCount	815
6.302.2.4 ServerMode	816
6.302.2.5 TickRate	816
6.302.2.6 Topology	816
6.303 NetAddress Struct Reference	816
6.303.1 Detailed Description	817
6.303.2 Member Function Documentation	817
6.303.2.1 Any()	817
6.303.2.2 AnyIPv6()	818
6.303.2.3 CreateFromIpPort()	818
6.303.2.4 FromActorId()	818
6.303.2.5 LocalhostIPv4()	819
6.303.2.6 LocalhostIPv6()	819
6.303.3 Property Documentation	819
6.303.3.1 ActorId	820
6.303.3.2 HasAddress	820
6.303.3.3 IsIPv4	820
6.303.3.4 IsIPv6	820
6.303.3.5 IsRelayAddr	820
6.303.3.6 IsValid	820
6.304 NetBitBufferList Struct Reference	821
6.304.1 Detailed Description	821
6.304.2 Member Function Documentation	821
6.304.2.1 AddFirst()	821
6.304.2.2 AddLast()	822
6.304.2.3 IsInList()	822
6.304.2.4 Remove()	822
6.304.2.5 RemoveHead()	822
6.305 NetCommandAccepted Struct Reference	823
6.305.1 Detailed Description	823
6.306 NetCommandConnect Struct Reference	823

6.306.1 Detailed Description	824
6.307 NetCommandDisconnect Struct Reference	824
6.307.1 Detailed Description	824
6.308 NetCommandHeader Struct Reference	824
6.308.1 Detailed Description	825
6.308.2 Member Function Documentation	825
6.308.2.1 Create()	825
6.309 NetCommandRefused Struct Reference	825
6.309.1 Detailed Description	826
6.310 NetConfig Struct Reference	826
6.310.1 Detailed Description	827
6.310.2 Member Data Documentation	827
6.310.2.1 Address	827
6.310.2.2 ConnectAttempts	827
6.310.2.3 ConnectInterval	827
6.310.2.4 ConnectionDefaultRtt	827
6.310.2.5 ConnectionGroups	828
6.310.2.6 ConnectionPingInterval	828
6.310.2.7 ConnectionSendBuffers	828
6.310.2.8 ConnectionShutdownTime	828
6.310.2.9 ConnectionTimeout	828
6.310.2.10 MaxConnections	828
6.310.2.11 Notify	829
6.310.2.12 OperationExpireTime	829
6.310.2.13 PacketSize	829
6.310.2.14 Simulation	829
6.310.2.15 SocketRecvBuffer	829
6.310.2.16 SocketSendBuffer	829
6.310.3 Property Documentation	830
6.310.3.1 ConnectionsPerGroup	830
6.310.3.2 Defaults	830
6.310.3.3 PacketSizeInBits	830
6.311 StunServers.StunServer Class Reference	830
6.311.1 Detailed Description	831
6.312 StartGameArgs Struct Reference	831
6.312.1 Detailed Description	832
6.312.2 Member Function Documentation	832
6.312.2.1 ToString()	832
6.312.3 Member Data Documentation	833
6.312.3.1 Address	833
6.312.3.2 AuthValues	833
6.312.3.3 Config	833

6.312.3.4 ConnectionToken	833
6.312.3.5 CustomCallbackInterfaces	833
6.312.3.6 CustomLobbyName	834
6.312.3.7 CustomPhotonAppSettings	834
6.312.3.8 CustomPublicAddress	834
6.312.3.9 CustomSTUNServer	834
6.312.3.10 DisableNATPunchthrough	834
6.312.3.11 EnableClientSessionCreation	834
6.312.3.12 GameMode	835
6.312.3.13 HostMigrationResume	835
6.312.3.14 HostMigrationToken	835
6.312.3.15 IsOpen	835
6.312.3.16 IsVisible	835
6.312.3.17 MatchmakingMode	835
6.312.3.18 ObjectInitializer	836
6.312.3.19 ObjectProvider	836
6.312.3.20 OnGameStarted	836
6.312.3.21 PlayerCount	836
6.312.3.22 Scene	836
6.312.3.23 SceneManager	836
6.312.3.24 SessionName	837
6.312.3.25 SessionNameGenerator	837
6.312.3.26 SessionProperties	837
6.312.3.27 StartGameCancellationToken	837
6.312.3.28 Updater	837
6.312.3.29 UseCachedRegions	837
6.312.3.30 UseDefaultPhotonCloudPorts	838
6.313 StartGameResult Class Reference	838
6.313.1 Detailed Description	838
6.313.2 Member Function Documentation	838
6.313.2.1 ToString()	838
6.313.3 Property Documentation	839
6.313.3.1 ErrorMessage	839
6.313.3.2 Ok	839
6.313.3.3 ShutdownReason	839
6.313.3.4 StackTrace	839
6.314 BehaviourStatisticsManager Class Reference	839
6.314.1 Detailed Description	840
6.314.2 Property Documentation	840
6.314.2.1 CompletedSnapshot	840
6.315 BehaviourStatisticsSnapshot Class Reference	840
6.315.1 Detailed Description	840

6.315.2 Property Documentation	840
6.315.2.1 FixedUpdateNetworkExecutionCount	840
6.315.2.2 FixedUpdateNetworkExecutionTime	841
6.315.2.3 RenderExecutionCount	841
6.315.2.4 RenderExecutionTime	841
6.316 FusionStatisticsManager Class Reference	841
6.316.1 Detailed Description	841
6.316.2 Property Documentation	841
6.316.2.1 CompleteSnapshot	842
6.316.2.2 ObjectStatisticsManager	842
6.317 FusionStatisticsSnapshot Class Reference	842
6.317.1 Detailed Description	843
6.317.2 Property Documentation	843
6.317.2.1 ForwardTicks	843
6.317.2.2 GeneralAllocMemoryFreeInBytes	843
6.317.2.3 GeneralAllocMemoryUsedInBytes	844
6.317.2.4 InBandwidth	844
6.317.2.5 InObjectUpdates	844
6.317.2.6 InPackets	844
6.317.2.7 InputInBandwidth	844
6.317.2.8 InputOutBandwidth	844
6.317.2.9 InputReceiveDelta	845
6.317.2.10 InterpolationOffset	845
6.317.2.11 InterpolationSpeed	845
6.317.2.12 ObjectsAllocMemoryFreeInBytes	845
6.317.2.13 ObjectsAllocMemoryUsedInBytes	845
6.317.2.14 OutBandwidth	845
6.317.2.15 OutObjectUpdates	846
6.317.2.16 OutPackets	846
6.317.2.17 Resimulations	846
6.317.2.18 RoundTripTime	846
6.317.2.19 SimulationSpeed	846
6.317.2.20 SimulationTimeOffset	846
6.317.2.21 StateReceiveDelta	847
6.317.2.22 TimeResets	847
6.317.2.23 WordsReadCount	847
6.317.2.24 WordsReadSize	847
6.317.2.25 WordsWrittenCount	847
6.317.2.26 WordsWrittenSize	847
6.318 LagCompensationStatisticsSnapshot Class Reference	848
6.318.1 Detailed Description	848
6.318.2 Property Documentation	848

6.318.2.1 AddOnBufferTime	848
6.318.2.2 AddOnBVHTime	849
6.318.2.3 AdvanceBufferTime	849
6.318.2.4 BVHMaxDeep	849
6.318.2.5 BVHNodesCount	849
6.318.2.6 HitboxesCount	849
6.318.2.7 RefitBVHTime	849
6.318.2.8 TotalElapsedTime	850
6.318.2.9 UpdateBufferTime	850
6.318.2.10 UpdateBVHTime	850
6.319 MemoryStatisticsSnapshot Struct Reference	850
6.319.1 Detailed Description	851
6.319.2 Member Enumeration Documentation	851
6.319.2.1 TargetAllocator	851
6.319.3 Member Data Documentation	851
6.319.3.1 BUCKET_COUNT	851
6.319.3.2 BucketFreeBlocksCount	851
6.319.3.3 BucketFullBlocksCount	851
6.319.3.4 BucketUsedBlocksCount	852
6.319.3.5 TotalFreeBlocks	852
6.320 NetworkObjectStatisticsManager Class Reference	852
6.320.1 Detailed Description	852
6.320.2 Member Function Documentation	852
6.320.2.1 ClearMonitoredNetworkObjects()	852
6.320.2.2 GetNetworkObjectStatistics()	853
6.320.2.3 MonitorNetworkObjectStatistics()	853
6.321 NetworkObjectStatisticsSnapshot Class Reference	853
6.321.1 Detailed Description	853
6.321.2 Property Documentation	854
6.321.2.1 InBandwidth	854
6.321.2.2 InPackets	854
6.321.2.3 OutBandwidth	854
6.321.2.4 OutPackets	854
6.322 Tick Struct Reference	854
6.322.1 Detailed Description	856
6.322.2 Member Function Documentation	856
6.322.2.1 CompareTo()	856
6.322.2.2 Equals() [1/2]	856
6.322.2.3 Equals() [2/2]	856
6.322.2.4 GetHashCode()	857
6.322.2.5 Next()	857
6.322.2.6 operator bool()	857

6.322.2.7 operator int()	858
6.322.2.8 operator Tick()	858
6.322.2.9 operator"!=()"	858
6.322.2.10 operator<()	859
6.322.2.11 operator<=()	859
6.322.2.12 operator==()	859
6.322.2.13 operator>()	859
6.322.2.14 operator>=()	859
6.322.2.15 ToString()	860
6.322.3 Member Data Documentation	860
6.322.3.1 ALIGNMENT	860
6.322.3.2 Raw	860
6.322.3.3 SIZE	860
6.323 Tick.EqualityComparer Class Reference	860
6.323.1 Detailed Description	861
6.323.2 Member Function Documentation	861
6.323.2.1 Equals()	861
6.323.2.2 GetHashCode()	861
6.324 Tick.RelationalComparer Class Reference	861
6.324.1 Detailed Description	862
6.324.2 Member Function Documentation	862
6.324.2.1 Compare()	862
6.325 TickAccumulator Struct Reference	862
6.325.1 Detailed Description	863
6.325.2 Member Function Documentation	863
6.325.2.1 AddTicks()	863
6.325.2.2 AddTime()	864
6.325.2.3 Alpha()	864
6.325.2.4 ConsumeTick()	864
6.325.2.5 Start()	865
6.325.2.6 StartNew()	865
6.325.2.7 Stop()	865
6.325.3 Property Documentation	865
6.325.3.1 Pending	865
6.325.3.2 Remainder	865
6.325.3.3 Running	865
6.325.3.4 TimeScale	865
6.326 TickRate Struct Reference	866
6.326.1 Detailed Description	867
6.326.2 Member Enumeration Documentation	867
6.326.2.1 ValidateResult	867
6.326.3 Member Function Documentation	868

6.326.3.1 ClampSelection()	868
6.326.3.2 Get()	868
6.326.3.3 GetDivisor()	868
6.326.3.4 GetTickRate()	869
6.326.3.5 Init()	869
6.326.3.6 IsValid() [1 / 2]	870
6.326.3.7 IsValid() [2 / 2]	870
6.326.3.8 Resolve()	870
6.326.3.9 ToArray()	871
6.326.3.10 Validate()	871
6.326.3.11 ValidateSelection()	871
6.326.4 Property Documentation	872
6.326.4.1 Available	872
6.326.4.2 Client	872
6.326.4.3 Count	872
6.326.4.4 this[int index]	872
6.327 TickRate.Resolved Struct Reference	873
6.327.1 Detailed Description	874
6.327.2 Member Data Documentation	874
6.327.2.1 Client	874
6.327.2.2 ClientSend	874
6.327.2.3 Server	874
6.327.2.4 ServerSend	874
6.327.2.5 SIZE	874
6.327.2.6 WORDS	875
6.327.3 Property Documentation	875
6.327.3.1 ClientSendDelta	875
6.327.3.2 ClientTickDelta	875
6.327.3.3 ClientTickStride	875
6.327.3.4 ServerSendDelta	875
6.327.3.5 ServerTickDelta	875
6.327.3.6 ServerTickStride	876
6.328 TickRate.Selection Struct Reference	876
6.328.1 Detailed Description	876
6.328.2 Member Data Documentation	876
6.328.2.1 Client	876
6.328.2.2 ClientSendIndex	876
6.328.2.3 ServerIndex	877
6.328.2.4 ServerSendIndex	877
6.329 TickTimer Struct Reference	877
6.329.1 Detailed Description	878
6.329.2 Member Function Documentation	878

6.329.2.1 CreateFromSeconds()	878
6.329.2.2 CreateFromTicks()	878
6.329.2.3 Expired()	879
6.329.2.4 ExpiredOrNotRunning()	879
6.329.2.5 RemainingTicks()	880
6.329.2.6 RemainingTime()	880
6.329.2.7 ToString()	880
6.329.3 Property Documentation	880
6.329.3.1 IsRunning	881
6.329.3.2 None	881
6.329.3.3 TargetTick	881
6.330 TimeSyncConfiguration Class Reference	881
6.330.1 Detailed Description	881
6.330.2 Member Data Documentation	881
6.330.2.1 MaxLateInputs	882
6.330.2.2 MaxLateSnapshots	882
6.330.2.3 RedundantInputs	882
6.330.2.4 RedundantSnapshots	882
6.330.2.5 SampleWindowSeconds	882
6.331 ToggleLeftAttribute Class Reference	882
6.331.1 Detailed Description	883
6.332 UnitAttribute Class Reference	883
6.332.1 Detailed Description	883
6.332.2 Constructor & Destructor Documentation	883
6.332.2.1 UnitAttribute()	883
6.332.3 Property Documentation	884
6.332.3.1 Unit	884
6.333 UnityAddressablesRuntimeKeyAttribute Class Reference	884
6.333.1 Detailed Description	884
6.334 UnityAssetGuidAttribute Class Reference	884
6.334.1 Detailed Description	884
6.335 UnityContextMenuItemAttribute Class Reference	884
6.335.1 Detailed Description	885
6.335.2 Constructor & Destructor Documentation	885
6.335.2.1 UnityContextMenuItemAttribute()	885
6.335.3 Property Documentation	885
6.335.3.1 order	885
6.336 UnityDelayedAttribute Class Reference	885
6.336.1 Detailed Description	886
6.336.2 Property Documentation	886
6.336.2.1 order	886
6.337 UnityFormerlySerializedAsAttribute Class Reference	886

6.337.1 Detailed Description	886
6.337.2 Constructor & Destructor Documentation	886
6.337.2.1 UnityFormerlySerializedAsAttribute()	886
6.338 UnityHeaderAttribute Class Reference	887
6.338.1 Detailed Description	887
6.338.2 Constructor & Destructor Documentation	887
6.338.2.1 UnityHeaderAttribute()	887
6.338.3 Property Documentation	887
6.338.3.1 order	887
6.339 UnityMinAttribute Class Reference	888
6.339.1 Detailed Description	888
6.339.2 Constructor & Destructor Documentation	888
6.339.2.1 UnityMinAttribute()	888
6.339.3 Property Documentation	888
6.339.3.1 order	888
6.340 UnityMultilineAttribute Class Reference	889
6.340.1 Detailed Description	889
6.340.2 Property Documentation	889
6.340.2.1 order	889
6.341 UnityNonReorderableAttribute Class Reference	889
6.341.1 Detailed Description	889
6.341.2 Property Documentation	889
6.341.2.1 order	890
6.342 UnityNonSerializedAttribute Class Reference	890
6.342.1 Detailed Description	890
6.343 UnityRangeAttribute Class Reference	890
6.343.1 Detailed Description	890
6.343.2 Constructor & Destructor Documentation	890
6.343.2.1 UnityRangeAttribute()	891
6.343.3 Property Documentation	891
6.343.3.1 order	891
6.344 UnityResourcePathAttribute Class Reference	891
6.344.1 Detailed Description	891
6.344.2 Constructor & Destructor Documentation	891
6.344.2.1 UnityResourcePathAttribute()	891
6.344.3 Property Documentation	892
6.344.3.1 ResourceType	892
6.345 UnitySerializeField Class Reference	892
6.345.1 Detailed Description	892
6.346 UnitySerializeReference Class Reference	892
6.346.1 Detailed Description	892
6.347 UnitySpaceAttribute Class Reference	892

6.347.1 Detailed Description	893
6.347.2 Constructor & Destructor Documentation	893
6.347.2.1 UnitySpaceAttribute()	893
6.347.3 Property Documentation	893
6.347.3.1 order	893
6.348 UnityTooltipAttribute Class Reference	893
6.348.1 Detailed Description	894
6.348.2 Constructor & Destructor Documentation	894
6.348.2.1 UnityTooltipAttribute()	894
6.348.3 Property Documentation	894
6.348.3.1 order	894
6.349 Vector2Compressed Struct Reference	894
6.349.1 Detailed Description	895
6.349.2 Member Function Documentation	895
6.349.2.1 Equals() [1/2]	895
6.349.2.2 Equals() [2/2]	896
6.349.2.3 GetHashCode()	896
6.349.2.4 operator Vector2()	896
6.349.2.5 operator Vector2Compressed()	897
6.349.2.6 operator"!="()	897
6.349.2.7 operator==()	898
6.349.3 Member Data Documentation	898
6.349.3.1 xEncoded	898
6.349.3.2 yEncoded	898
6.349.4 Property Documentation	898
6.349.4.1 X	898
6.349.4.2 Y	899
6.350 Vector3Compressed Struct Reference	899
6.350.1 Detailed Description	900
6.350.2 Member Function Documentation	900
6.350.2.1 Equals() [1/2]	900
6.350.2.2 Equals() [2/2]	900
6.350.2.3 GetHashCode()	901
6.350.2.4 operator Vector2()	901
6.350.2.5 operator Vector3()	901
6.350.2.6 operator Vector3Compressed() [1/2]	902
6.350.2.7 operator Vector3Compressed() [2/2]	902
6.350.2.8 operator"!="()	902
6.350.2.9 operator==()	903
6.350.3 Member Data Documentation	903
6.350.3.1 xEncoded	903
6.350.3.2 yEncoded	903

6.350.3.3 zEncoded	904
6.350.4 Property Documentation	904
6.350.4.1 X	904
6.350.4.2 Y	904
6.350.4.3 Z	904
6.351 Vector4Compressed Struct Reference	904
6.351.1 Detailed Description	905
6.351.2 Member Function Documentation	905
6.351.2.1 Equals() [1/2]	905
6.351.2.2 Equals() [2/2]	906
6.351.2.3 GetHashCode()	906
6.351.2.4 operator Vector4()	906
6.351.2.5 operator Vector4Compressed()	907
6.351.2.6 operator"!="()	907
6.351.2.7 operator==()	908
6.351.3 Member Data Documentation	908
6.351.3.1 wEncoded	908
6.351.3.2 xEncoded	908
6.351.3.3 yEncoded	908
6.351.3.4 zEncoded	909
6.351.4 Property Documentation	909
6.351.4.1 W	909
6.351.4.2 X	909
6.351.4.3 Y	909
6.351.4.4 Z	909
6.352 WarnIfAttribute Class Reference	909
6.352.1 Detailed Description	910
6.352.2 Constructor & Destructor Documentation	910
6.352.2.1 WarnIfAttribute()	910
6.352.3 Member Data Documentation	910
6.352.3.1 AsBox	911
6.352.3.2 Message	911
6.353 WeaverGeneratedAttribute Class Reference	911
6.353.1 Detailed Description	911
Index	913

Chapter 1

Photon Fusion API Documentation

Welcome to the Photon [Fusion](#) API online documentation.

1.1 Main Fusion API

Class	Description
Fusion.NetworkRunner	Represents a Server or Client Simulation
Fusion.NetworkObject	This stores the object's network identity and manages the object's state and input authority
Fusion.NetworkProjectConfig	The core Fusion config file that is shared with all peers at startup
Fusion.NetworkBehaviour	Base class for Fusion network components, which are associated with a Fusion.NetworkObject
Fusion.NetworkTransform	Replicates a Unity Transform's position and rotation state

Chapter 2

Photon Fusion Overview

Fusion is a new high performance state synchronization networking library for Unity. Fusion is built with simplicity in mind to integrate naturally into the common Unity workflow, while also offering advanced features like data compression, client-side prediction and lag compensation out of the box.

Behind the covers, Fusion relies on a state-of-the-art compression algorithm to reduce bandwidth requirements with minimal CPU overhead. Data is transferred as partial chunks with eventual consistency. A fully configurable area-of-interest system is supplied to allow support for very high player counts.

The Fusion API is designed to be similar to regular Unity MonoBehaviour code. For example, RPCs and network state is defined with attributes on methods and properties of MonoBehaviour with no need for explicit serialization code and network objects can be defined as prefabs using all of Unity's most recent prefab features like nesting and variants.

Inputs, Networked Properties and RPCs provide the foundation for writing gameplay code with Fusion.

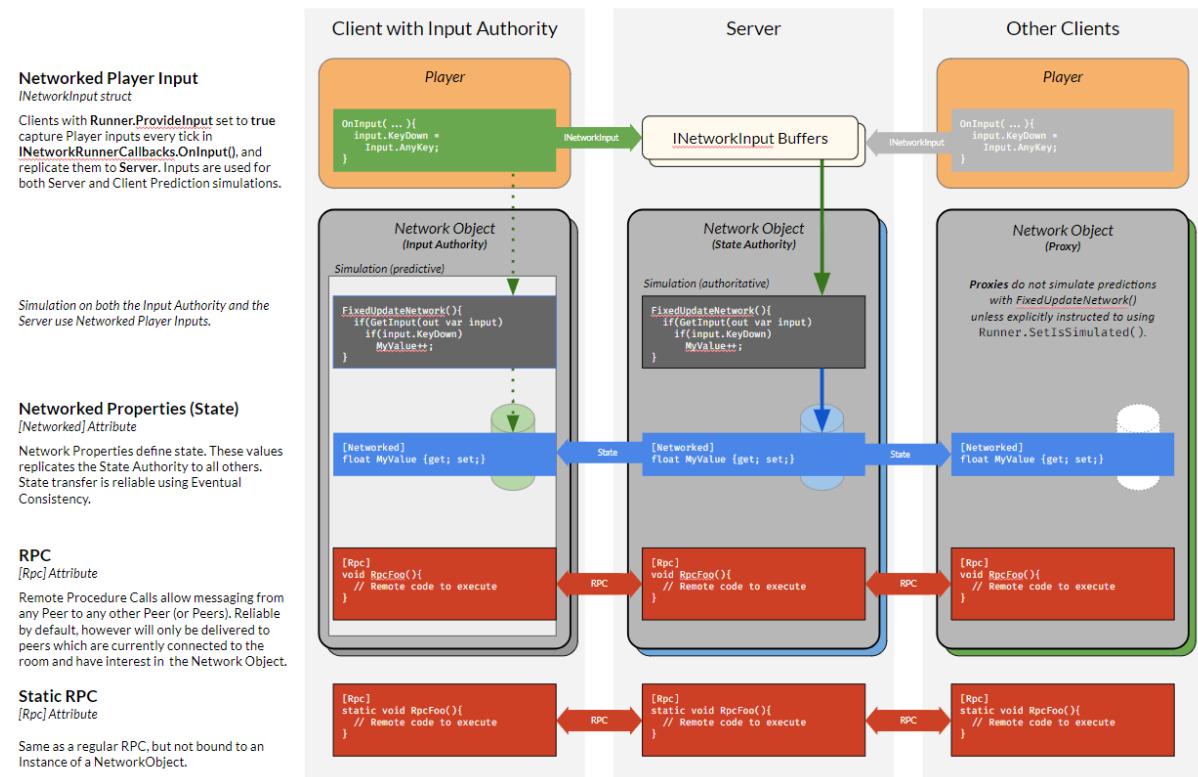


Figure 2.1 Overview of the Core Fusion APIs

Using a Networked Property in **Fusion**:

```
[Networked] public byte life { get; set; }
```

Using Network Input for client-side predicted movement:

```
public override void FixedUpdateNetwork
{
    if (GetInput(out NetworkInputData data))
    {
        data.direction.Normalize(); // normalize to prevent cheating with impossible inputs
        _characterController.Move(5 * data.direction * Runner.deltaTime);
    }
}
```

Declaring a Remote Procedure Call (RPC) in **Fusion**:

```
[Rpc(RpcSources.InputAuthority, RpcTargets.StateAuthority)]
public void RPC_Configure(string name, Color color)
{
    playerName = name;
    playerColor = color;
}
```

2.1 Choosing the Right Mode

Fusion supports two fundamentally different network topologies with the same API as well as a single player mode with no network connection.

The first step when starting with **Fusion** is to chose between Server/Host and Shared mode.

The **Quadrant** provides a good starting point for deciding what mode is right for your application.

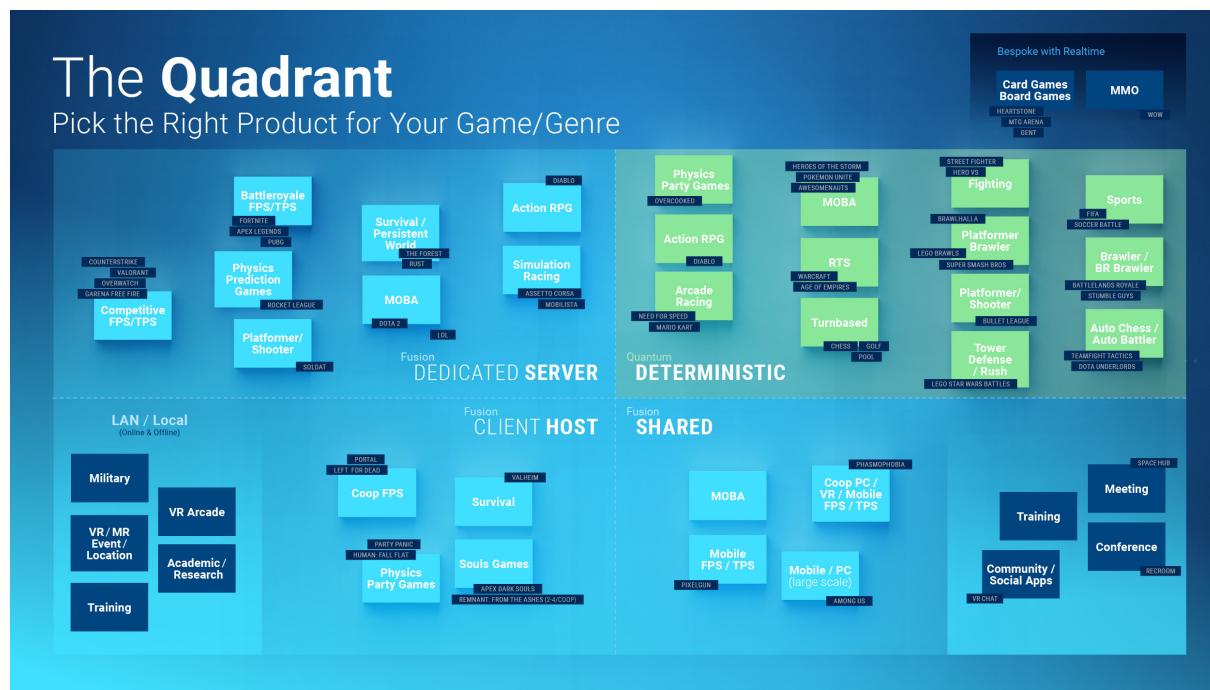


Figure 2.2 The Quadrant

2.2 Topology Differences

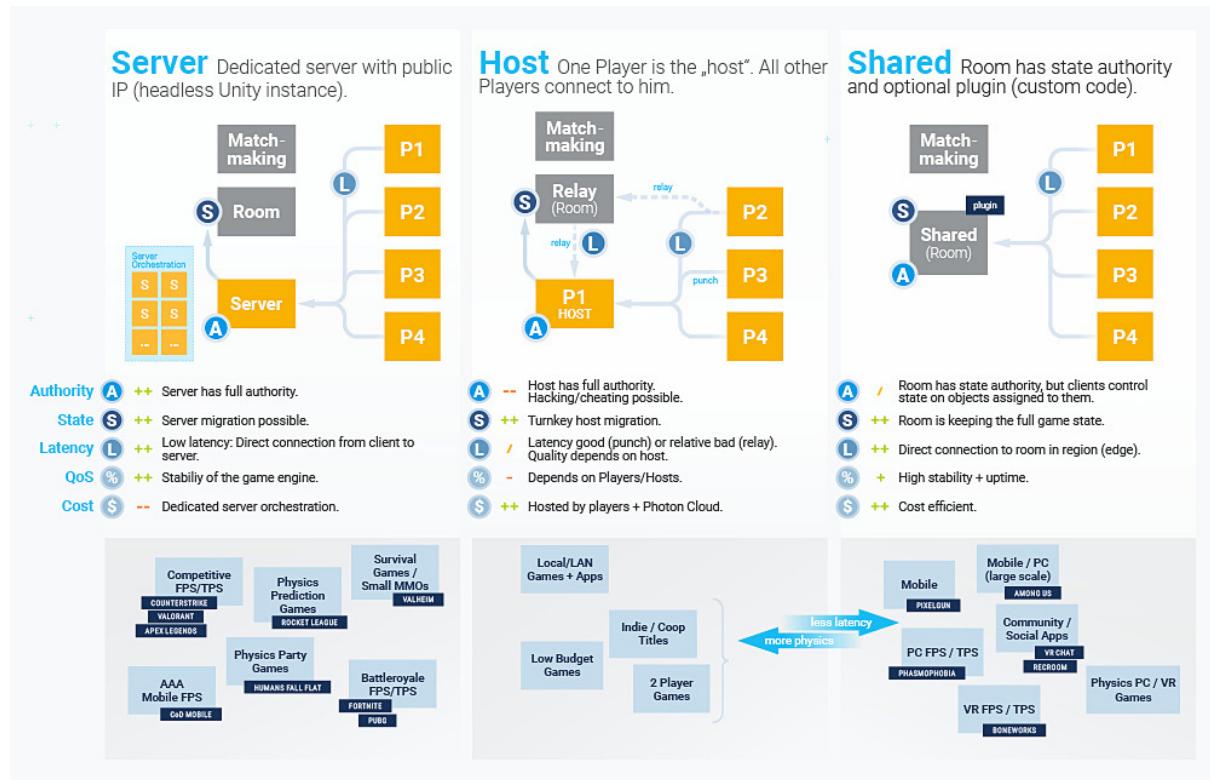


Figure 2.3 Fusion Network Topologies

2.2.1 Server Mode

In Server Mode the server has full and exclusive State Authority over all objects, no exceptions.

Clients can only modify networked objects by sending their input to the server (and have the server react to that input) or by requesting a change using an RPC.

The server application is built from the Unity project and runs a full headless Unity build. This headless build needs to be hosted on a server machine or a cloud hosted server. Photon does not provide servers for hosting a dedicated fusion server application.

2.2.1.1 Client Side Prediction

Client Side Prediction is a popular multiplayer architecture in which clients use their own inputs to predict their movement before receiving confirmation from the server. This allows the gameplay to feel snappy and hides latency.

In **Fusion** Server Mode, any changes a client makes directly to the networked state is only a local prediction, which will be overridden with actual authoritative snapshots from the server when those are received. This is known as reconciliation, as the client is rolled back to the server-provided state and re-simulated forward to the local (predicted) tick.

If previous predictions were accurate, this process is seamless. If not, the state will be updated and because the network state is separate from the rendering state, the rendering may either snap to this new state or use various forms of interpolation, error correction and smoothing to reduce the visual artifacts caused by the correction.

2.2.2 Host Mode

In Host Mode, the host acts as both a server and a client. The host has a local player and polls input for it and interpolates on rendering as expected of a client.

Overall the mode is equivalent to a dedicated server albeit much cheaper to run as no dedicated server hosting costs are incurred. This benefit comes in expense of losing a trustworthy authority; in other words a rogue host can cheat.

When running hosted mode from behind a firewall or a router, the Photon cloud transparently provides UDP punch through or package relay as needed,

Since the session is owned by the host, it will be lost if the host disconnects. [Fusion](#) does provide a host migration mechanism to allow transfer of network authority to a new client in the event that the current host is disconnected. Do note that, unlike Shared Mode, this requires special handling in client code.

2.2.3 Shared Mode

In shared mode, authority over network objects is distributed among all clients. Specifically, each client initially has State Authority over objects they spawn, but are free to release that State Authority to other clients. Optionally, clients may be allowed to take State Authority at will.

In shared mode features such as client side prediction and rollback are not available. Simulation always moves forward at the same tick rate on all clients.

The Shared Mode network session is owned by the Photon cloud and remains alive as long as any client is connected to it. The Photon cloud serves as a package relay and has full access to the network state with no need to run Unity, allowing for lightweight server logic and data validation (e.g. cheat protection) to be implemented without the need to spin up dedicated server hardware.

For those coming from Photon Unity Networking (PUN). Shared mode is in many ways similar to PUN, albeit more feature complete, faster, and with no run-time allocation overhead.

2.2.4 Cost

The same CCU costs apply for all modes. The server and clients all have to be connected to the Photon Cloud at all times for connection management. In Server Mode there are additional costs for hosting the dedicated server on a cloud service or your own hardware.

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Allocator	75
Allocator.Config	77
AssetObject	93
TaskManager	93
AuthorityMasks	95
Behaviour	96
Hitbox	159
NetworkEvents	409
NetworkObject	445
NetworkObjectPrefabData	483
NetworkRunner	548
SimulationBehaviour	786
HitboxManager	163
NetworkBehaviour	304
HitboxRoot	177
NetworkMecanimAnimator	443
NetworkTRSP	675
NetworkTransform	672
CapacityAttribute	99
DecoratingPropertyAttribute	100
ArrayLengthAttribute	90
DisplayNameAttribute	104
DolfAttributeBase	105
DrawIfAttribute	109
ErrorIfAttribute	127
WarnIfAttribute	909
FieldEditorButtonAttribute	129
HideArrayElementLabelAttribute	159
InlineHelpAttribute	221
ReadOnlyAttribute	706
SerializeReferenceTypePickerAttribute	761
UnitAttribute	883
DefaultForPropertyAttribute	101
DrawerPropertyAttribute	108

AssemblyNameAttribute	92
BinaryDataAttribute	98
BitSetAttribute	98
DisplayAsEnumAttribute	103
DrawInlineAttribute	110
ExpandableEnumAttribute	128
LayerAttribute	278
LayerMatrixAttribute	278
MaxStringByteCountAttribute	284
RangeExAttribute	704
ScenePathAttribute	741
ToggleLeftAttribute	882
UnityAssetGuidAttribute	884
UnityResourcePathAttribute	891
DynamicHeap	110
DynamicHeap.Ignore	114
DynamicHeapInstance	114
EditorButtonAttribute	119
IDataEncryption	124
DataEncryptor	123
FieldsMask< T >	130
FixedSize< T >	133
FixedSize< T >.Enumerator	140
FixedBufferPropertyAttribute	142
FixedStorage	143
FloatUtils	147
FusionGlobalScriptableObjectAttribute	152
FusionGlobalScriptableObjectLoadResult	153
FusionGlobalScriptableObjectSourceAttribute	155
FusionMonoBehaviour	157
FusionScriptableObject	157
FusionGlobalScriptableObject< NetworkProjectConfigAsset >	149
NetworkProjectConfigAsset	536
FusionGlobalScriptableObject< T >	149
HeapConfiguration	157
HostMigrationConfig	183
HostMigrationToken	184
IAfterAllTicks	184
NetworkMecanimAnimator	443
NetworkTransform	672
IAfterClientPredictionReset	185
IAfterHostMigration	185
IAfterRender	186
IAfterSpawned	186
IAfterTick	187
HitboxManager	163
IAfterUpdate	188
IAsyncOperation	188
ICoroutine	194
IBeforeAllTicks	189
NetworkTransform	672
IBeforeClientPredictionReset	190
IBeforeCopyPreviousState	191
NetworkTransform	672
IBeforeHitboxRegistration	191
IBeforeSimulation	192

HitboxManager	163
IBeforeTick	193
IBeforeUpdate	193
IDespawned	194
NetworkBehaviour	304
IElementReaderWriter< T >	195
IFixedStorage	197
_128	67
_16	68
_2	69
_256	69
_32	70
_4	71
_512	72
_64	73
_8	74
IInputAuthorityGained	197
IInputAuthorityLost	198
IInterestEnter	198
IInterestExit	199
ILocalPrefabCreated	200
INetworkArray	200
NetworkArray< T >	293
INetworkAssetSource< T >	202
INetworkDictionary	203
NetworkDictionary< K, V >	395
INetworkInput	204
INetworkLinkedList	204
NetworkLinkedList< T >	427
INetworkObjectInitializer	205
NetworkObjectInitializerUnity	476
INetworkObjectProvider	206
NetworkObjectProviderDummy	483
INetworkRunnerCallbacks	208
NetworkDelegates	394
NetworkEvents	409
INetworkRunnerUpdater	214
NetworkRunnerUpdaterDefault	614
INetworkSceneManager	216
INetworkStruct	220
Angle	81
FloatCompressed	144
NetworkBehaviourId	356
NetworkBool	378
NetworkButtons	381
NetworkId	414
NetworkObjectGuid	458
NetworkObjectHeader	468
NetworkObjectNestingKey	478
NetworkObjectTypeId	488
NetworkPhysicsInfo	499
NetworkPrefabId	502
NetworkPrefabRef	510
NetworkRNG	540
NetworkSceneInfo	626

NetworkString< TSize >	645
NetworkTRSPData	678
PlayerRef	683
Ptr	692
QuaternionCompressed	699
SceneRef	742
TickTimer	877
Vector2Compressed	894
Vector3Compressed	899
Vector4Compressed	904
_128	67
_16	68
_2	69
_256	69
_32	70
_4	71
_512	72
_64	73
_8	74
INetworkTRSPTeleport	220
NetworkTransform	672
IUnitySurrogate	222
IUnityValueSurrogate< T >	223
UnityValueSurrogate< T, TReaderWriter >	232
UnitySurrogateBase	230
UnityArraySurrogate< T, ReaderWriter >	224
UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >	226
UnityLinkedListSurrogate< T, ReaderWriter >	228
UnityValueSurrogate< T, TReaderWriter >	232
InterpolatedErrorCorrectionSettings	234
IPlayerJoined	237
IPlayerLeft	237
IRemotePrefabCreated	238
ISceneLoadDone	239
ISceneLoadStart	239
ISimulationEnter	240
ISimulationExit	241
ISpawned	241
HitboxManager	163
NetworkBehaviour	304
IStateAuthorityChanged	242
LagCompensatedHit	242
AABB	246
BoxOverlapQueryParams	250
BVHDraw	252
BVHNodeDrawInfo	253
ColliderDrawInfo	254
HitboxColliderContainerDraw	256
LagCompensatedExt	256
LagCompensationDraw	257
LagCompensationUtils.ContactData	258
PositionRotationQueryParams	259
Query	261
BoxOverlapQuery	248
RaycastQuery	268
RaycastAllQuery	267
SphereOverlapQuery	273

QueryParams	265
RaycastQueryParams	270
SnapshotHistoryDraw	272
SphereOverlapQueryParams	275
LagCompensationSettings	276
LobbyInfo	279
LogSimpleUnity	280
Mask256	280
NestedComponentUtilities	285
NetworkArray< T >.Enumerator	300
NetworkArrayExtensions	302
NetworkArrayReadOnly< T >	303
NetworkAssemblyIgnoreAttribute	304
NetworkAssemblyWeavedAttribute	304
NetworkBehaviour.ArrayReader< T >	327
NetworkBehaviour.BehaviourReader< T >	328
NetworkBehaviour.ChangeDetector	329
NetworkBehaviour.ChangeDetector.Enumerable	332
NetworkBehaviour.ChangeDetector.Enumerator	333
NetworkBehaviour.DictionaryReader< K, V >	334
NetworkBehaviour.LinkListReader< T >	335
NetworkBehaviour.PropertyReader< T >	336
NetworkBehaviourBuffer	337
NetworkBehaviourBufferInterpolator	345
NetworkBehaviourUtils	360
NetworkBehaviourUtils.ArrayInitializer< T >	374
NetworkBehaviourUtils.DictionaryInitializer< K, V >	376
NetworkBehaviourUtils.MetaData	377
NetworkBehaviourWeavedAttribute	377
NetworkConfiguration	391
NetworkDeserializeMethodAttribute	395
NetworkDictionary< K, V >.Enumerator	402
NetworkDictionaryReadOnly< K, V >	404
NetworkedAttribute	407
NetworkedWeavedAttribute	408
NetworkEvents.ConnectFailedEvent	410
NetworkEvents.ConnectRequestEvent	411
NetworkEvents.CustomAuthenticationResponse	411
NetworkEvents.DisconnectFromServerEvent	411
NetworkEvents.HostMigrationEvent	411
NetworkEvents.InputEvent	412
NetworkEvents.InputPlayerEvent	412
NetworkEvents.ObjectEvent	412
NetworkEvents.ObjectPlayerEvent	412
NetworkEvents.PlayerEvent	413
NetworkEvents.ReliableDataEvent	413
NetworkEvents.ReliableProgressEvent	413
NetworkEvents.RunnerEvent	413
NetworkEvents.SessionListUpdateEvent	414
NetworkEvents.ShutdownEvent	414
NetworkEvents.SimulationMessageEvent	414
NetworkId.EqualityComparer	420
NetworkInput	421
NetworkInputUtils	424
NetworkInputWeavedAttribute	426
NetworkLinkedList< T >.Enumerator	433
NetworkLinkedListReadOnly< T >	435
NetworkLoadSceneParameters	438

NetworkObjectFlagsExtensions	456
NetworkObjectGuid.EqualityComparer	467
NetworkObjectHeaderPtr	475
NetworkObjectMeta	477
NetworkObjectNestingKey.EqualityComparer	482
NetworkObjectReleaseContext	484
NetworkObjectSortKeyComparer	486
NetworkObjectSpawnException	487
NetworkObjectTypeId.EqualityComparer	497
NetworkPrefabAcquireContext	500
NetworkPrefabAttribute	502
NetworkPrefabId.EqualityComparer	507
NetworkPrefabInfo	508
NetworkPrefabRef.EqualityComparer	517
NetworkPrefabTable	518
NetworkPrefabTableOptions	526
NetworkProjectConfig	527
NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta	539
NetworkRpcStaticWeavedInvokerAttribute	545
NetworkRpcWeavedInvokerAttribute	546
NetworkRunnerCallbackArgs	612
NetworkRunnerCallbackArgs.ConnectRequest	613
NetworkRunnerUpdaterDefault.NetworkRunnerRender	616
NetworkRunnerUpdaterDefault.NetworkRunnerUpdate	616
NetworkRunnerUpdaterDefaultInvokeSettings	617
NetworkSceneAsyncOp	620
NetworkSceneAsyncOp.Awaiter	625
NetworkSceneLoadId	632
NetworkSceneObjectId	634
NetworkSerializeMethodAttribute	636
NetworkSimulationConfiguration	637
NetworkSpawnOp	641
NetworkSpawnOp.Awaiter	643
NetworkStructUtils	670
NetworkStructWeavedAttribute	671
NormalizedRectAttribute	681
OnChangedRenderAttribute	682
PreserveInPluginAttribute	691
IMessage	691
Versioning	691
Ptr.EqualityComparer	697
ReadWriteUtils	707
ReadWriteUtilsForWeaver	713
ReflectionUtils	719
RenderAttribute	722
RenderTimeline	724
RenderWeavedAttribute	725
ResolveNetworkPrefabSourceAttribute	725
RpcAttribute	726
RpcHeader	728
RpcInfo	732
RpcInvokeData	735
RpcInvokeInfo	736
RpcSendResult	738
RpcTargetAttribute	739
SceneLoadDoneArgs	740
ScriptHelpAttribute	749
SerializableDictionary< TKey, TValue >	750

SerializableType< BaseType >	756
SerializableTypeAttribute	760
SessionInfo	762
Simulation	765
Simulation.AreaOfInterest	783
SimulationBehaviourAttribute	788
SimulationBehaviourListScope	790
SimulationConfig	790
SimulationInput	795
SimulationInput.Buffer	797
SimulationInputHeader	801
SimulationMessage	802
SimulationMessagePtr	814
SimulationRuntimeConfig	814
NetAddress	816
NetBitBufferList	821
NetCommandAccepted	823
NetCommandConnect	823
NetCommandDisconnect	824
NetCommandHeader	824
NetCommandRefused	825
NetConfig	826
StunServers.StunServer	830
StartGameArgs	831
StartGameResult	838
BehaviourStatisticsManager	839
BehaviourStatisticsSnapshot	840
FusionStatisticsManager	841
FusionStatisticsSnapshot	842
LagCompensationStatisticsSnapshot	848
MemoryStatisticsSnapshot	850
NetworkObjectStatisticsManager	852
NetworkObjectStatisticsSnapshot	853
Tick	854
Tick.EqualityComparer	860
Tick.RelationalComparer	861
TickAccumulator	862
TickRate	866
TickRate.Resolved	873
TickRate.Selection	876
TimeSyncConfiguration	881
UnityAddressablesRuntimeKeyAttribute	884
UnityContextMenuItemAttribute	884
UnityDelayedAttribute	885
UnityFormerlySerializedAsAttribute	886
UnityHeaderAttribute	887
UnityMinAttribute	888
UnityMultilineAttribute	889
UnityNonReorderableAttribute	889
UnityNonSerializedAttribute	890
UnityRangeAttribute	890
UnitySerializeField	892
UnitySerializeReference	892
UnitySpaceAttribute	892
UnityTooltipAttribute	893
WeaverGeneratedAttribute	911
IElementReaderWriter< K >	195
IElementReaderWriter< V >	195

INetworkAssetSource< NetworkObject >	202
INetworkPrefabSource	207
NetworkBehaviour.PropertyReader< Fusion.NetworkBehaviourId >	336
SerializableType< Fusion.SimulationBehaviour >	756

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

_128	A FixedStorage that can hold up to 128 words	67
_16	A FixedStorage that can hold up to 16 words	68
_2	A FixedStorage that can hold up to 2 words	69
_256	A FixedStorage that can hold up to 256 words	69
_32	A FixedStorage that can hold up to 32 words	70
_4	A FixedStorage that can hold up to 4 words	71
_512	A FixedStorage that can hold up to 512 words	72
_64	A FixedStorage that can hold up to 64 words	73
_8	A FixedStorage that can hold up to 8 words	74
Allocator	Memory Allocator	75
Allocator.Config	Memory Allocator Configuration	77
Angle	A Networked fusion type for degrees. This can be used with the NetworkedAttribute , in RPCs, or in NetworkInput structs	81
ArrayLengthAttribute	Editor attribute for selecting the minimum and maximum length constraints for an array field	90
AssemblyNameAttribute	Specifies that the attributed field represents the name of an assembly	92
AssetObject	Base class for all Fusion assets	93
TaskManager	Task Factory is used to create new Tasks and Schedule long running Tasks	93
AuthorityMasks	Provides constants and methods for managing authority masks	95

Behaviour	Alternative base class to Unity's MonoBehaviour. This allows for components that work both in Unity, as well as the Photon relays	96
BinaryDataAttribute	Specifies that the field represents binary data	98
BitSetAttribute	Represents an attribute that specifies the number of bits in a bit set	98
CapacityAttribute	Capacity Attribute	99
DecoratingPropertyAttribute	A base class for property attributes that decorate other property attributes	100
DefaultForPropertyAttribute	Default For Property Attribute	101
DisplayAsEnumAttribute	Casts an enum or int value in the inspector to specific enum type for rendering of its popup list. Supplying a method name rather than a type allows a property with the type Type to be used to dynamically get the enum type	103
DisplayNameAttribute	Specifies the display name for a field	104
DolfAttributeBase	Editor attribute for selective editor rendering. Condition member can be a property, field or method (with a return value)	105
DrawerPropertyAttribute	A base class for property attributes that are used to draw properties in the inspector	108
DrawIfAttribute	Editor attribute for selectively drawing/hiding fields. Condition member can be a property, field or method (with a return value)	109
DrawInlineAttribute	Specifies that a field should be drawn inline in the inspector	110
DynamicHeap	A dynamic heap for allocating and tracking unmanaged objects	110
DynamicHeap.Ignore	Ignore this field when scanning for pointers	114
DynamicHeapInstance	Dynamic heap instance	114
EditorButtonAttribute	Specifies that a method should be displayed as a button in the Unity editor	119
DataEncryptor	Responsible for encrypting and decrypting data buffers	123
IDataEncryption	Interface for classes that manage the encryption/decryption of byte arrays	124
ErrorIfAttribute	Editor attribute for adding notices to fields if the condition member evaluates as true. Condition member can be a property, field or method (with a return value)	127
ExpandableEnumAttribute	Editor attribute that shows an enum as an expandable list of options in the inspector	128
FieldEditorButtonAttribute	Editor attribute to add a button that invokes a custom method in the inspector	129
FieldsMask< T >	Base class for FieldsMask<T>	130
FixedSize< T >	A fixed size array that can be used in structs	133
FixedSize< T >.Enumerator	Enumerator for the FixedSize struct	140
FixedBufferPropertyAttribute	Fixed Buffer Property Attribute	142
FixedStorage	Provides utility methods for fixed storage types	143

FloatCompressed	Represents a compressed float value for network transmission	144
FloatUtils	Provides utility methods for compressing and decompressing float values	147
FusionGlobalScriptableObject< T >	A base class for ScriptableObjects that are meant to be globally accessible, at edit-time and run-time. The way such objects are loaded is driven by usages of FusionGlobalScriptableObjectSourceAttribute attributes	149
FusionGlobalScriptableObjectAttribute	Provides additional information for a global scriptable object	152
FusionGlobalScriptableObjectLoadResult	The result of FusionGlobalScriptableObjectSourceAttribute.Load . Contains the loaded object and an optional unloader delegate	153
FusionGlobalScriptableObjectSourceAttribute	Base class for all attributes that can be used to load FusionGlobalScriptableObject . Attributes need to be registered at the assembly level. For instance, this snippet is used to register a default loader, that attempts to load from Resources based on FusionGlobalScriptableObjectAttribute.DefaultPath :	155
FusionMonoBehaviour	Base class for all Fusion MonoBehaviours	157
FusionScriptableObject	Base class for all Fusion scriptable objects	157
HeapConfiguration	Memory Heap Settings	157
HideArrayElementLabelAttribute	Attribute used to hide the label of an array element in the inspector	159
Hitbox	Represents a single lag-compensated collider. Multiple component instances can be added anywhere in the hierarchy of a NetworkObject which includes a HitboxRoot	159
HitboxManager	Entry point for lag compensated Hitbox queries, which maintains a history buffer, and provides lag compensated raycast and overlap methods. Singleton instance is accessible through the property Runner.LagCompensation	163
HitboxRoot	Root Hitbox group container. Manages registering/unregistering hitboxes with the group, and defines the broadphase geometry for the group	177
HostMigrationConfig	Project configuration settings specific to how the Host Migration behaves	183
HostMigrationToken	Transitory Holder with all necessary information to restart the Fusion Runner after the Host Migration has completed	184
IAfterAllTicks	Interface for AfterAllTicks callback. Called after the re-simulation loop (when applicable), and also after the forward simulation loop. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	184
IAfterClientPredictionReset	Callback interface for AfterClientPredictionReset . Called at the very start of the resimulation loop (on clients with prediction enabled), immediately after state is set to the latest server snapshot. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	185
IAfterHostMigration	Used to mark NetworkBehaviors that need to be react after a Host Migration process	185
IAfterRender	Interface for AfterRender callback. Called after the render loop	186
IAfterSpawned	Interface for AfterSpawned callback. Called after the object is spawned	186
IAfterTick	Interface for AfterTick callback. Called after each tick simulation completes. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	187

IAfterUpdate	Interface for the AfterUpdate callback, which is called at the end of each Fusion Update segment.	
	Implement this interface on SimulationBehaviour and NetworkBehaviour classes	188
IAsyncOperation	Defines an asynchronous operation	188
IBeforeAllTicks	Interface for BeforeAllTicks callback. Called before the re-simulation loop (when applicable), and also before the forward simulation loop. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	189
IBeforeClientPredictionReset	Callback interface for BeforeClientPredictionReset . Called at the very start of the re-simulation loop (on clients with prediction enabled), before state is set to the latest server snapshot. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	190
IBeforeCopyPreviousState	Interface for BeforeCopyPreviousState callback. Called before the copy of the previous state	191
IBeforeHitboxRegistration	Interface for BeforeHitboxRegistration callback. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	191
IBeforeSimulation	Interface for BeforeSimulation callback. Called before both the re-simulation (when applicable) and forward simulation loops. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	192
IBeforeTick	Interface for BeforeTick callback. Called before each tick is simulated. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	193
IBeforeUpdate	Interface for the BeforeUpdate callback, which is called at the beginning of each Fusion Update segment. Implement this interface on SimulationBehaviour and NetworkBehaviour classes	193
ICoroutine	Defines a coroutine	194
IDespawned	Interface for Despawned callback. Called when a NetworkBehaviour is despawned	194
IElementReaderWriter< T >	Defines the interface for reading and writing elements in a byte array	195
IFixedStorage	Interface for fixed storage types	197
IInputAuthorityGained	Interface for handling the event when the input authority is gained	197
IInputAuthorityLost	Interface for handling the event when the input authority is lost	198
IInterestEnter	Interface for handling the event when a player enters the area of interest	198
IInterestExit	Interface for handling the event when a player exits the area of interest	199
ILocalPrefabCreated	Interface for handling the event when a local prefab is created	200
INetworkArray	Defines the interface for a networked array	200
INetworkAssetSource< T >	Interface for a network asset source	202
INetworkDictionary	Defines the interface for a networked dictionary	203
INetworkInput	Flag interface for custom NetworkInput structs	204
INetworkLinkedList	Defines the interface for a networked linked list	204
INetworkObjectInitializer	Interface for initializing network objects	205

INetworkObjectProvider	Interface which defines the handlers for <code>NetworkRunner</code> <code>Spawn()</code> and <code>Despawn()</code> actions. Passing an instance of this interface to <code>NetworkRunner.StartGame(StartGameArgs)</code> as the <code>StartGameArgs.ObjectProvider</code> argument value will assign that instance as the handler for runner <code>Spawn()</code> and <code>Despawn()</code> actions. By default (if <code>StartGameArgs.ObjectProvider == null</code>) actions will use <code>Instantiate()</code> , and <code>Despawn()</code> actions will use <code>Destroy()</code>	206
INetworkPrefabSource	Interface for a network prefab source	207
INetworkRunnerCallbacks	Interface for <code>NetworkRunner</code> callbacks. Register a class/struct instance which implements this interface with <code>NetworkRunner.AddCallbacks(INetworkRunnerCallbacks[])</code>	208
INetworkRunnerUpdater	Interface which defines the handlers for <code>NetworkRunner</code> Updates. An implementation is responsible for calling <code>NetworkRunner.UpdateInternal(double)</code> and <code>NetworkRunner.RenderInternal</code> periodically	214
INetworkSceneManager	Interface for a <code>NetworkRunner</code> scene manager. A scene manager is responsible for loading and unloading scenes	216
INetworkStruct	Base interface for all <code>Fusion</code> Network Structs	220
INetworkTRSPTeleport	Implement this interface on a <code>NetworkTRSP</code> implementation to indicate it can be teleported	220
InlineHelpAttribute	If applied to a field, checks if there is a help text for the field or the field's type and shows it in the inspector	221
IUnitySurrogate	Represents an interface for Unity surrogates. This interface provides methods for reading and writing data	222
IUnityValueSurrogate< T >	Represents an interface for Unity value surrogates. This interface provides a property for accessing and modifying the data	223
UnityArraySurrogate< T, ReaderWriter >	A base class for Unity array surrogates	224
UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >	A surrogate for serializing a dictionary	226
UnityLinkedListSurrogate< T, ReaderWriter >	A surrogate for serializing a linked list	228
UnitySurrogateBase	Represents a base class for Unity surrogates. This class is serializable and provides abstract methods for reading, writing, and initializing data	230
UnityValueSurrogate< T, TReaderWriter >	Represents a base class for Unity value surrogates. This class is serializable and provides methods for reading, writing, and initializing data	232
InterpolatedErrorCorrectionSettings	A set of parameters that tune the interpolated correction of prediction error on transform data	234
IPlayerJoined	Interface for handling the event when a player joins the game	237
IPlayerLeft	Interface for handling the event when a player leaves the game	237
IRemotePrefabCreated	Interface for handling the event when a remote prefab is created	238
ISceneLoadDone	Interface for handling the event when a scene load operation is completed	239
ISceneLoadStart	Interface for handling the event when a scene load operation is started	239
ISimulationEnter	Interface for <code>SimulationEnter</code> callback. Called when the <code>NetworkObject</code> joins <code>AreaOfInterest</code> . Implement this interface on <code>SimulationBehaviour</code> and <code>NetworkBehaviour</code> classes	240

ISimulationExit	Interface for the SimulationExit callback. Called when the NetworkObject leaves AreaOfInterest.	
	Implement this interface on SimulationBehaviour and NetworkBehaviour classes	241
ISpawned	Interface for handling the event when an object is spawned	241
IStateAuthorityChanged	Interface for handling the event when the state authority changes	242
LagCompensatedHit	Defines a lag compensated query hit result	242
AABB	Represents an Axis-Aligned Bounding Box (AABB)	246
BoxOverlapQuery	Class that represents a box overlap query. Used to query against the NetworkRunner.LagCompensation API	248
BoxOverlapQueryParams	Base parameters needed to execute a box overlap query	250
BVHDraw	Provide a way to iterate over BVH and return a BVHNodeDrawInfo for each node	252
BVHNodeDrawInfo	Container class to provide the necessary info to draw nodes from the BVH	253
ColliderDrawInfo	Container class to provide the necessary information to draw a hitbox collider	254
HitboxColliderContainerDraw	Provide a way to iterate over the HitboxBuffer.HitboxSnapshot and return the ColliderDrawInfo for each collider on the snapshot	256
LagCompensatedExt	LagCompensated Extension methods	256
LagCompensationDraw	Provide access to iterate over the lag compensation system components and give the necessary information to draw them	257
LagCompensationUtils.ContactData	Details regarding a shape intersection. It does not carry information about the intersection happening or not	258
PositionRotationQueryParams	Query parameters for position rotation query	259
Query	Base class for all Lag Compensation queries	261
QueryParams	Base parameters needed to execute a query	265
RaycastAllQuery	Class that represents a raycast all query. Used to query against the NetworkRunner.LagCompensation API	267
RaycastQuery	Class that represents a raycast query. Used to query against the NetworkRunner.LagCompensation API	268
RaycastQueryParams	Base parameters needed to execute a raycast query	270
SnapshotHistoryDraw	Provide a way to iterate over the HitboxBuffer and return the HitboxColliderContainerDraw container for each snapshot on the buffer	272
SphereOverlapQuery	Class that represents a sphere overlap query. Used to query against the NetworkRunner.LagCompensation API	273
SphereOverlapQueryParams	Base parameters needed to execute a sphere overlap query	275
LagCompensationSettings	Settings for lag compensation history	276

LayerAttribute	Specifies that an int field should be drawn as a layer field in the inspector	278
LayerMatrixAttribute	Specifies that the integer array field should be drawn as a layer matrix in the inspector	278
LobbyInfo	Holds information about a Lobby	279
LogSimpleUnity	Log Simple Unity	280
Mask256	Mask256 is a 256-bit mask that can be used to store 256 boolean values	280
MaxStringByteCountAttribute	Specifies that the string field should be drawn as a text field with a maximum byte count for given encoding	284
NestedComponentUtilities	Tools to replace GetComponent variants that respects nested objects. These are used to find components of a NetworkedObjects without also finding components that belong to parent or child NetworkedObjects	285
NetworkArray< T >	Fusion type for networking arrays. Maximum capacity is fixed, and is set with the CapacityAttribute.	
	293	
NetworkArray< T >.Enumerator	Enumerator for NetworkArray	300
NetworkArrayExtensions	Provides extension methods for the NetworkArray class	302
NetworkArrayReadOnly< T >	Provides a read-only view of a network array	303
NetworkAssemblyIgnoreAttribute	Network Assembly Ignore Attribute	304
NetworkAssemblyWeavedAttribute	Network Assembly Weaved Attribute	304
NetworkBehaviour	Base class for Fusion network components, which are associated with a NetworkObject	304
NetworkBehaviour.ArrayReader< T >	Provides a reader for network arrays of type T	327
NetworkBehaviour.BehaviourReader< T >	Provides a reader for network behaviours of type T	328
NetworkBehaviour.ChangeDetector	Change detector for a NetworkBehaviour	329
NetworkBehaviour.ChangeDetector.Enumerable	Struct representing a collection of changes detected in a NetworkBehaviour	332
NetworkBehaviour.ChangeDetector.Enumerator	Enumerator for the collection of changes detected in a NetworkBehaviour	333
NetworkBehaviour.DictionaryReader< K, V >	Provides a reader for network dictionaries with keys of type K and values of type V	334
NetworkBehaviour.LinkListReader< T >	Provides a reader for network linked lists of type T	335
NetworkBehaviour.PropertyReader< T >	Provides a reader for properties of type T in a network behaviour	336
NetworkBehaviourBuffer	Provides low level accesss to data buffers that can be read using a NetworkBehaviour.Reader	337
NetworkBehaviourBufferInterpolator	The NetworkBehaviourBufferInterpolator struct is used to interpolate between two NetworkBehaviourBuffer instances. This is a read-only, ref struct, meaning it cannot be boxed and it can only be used on the stack	345
NetworkBehaviourId	Represents the unique identifier for a NetworkBehaviour instance	356

NetworkBehaviourUtils	
This static class provides utility methods for working with NetworkBehaviour objects	360
NetworkBehaviourUtils.ArrayInitializer< T >	
A utility structure for initializing NetworkArray and NetworkLinkedList with inline initialization	374
NetworkBehaviourUtils.DictionaryInitializer< K, V >	
A utility structure for initializing NetworkDictionary with inline initialization	376
NetworkBehaviourUtils.MetaData	
This structure holds metadata for a NetworkBehaviour object	377
NetworkBehaviourWeavedAttribute	
Network Behaviour Weaved Attribute	377
NetworkBool	
Represents a boolean value that can be networked	378
NetworkButtons	
Represents a set of buttons that can be networked	381
NetworkConfiguration	
Main network configuration class	391
NetworkDelegates	
Network Runner Callbacks Delegates	394
NetworkDeserializeMethodAttribute	
Network Deserialize Method Attribute	395
NetworkDictionary< K, V >	
Fusion type for networking Dictionaries. Maximum capacity is fixed, and is set with the CapacityAttribute.	
395	
NetworkDictionary< K, V >.Enumerator	
Enumerator for NetworkDictionary	402
NetworkDictionaryReadOnly< K, V >	
A read-only version of NetworkDictionary< TKey, TValue >	404
NetworkedAttribute	
407	
NetworkedWeavedAttribute	
Networked Weaved Attribute	408
NetworkEvents	
Companion component for NetworkRunner . Exposes INetworkRunnerCallbacks as UnityEvents, which can be wired up to other components in the inspector	409
NetworkEvents.ConnectFailedEvent	
UnityEvent for ConnectFailed	410
NetworkEvents.ConnectRequestEvent	
UnityEvent for ConnectRequest	411
NetworkEvents.CustomAuthenticationResponse	
UnityEvent for Custom Authentication	411
NetworkEvents.DisconnectFromServerEvent	
UnityEvent for DisconnectFromServer	411
NetworkEvents.HostMigrationEvent	
UnityEvent for HostMigration	411
NetworkEvents.InputEvent	
UnityEvent for NetworkInput	412
NetworkEvents.InputPlayerEvent	
UnityEvent for NetworkInput with PlayerRef	412
NetworkEvents.ObjectEvent	
UnityEvent for NetworkObject	412
NetworkEvents.ObjectPlayerEvent	
UnityEvent for NetworkObject with PlayerRef	412
NetworkEvents.PlayerEvent	
UnityEvent for PlayerRef	413
NetworkEvents.ReliableDataEvent	
UnityEvent for Reliable Data	413

NetworkEvents.ReliableProgressEvent	UnityEvent for Reliable Data Progress	413
NetworkEvents.RunnerEvent	UnityEvent for NetworkRunner	413
NetworkEvents.SessionListUpdateEvent	UnityEvent for SessionInfo List	414
NetworkEvents.ShutdownEvent	UnityEvent for Shutdown	414
NetworkEvents.SimulationMessageEvent	UnityEvent for SimulationMessage	414
NetworkId	The unique identifier for a network entity	414
NetworkId.EqualityComparer	IEqualityComparer interface for NetworkId objects	420
NetworkInput	NetworkInput Struct	421
NetworkInputUtils	Utility methods for NetworkInput	424
NetworkInputWeavedAttribute	Network Input Weaved Attribute	426
NetworkLinkedList< T >	Fusion type for networking LinkedLists. Maximum capacity is fixed, and is set with the CapacityAttribute. Typical Usage:	427
NetworkLinkedList< T >.Enumerator	Enumerator for NetworkLinkedList< T >	433
NetworkLinkedListReadOnly< T >	Read-only version of NetworkLinkedList< T >	435
NetworkLoadSceneParameters	Parameters for loading a scene	438
NetworkMecanimAnimator	A component for synchronizing the Animator controller state from the State Authority to network proxies. Requires a Unity Animator component, and a NetworkObject component. NOTE: Animator Root Motion is not compatible with re-simulation and prediction	443
NetworkObject	The primary Fusion component for networked GameObject entities. This stores the object's network identity and manages the object's state and input authority	445
NetworkObjectFlagsExtensions	Extension methods for the NetworkObjectFlags enum	456
NetworkObjectGuid	NetworkObjectGuid	458
NetworkObjectGuid.EqualityComparer	EqualityComparer for NetworkObjectGuid	467
NetworkObjectHeader	Network object header information for a NetworkObject	468
NetworkObjectHeaderPtr	Represents a pointer to a NetworkObjectHeader . This struct is unsafe because it uses pointers	475
NetworkObjectInitializerUnity	Initializes network objects for Unity	476
NetworkObjectMeta	Meta information about a network object	477
NetworkObjectNestingKey	A key used to identify a network object nesting	478
NetworkObjectNestingKey.EqualityComparer	Implements the IEqualityComparer interface	482
NetworkObjectPrefabData	This class represents the data for a network object prefab	483

NetworkObjectProviderDummy	A dummy implementation of the INetworkObjectProvider interface. This class is used for testing purposes and throws a NotImplementedException for all its methods	483
NetworkObjectReleaseContext	Represents the context for releasing a network object. This struct is unsafe because it uses pointers	484
NetworkObjectSortKeyComparer	This class is used to compare two NetworkObject instances based on their SortKey. It implements the IComparer interface	486
NetworkObjectSpawnException	Network Object Spawn Exception	487
NetworkObjectTypeId	ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable	488
NetworkObjectTypeId.EqualityComparer	NetworkObjectTypeId Comparer	497
NetworkPhysicsInfo	Network Physics INetworkStruct	499
NetworkPrefabAcquireContext	Represents the context for acquiring a prefab instance for a network object. This struct is unsafe because it uses pointers	500
NetworkPrefabAttribute	Network Prefab Attribute	502
NetworkPrefabId	ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable	502
NetworkPrefabId.EqualityComparer	Equality comparer for NetworkPrefabId	507
NetworkPrefabInfo	Meta data for a NetworkObject prefab which has been cataloged in a NetworkProjectConfig.PrefabTable	508
NetworkPrefabRef	NetworkPrefabRef	510
NetworkPrefabRef.EqualityComparer	EqualityComparer for NetworkPrefabRef	517
NetworkPrefabTable	Class representing a table of network prefabs	518
NetworkPrefabTableOptions	Options for the NetworkPrefabTable	526
NetworkProjectConfig	The core Fusion config file that is shared with all peers at startup	527
NetworkProjectConfigAsset	Manages and references the current instance of NetworkProjectConfig	536
NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta	An auto-generated list containing meta information about all the SimulationBehaviours in the project, e.g. execution order	539
NetworkRNG	PCG32 random generator, 16 bytes in size. http://www.pcg-random.org	540
NetworkRpcStaticWeavedInvokerAttribute	Network Rpc Static Weaved Invoker Attribute Contains info about a static weaved RPC Method	545
NetworkRpcWeavedInvokerAttribute	Network Rpc Weaved Invoker Attribute Contains info about a weaved RPC Method	546
NetworkRunner	Host Migration related code in order to get a copy of the Simulation State	548
NetworkRunnerCallbackArgs	Stores data types used on the INetworkRunnerCallbacks interface	612
NetworkRunnerCallbackArgs.ConnectRequest	Data holder of a Connection Request from a remote client	613
NetworkRunnerUpdaterDefault	Default implementation of INetworkRunnerUpdater that uses the Unity PlayerLoop	614

NetworkRunnerUpdaterDefault.NetworkRunnerRender	Used to invoke NetworkRunner.RenderInternal in the PlayerLoop	616
NetworkRunnerUpdaterDefault.NetworkRunnerUpdate	Used to invoke NetworkRunner.UpdateInternal(double) in the PlayerLoop	616
NetworkRunnerUpdaterDefault.InvokeSettings	Settings for the NetworkRunnerUpdaterDefault	617
NetworkSceneAsyncOp	A wrapper for async scene operations	620
NetworkSceneAsyncOp.Awaiter	Awaiter for NetworkSceneAsyncOp	625
NetworkSceneInfo	Can store up to 8 active scenes and allows for duplicates. Each write increases Version which can be used to generate unique scene objects ids for when a scene is supposed to be reloaded	626
NetworkSceneLoadId	A unique identifier for a scene load operation	632
NetworkSceneObjectId	A unique identifier for a scene object	634
NetworkSerializeMethodAttribute	Network Serialize Method Attribute	636
NetworkSimulationConfiguration	Configuration for network conditions simulation (induced latency and loss)	637
NetworkSpawnOp	Spawn Operation	641
NetworkSpawnOp.Awaiter	Awaiter for NetworkSpawnOp	643
NetworkString< TSize >	Fixed-size UTF32 string. All operations are alloc-free, except for converting to System.String	645
NetworkStructUtils	Utility methods for INetworkStruct	670
NetworkStructWeavedAttribute	Describes the total number of WORDs a INetworkStruct uses	671
NetworkTransform	Add to any NetworkObject Transform, or its associated child Transforms to automatically synchronize TRSP (Position/Rotation/Scale/Parent)	672
NetworkTRSP	Base class for spatial (Position/Rotation/Scale/Parent) synchronization component, such as NetworkTransform . Provides the base logic for render interpolation, parenting synchronization, and teleport, that can be used in components derived from this class	675
NetworkTRSPData	Data structure storing spatial (Position/Rotation/Scale/Parent) synchronization data for spatial synchronization components, NetworkTRSP and its subclass NetworkTransform	678
NormalizedRectAttribute	Enables a special inspector drawer for Unity Rect type, specially designed for editing Rect→ Transforms using normalized values	681
OnChangedRenderAttribute	OnChangedRender Attribute	682
PlayerRef	Represents a Fusion player	683
PreserveInPluginAttribute	Preserve In Plugin Attribute	691
IMessage	Represents a Protocol Message	691
Versioning	Versioning Information	691
Ptr	Ptr	692
Ptr.EqualityComparer	Ptr Equality Comparer	697

QuaternionCompressed	Represents a compressed Quaternion value for network transmission	699
RangeExAttribute	Represents an attribute that specifies a range of values for a field or property	704
ReadOnlyAttribute	Attribute used to mark a field as read-only	706
ReadWriteUtils	Provides utility methods for reading and writing data	707
ReadWriteUtilsForWeaver	Provides utility methods for reading and writing data	713
ReflectionUtils	Provides utility methods for reflection	719
RenderAttribute	Override default render settings for [Networked] properties	722
RenderTimeline	Can be used to acquire interpolated data for different points in time	724
RenderWeavedAttribute	Render Weaved Attribute	725
ResolveNetworkPrefabSourceAttribute	Resolve Network Prefab Source Attribute	725
RpcAttribute	Flags a method as being a networked Remote Procedure Call. Only usable in a NetworkBehaviour . Calls to this method (from the indicated allowed RpcSources) will generate a network message, which will execute the method remotely on the indicated RpcTargets . The RPC method can include an empty RpcInfo argument, that will include meta information about the RPC on the receiving peer	726
RpcHeader	Header for RPC messages	728
RpcInfo	RpcInfo is a struct that contains information about the RPC message	732
RpclinevokeData	Represents the data required to invoke an RPC message	735
RpclinevokeInfo	May be used as an optional RpcAttribute return value. Contains meta data about the RPC send, such as failure to send reasons, culling, message size, etc	736
RpcSendResult	RPC send operation result information	738
RpcTargetAttribute	RPC attribute used to indicate a specific target player for an RPC when sending from one player to another. RPC is sent to the server, and then is forwarded to the specified player. Usage:	739
SceneLoadDoneArgs	Struct that contains information about a scene after it has been loaded	740
ScenePathAttribute	Specifies that a string field represents a scene path	741
SceneRef	Scene reference struct. Can be used to reference a scene by index or by path	742
ScriptHelpAttribute	Defines the appearance of the script header in the Unity inspector	749
SerializableDictionary< TKey, TValue >	A serializable dictionary	750
SerializableType< BaseType >	A System.Type wrapper that can be serialized	756
SerializableTypeAttribute	Specifies that either a string field represents a type name or sets additional options for SerializableType field	760
SerializeReferenceTypePickerAttribute	Attribute used to show a type picker for a field with [SerializeField]	761

SessionInfo	Holds information about the Game Session	762
Simulation	Main simulation class	765
Simulation.AreaOfInterest	Area of Interest Definition	783
SimulationBehaviour	Base class for a Fusion aware Behaviour (derived from <code>UnityEngine.MonoBehaviour</code>). If a SimulationBehaviour is found on a NetworkRunner game object during the runner initialisation, the SimulationBehaviour is automatically registered. Objects derived from this object can be associated with a NetworkRunner and Simulation using NetworkRunner.AddGlobal()	786
SimulationBehaviourAttribute	Attribute for specifying which SimulationStages and SimulationModes this SimulationBehaviour will execute in. Can be used to limit execution to only Host, Server or Client peers, or to only execute on Resimulation or Forward ticks. Usage:	788
SimulationBehaviourListScope	Provides a scope for a SimulationBehaviourUpdater.BehaviourList , incrementing its lock count on creation and decrementing it on disposal. If the lock count reaches zero on disposal, all pending removals in the list are performed	790
SimulationConfig	Project configuration settings specific to how the Simulation class behaves	790
SimulationInput	Simulation Input	795
SimulationInput.Buffer	Buffer for SimulationInputs	797
SimulationInputHeader	Simulation Input Header	801
SimulationMessage	Simulation Message	802
SimulationMessagePtr	Simulation Message Pointer	814
SimulationRuntimeConfig	Stores the runtime configuration of the simulation	814
NetAddress	Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address	816
NetBitBufferList	Represents a linked list of <code>Fusion.Sockets.NetBitBuffer</code>	821
NetCommandAccepted	Accepted Command, sent by the server when a remote client connection is accepted	823
NetCommandConnect	Connect Command used to signal a remote server that a client is trying to connect to it	823
NetCommandDisconnect	Disconnect Command, it can be used by either side of the connection	824
NetCommandHeader	Network Command Header Describe its type and usual settings for all commands	824
NetCommandRefused	Refuse Command, sent by the server when the connection was refused. This happens when the server has reached its max connection capacity	825
NetConfig	General configuration used to drive the behavior of the Socket library	826
StunServers.StunServer	Stores Addresses of a STUN Server	830
StartGameArgs	Fusion Start Arguments, used to configure the simulation mode and other settings	831
StartGameResult	Represents the result of starting the Fusion Simulation	838

BehaviourStatisticsManager	Behaviour statistics manager will provide access to the behaviour statistics snapshots	839
BehaviourStatisticsSnapshot	Represents a snapshot of statistics related to Fusion Behaviour type execution	840
FusionStatisticsManager	Represents a fusion statistics manager	841
FusionStatisticsSnapshot	Represents a snapshot of Fusion statistics	842
LagCompensationStatisticsSnapshot	Represents a snapshot of lag compensation statistics	848
MemoryStatisticsSnapshot	Represents a snapshot of specific memory statistics. For basic total allocated memory and total free memory check the Fusion Statistics	850
NetworkObjectStatisticsManager	Manages network object statistics for monitored network objects	852
NetworkObjectStatisticsSnapshot	Represents a snapshot of network object statistics	853
Tick	A tick is a 32-bit integer that represents a frame number	854
Tick.EqualityComparer	Provides a mechanism for comparing two Tick objects for equality	860
Tick.RelationalComparer	Provides a mechanism for comparing two Tick objects	861
TickAccumulator	A tick accumulator	862
TickRate	A tick rate is a collection of tick rates	866
TickRate.Resolved	Represents a resolved tick rate	873
TickRate.Selection	Represents a selection of tick rates for client and server	876
TickTimer	A timer that is based on ticks instead of seconds	877
TimeSyncConfiguration	Time Synchronization Configuration	881
ToggleLeftAttribute	Specifies that the bool field should be drawn as the toggle on the left side of the label	882
UnitAttribute	Unit Attribute class. Used to mark a field with the respective Units	883
UnityAddressablesRuntimeKeyAttribute	Specifies that the string field represents a key for Unity Addressables	884
UnityAssetGuidAttribute	Specifies that the string field represents a GUID of an asset	884
UnityContextMenuMenuItemAttribute	Unity ContextMenuItemAttribute	884
UnityDelayedAttribute	Unity DelayedAttribute	885
UnityFormerlySerializedAsAttribute	Unity FormerlySerializedAsAttribute	886
UnityHeaderAttribute	Unity HeaderAttribute	887
UnityMinAttribute	Unity MinAttribute	888
UnityMultilineAttribute	Unity MultilineAttribute	889
UnityNonReorderableAttribute	Unity NonReorderableAttribute	889

UnityNonSerializedAttribute	
Unity NonSerializedAttribute	890
UnityRangeAttribute	
Unity RangeAttribute	890
UnityResourcePathAttribute	
Specifies that the string field represents a path to a Unity resource	891
UnitySerializeField	
Unity SerializeField	892
UnitySerializeReference	
Unity SerializeReference	892
UnitySpaceAttribute	
Unity SpaceAttribute	892
UnityTooltipAttribute	
Unity TooltipAttribute	893
Vector2Compressed	
Represents a compressed Vector2 value for network transmission	894
Vector3Compressed	
Represents a compressed Vector3 value for network transmission	899
Vector4Compressed	
Represents a compressed Vector4 value for network transmission	904
WarnIfAttribute	
Editor attribute for adding notices to fields if the condition member evaluates as <code>true</code> . Condition member can be a property, field or method (with a return value)	909
WeaverGeneratedAttribute	
Weaver Generated Attribute	911

Chapter 5

Namespace Documentation

5.1 Fusion Namespace Reference

Classes

- struct [_128](#)
A *FixedStorage* that can hold up to 128 words.
- struct [_16](#)
A *FixedStorage* that can hold up to 16 words.
- struct [_2](#)
A *FixedStorage* that can hold up to 2 words.
- struct [_256](#)
A *FixedStorage* that can hold up to 256 words.
- struct [_32](#)
A *FixedStorage* that can hold up to 32 words.
- struct [_4](#)
A *FixedStorage* that can hold up to 4 words.
- struct [_512](#)
A *FixedStorage* that can hold up to 512 words.
- struct [_64](#)
A *FixedStorage* that can hold up to 64 words.
- struct [_8](#)
A *FixedStorage* that can hold up to 8 words.
- struct [Allocator](#)
Memory Allocator
- struct [Angle](#)
A Networked fusion type for degrees. This can be used with the [NetworkedAttribute](#), in RPCs, or in [NetworkInput](#) structs.
- class [ArrayLengthAttribute](#)
Editor attribute for selecting the minimum and maximum length constraints for an array field.
- class [AssemblyNameAttribute](#)
Specifies that the attributed field represents the name of an assembly.
- class [AssetObject](#)
Base class for all Fusion assets.
- class [AuthorityMasks](#)
Provides constants and methods for managing authority masks.

- class [Behaviour](#)

Alternative base class to Unity's MonoBehaviour. This allows for components that work both in Unity, as well as the Photon relays.
- class [BinaryDataAttribute](#)

Specifies that the field represents binary data.
- class [BitSetAttribute](#)

Represents an attribute that specifies the number of bits in a bit set.
- class [CapacityAttribute](#)

Capacity Attribute
- class [DecoratingPropertyAttribute](#)

A base class for property attributes that decorate other property attributes.
- class [DefaultForPropertyAttribute](#)

Default For Property Attribute
- class [DisplayAsEnumAttribute](#)

Casts an enum or int value in the inspector to specific enum type for rendering of its popup list. Supplying a method name rather than a type allows a property with the type Type to be used to dynamically get the enum type.
- class [DisplayNameAttribute](#)

Specifies the display name for a field.
- class [DolfAttributeBase](#)

Editor attribute for selective editor rendering. Condition member can be a property, field or method (with a return value).
- class [DrawerPropertyAttribute](#)

A base class for property attributes that are used to draw properties in the inspector.
- class [DrawIfAttribute](#)

Editor attribute for selectively drawing/hiding fields. Condition member can be a property, field or method (with a return value).
- class [DrawInlineAttribute](#)

Specifies that a field should be drawn inline in the inspector.
- struct [DynamicHeap](#)

A dynamic heap for allocating and tracking unmanaged objects.
- class [DynamicHeapInstance](#)

Dynamic heap instance.
- class [EditorButtonAttribute](#)

Specifies that a method should be displayed as a button in the Unity editor.
- class [ErrorIfAttribute](#)

Editor attribute for adding notices to fields if the condition member evaluates as true. Condition member can be a property, field or method (with a return value).
- class [ExpandableEnumAttribute](#)

Editor attribute that shows an enum as an expandable list of options in the inspector.
- class [FieldEditorButtonAttribute](#)

Editor attribute to add a button that invokes a custom method in the inspector.
- class [FieldsMask](#)

Base class for FieldsMask< T >.
- class [FixedArray](#)

A fixed size array that can be used in structs.
- class [FixedBufferPropertyAttribute](#)

Fixed Buffer Property Attribute
- class [FixedStorage](#)

Provides utility methods for fixed storage types.
- struct [FloatCompressed](#)

Represents a compressed float value for network transmission.
- class [FloatUtils](#)

- [Provides utility methods for compressing and decompressing float values.](#)
- class [FusionGlobalScriptableObject](#)

A base class for ScriptableObjects that are meant to be globally accessible, at edit-time and runtime. The way such objects are loaded is driven by usages of [FusionGlobalScriptableObjectSourceAttribute](#) attributes.
- class [FusionGlobalScriptableObjectAttribute](#)

Provides additional information for a global scriptable object.
- struct [FusionGlobalScriptableObjectLoadResult](#)

The result of [FusionGlobalScriptableObjectSourceAttribute.Load](#). Contains the loaded object and an optional unloader delegate.
- class [FusionGlobalScriptableObjectSourceAttribute](#)

Base class for all attributes that can be used to load [FusionGlobalScriptableObject](#). Attributes need to be registered at the assembly level. For instance, this snippet is used to register a default loader, that attempts to load from Resources based on [FusionGlobalScriptableObjectAttribute.DefaultPath](#):
- class [FusionMonoBehaviour](#)

Base class for all [Fusion](#) MonoBehaviours.
- class [FusionScriptableObject](#)

Base class for all [Fusion](#) scriptable objects.
- class [HeapConfiguration](#)

Memory Heap Settings
- class [HideArrayElementLabelAttribute](#)

Attribute used to hide the label of an array element in the inspector.
- class [Hitbox](#)

Represents a single lag-compensated collider. Multiple component instances can be added anywhere in the hierarchy of a [NetworkObject](#) which includes a [HitboxRoot](#).
- class [HitboxManager](#)

Entry point for lag compensated [Hitbox](#) queries, which maintains a history buffer, and provides lag compensated raycast and overlap methods. Singleton instance is accessible through the property [Runner.LagCompensation](#).
- class [HitboxRoot](#)

Root [Hitbox](#) group container. Manages registering/unregistering hitboxes with the group, and defines the broadphase geometry for the group.
- class [HostMigrationConfig](#)

Project configuration settings specific to how the Host Migration behaves.
- class [HostMigrationToken](#)

Transitory Holder with all necessary information to restart the [Fusion](#) Runner after the Host Migration has completed
- interface [IAfterAllTicks](#)

Interface for [AfterAllTicks](#) callback. Called after the re-simulation loop (when applicable), and also after the forward simulation loop. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IAfterClientPredictionReset](#)

Callback interface for [AfterClientPredictionReset](#). Called at the very start of the resimulation loop (on clients with prediction enabled), immediately after state is set to the latest server snapshot. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IAfterHostMigration](#)

Used to mark NetworkBehaviors that need to be react after a Host Migration process
- interface [IAfterRender](#)

Interface for [AfterRender](#) callback. Called after the render loop.
- interface [IAfterSpawned](#)

Interface for [AfterSpawned](#) callback. Called after the object is spawned.
- interface [IAfterTick](#)

Interface for [AfterTick](#) callback. Called after each tick simulation completes. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IAfterUpdate](#)

Interface for the [AfterUpdate](#) callback, which is called at the end of each [Fusion](#) Update segment. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

- interface [IAsyncOperation](#)
Defines an asynchronous operation.
- interface [IBeforeAllTicks](#)
Interface for [BeforeAllTicks](#) callback. Called before the re-simulation loop (when applicable), and also before the forward simulation loop. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IBeforeClientPredictionReset](#)
Callback interface for [BeforeClientPredictionReset](#). Called at the very start of the re-simulation loop (on clients with prediction enabled), before state is set to the latest server snapshot. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IBeforeCopyPreviousState](#)
Interface for [BeforeCopyPreviousState](#) callback. Called before the copy of the previous state.
- interface [IBeforeHitboxRegistration](#)
Interface for [BeforeHitboxRegistration](#) callback. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IBeforeSimulation](#)
Interface for [BeforeSimulation](#) callback. Called before both the re-simulation (when applicable) and forward simulation loops. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IBeforeTick](#)
Interface for [BeforeTick](#) callback. Called before each tick is simulated. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [IBeforeUpdate](#)
Interface for the [BeforeUpdate](#) callback, which is called at the beginning of each [Fusion](#) Update segment. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.
- interface [ICoroutine](#)
Defines a coroutine.
- interface [IDespawned](#)
Interface for [Despawned](#) callback. Called when a [NetworkBehaviour](#) is despawned.
- interface [IElementReaderWriter](#)
Defines the interface for reading and writing elements in a byte array.
- interface [IFixedStorage](#)
Interface for fixed storage types.
- interface [IInputAuthorityGained](#)
Interface for handling the event when the input authority is gained.
- interface [IInputAuthorityLost](#)
Interface for handling the event when the input authority is lost.
- interface [IInterestEnter](#)
Interface for handling the event when a player enters the area of interest.
- interface [IInterestExit](#)
Interface for handling the event when a player exits the area of interest.
- interface [ILocalPrefabCreated](#)
Interface for handling the event when a local prefab is created.
- interface [INetworkArray](#)
Defines the interface for a networked array.
- interface [INetworkAssetSource](#)
Interface for a network asset source.
- interface [INetworkDictionary](#)
Defines the interface for a networked dictionary.
- interface [INetworkInput](#)
Flag interface for custom [NetworkInput](#) structs.
- interface [INetworkLinkedList](#)
Defines the interface for a networked linked list.
- interface [INetworkObjectInitializer](#)

Interface for initializing network objects.

- interface [INetworkObjectProvider](#)

Interface which defines the handlers for `NetworkRunner` `Spawn()` and `Despawn()` actions. Passing an instance of this interface to `NetworkRunner.StartGame(StartGameArgs)` as the `StartGameArgs.ObjectProvider` argument value will assign that instance as the handler for runner `Spawn()` and `Despawn()` actions. By default (if `StartGameArgs.ObjectProvider == null`) actions will use `Instantiate()`, and `Despawn()` actions will use `Destroy()`.

- interface [INetworkPrefabSource](#)

Interface for a network prefab source.

- interface [INetworkRunnerCallbacks](#)

Interface for `NetworkRunner` callbacks. Register a class/struct instance which implements this interface with `NetworkRunner.AddCallbacks(INetworkRunnerCallbacks[])`.

- interface [INetworkRunnerUpdater](#)

Interface which defines the handlers for `NetworkRunner` Updates. An implementation is responsible for calling `NetworkRunner.UpdateInternal(double)` and `NetworkRunner.RenderInternal` periodically.

- interface [INetworkSceneManager](#)

Interface for a `NetworkRunner` scene manager. A scene manager is responsible for loading and unloading scenes

- interface [INetworkStruct](#)

Base interface for all `Fusion` Network Structs

- interface [INetworkTRSPTeleport](#)

Implement this interface on a `NetworkTRSP` implementation to indicate it can be teleported.

- class [InlineHelpAttribute](#)

If applied to a field, checks if there is a help text for the field or the field's type and shows it in the inspector.

- class [InterpolatedErrorCorrectionSettings](#)

A set of parameters that tune the interpolated correction of prediction error on transform data.

- interface [IPlayerJoined](#)

Interface for handling the event when a player joins the game.

- interface [IPlayerLeft](#)

Interface for handling the event when a player leaves the game.

- interface [IRemotePrefabCreated](#)

Interface for handling the event when a remote prefab is created.

- interface [ISceneLoadDone](#)

Interface for handling the event when a scene load operation is completed.

- interface [ISceneLoadStart](#)

Interface for handling the event when a scene load operation is started.

- interface [ISimulationEnter](#)

Interface for `SimulationEnter` callback. Called when the `NetworkObject` joins `AreaOfInterest`. Implement this interface on `SimulationBehaviour` and `NetworkBehaviour` classes.

- interface [ISimulationExit](#)

Interface for the `SimulationExit` callback. Called when the `NetworkObject` leaves `AreaOfInterest`. Implement this interface on `SimulationBehaviour` and `NetworkBehaviour` classes.

- interface [ISpawned](#)

Interface for handling the event when an object is spawned.

- interface [IStateAuthorityChanged](#)

Interface for handling the event when the state authority changes.

- struct [LagCompensatedHit](#)

Defines a lag compensated query hit result.

- class [LagCompensationSettings](#)

Settings for lag compensation history.

- class [LayerAttribute](#)

Specifies that an int field should be drawn as a layer field in the inspector.

- class [LayerMatrixAttribute](#)

Specifies that the integer array field should be drawn as a layer matrix in the inspector.

- class [LobbyInfo](#)
Holds information about a Lobby
- class [LogSimpleUnity](#)
Log Simple Unity
- struct [Mask256](#)
Mask256 is a 256-bit mask that can be used to store 256 boolean values.
- class [MaxStringByteCountAttribute](#)
Specifies that the string field should be drawn as a text field with a maximum byte count for given encoding.
- class [NestedComponentUtilities](#)
Tools to replace GetComponent variants that respects nested objects. These are used to find components of a NetworkedObjects without also finding components that belong to parent or child NetworkedObjects.
- struct [NetworkArray](#)
Fusion type for networking arrays. Maximum capacity is fixed, and is set with the CapacityAttribute.
- class [NetworkArrayExtensions](#)
Provides extension methods for the NetworkArray class.
- struct [NetworkArrayReadOnly](#)
Provides a read-only view of a network array.
- class [NetworkAssemblyIgnoreAttribute](#)
Network Assembly Ignore Attribute
- class [NetworkAssemblyWeavedAttribute](#)
Network Assembly Weaved Attribute
- class [NetworkBehaviour](#)
Base class for Fusion network components, which are associated with a NetworkObject.
- struct [NetworkBehaviourBuffer](#)
Provides low level accesss to data buffers that can be read using a NetworkBehaviour.Reader
- struct [NetworkBehaviourBufferInterpolator](#)
*The NetworkBehaviourBufferInterpolator struct is used to interpolate between two NetworkBehaviourBuffer instances.
This is a read-only, ref struct, meaning it cannot be boxed and it can only be used on the stack.*
- struct [NetworkBehaviourId](#)
Represents the unique identifier for a NetworkBehaviour instance.
- class [NetworkBehaviourUtils](#)
This static class provides utility methods for working with NetworkBehaviour objects.
- class [NetworkBehaviourWeavedAttribute](#)
Network Behaviour Weaved Attribute
- struct [NetworkBool](#)
Represents a boolean value that can be networked.
- struct [NetworkButtons](#)
Represents a set of buttons that can be networked.
- class [NetworkConfiguration](#)
Main network configuration class.
- class [NetworkDelegates](#)
Network Runner Callbacks Delegates
- class [NetworkDeserializeMethodAttribute](#)
Network Deserialize Method Attribute
- struct [NetworkDictionary](#)
Fusion type for networking Dictionaries. Maximum capacity is fixed, and is set with the CapacityAttribute.
- struct [NetworkDictionaryReadOnly](#)
A read-only version of NetworkDictionary< TKey, TValue >.

- class [NetworkedAttribute](#)
- class [NetworkedWeavedAttribute](#)
 - Networked Weaved Attribute*
- class [NetworkEvents](#)
 - Companion component for NetworkRunner. Exposes INetworkRunnerCallbacks as UnityEvents, which can be wired up to other components in the inspector.*
- struct [NetworkId](#)
 - The unique identifier for a network entity.*
- struct [NetworkInput](#)
 - NetworkInput Struct*
- class [NetworkInputUtils](#)
 - Utility methods for NetworkInput*
- class [NetworkInputWeavedAttribute](#)
 - Network Input Weaved Attribute*
- struct [NetworkLinkedList](#)
 - Fusion type for networking LinkedLists. Maximum capacity is fixed, and is set with the CapacityAttribute.*
 - Typical Usage:*
 - struct [NetworkLinkedListReadOnly](#)
 - Read-only version of NetworkLinkedList<T>.*
 - struct [NetworkLoadSceneParameters](#)
 - Parameters for loading a scene*
 - class [NetworkMecanimAnimator](#)
 - A component for synchronizing the Animator controller state from the State Authority to network proxies. Requires a Unity Animator component, and a NetworkObject component. NOTE: Animator Root Motion is not compatible with re-simulation and prediction.*
 - class [NetworkObject](#)
 - The primary Fusion component for networked GameObject entities. This stores the object's network identity and manages the object's state and input authority.*
 - class [NetworkObjectFlagsExtensions](#)
 - Extension methods for the NetworkObjectFlags enum.*
 - struct [NetworkObjectGuid](#)
 - NetworkObjectGuid*
 - struct [NetworkObjectHeader](#)
 - Network object header information for a NetworkObject.*
 - struct [NetworkObjectHeaderPtr](#)
 - Represents a pointer to a NetworkObjectHeader. This struct is unsafe because it uses pointers.*
 - class [NetworkObjectInitializerUnity](#)
 - Initializes network objects for Unity.*
 - class [NetworkObjectMeta](#)
 - Meta information about a network object.*
 - struct [NetworkObjectNestingKey](#)
 - A key used to identify a network object nesting.*
 - class [NetworkObjectPrefabData](#)
 - This class represents the data for a network object prefab.*
 - class [NetworkObjectProviderDummy](#)
 - A dummy implementation of the INetworkObjectProvider interface. This class is used for testing purposes and throws a NotImplementedException for all its methods.*
 - struct [NetworkObjectReleaseContext](#)
 - Represents the context for releasing a network object. This struct is unsafe because it uses pointers.*
 - class [NetworkObjectSortKeyComparer](#)

- This class is used to compare two [NetworkObject](#) instances based on their SortKey. It implements the [IComparer](#) interface.
- class [NetworkObjectSpawnException](#)
 Network Object Spawn Exception
 - struct [NetworkObjectTypeld](#)
 ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable.
 - struct [NetworkPhysicsInfo](#)
 Network Physics INetworkStruct
 - struct [NetworkPrefabAcquireContext](#)
 Represents the context for acquiring a prefab instance for a network object. This struct is unsafe because it uses pointers.
 - class [NetworkPrefabAttribute](#)
 Network Prefab Attribute
 - struct [NetworkPrefabId](#)
 ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable.
 - struct [NetworkPrefabInfo](#)
 Meta data for a NetworkObject prefab which has been cataloged in a NetworkProjectConfig.PrefabTable.
 - struct [NetworkPrefabRef](#)
 NetworkPrefabRef
 - class [NetworkPrefabTable](#)
 Class representing a table of network prefabs.
 - struct [NetworkPrefabTableOptions](#)
 Options for the NetworkPrefabTable.
 - class [NetworkProjectConfig](#)
 The core Fusion config file that is shared with all peers at startup.
 - class [NetworkProjectConfigAsset](#)
 Manages and references the current instance of NetworkProjectConfig
 - struct [NetworkRNG](#)
 PCG32 random generator, 16 bytes in size. <http://www.pcg-random.org>
 - class [NetworkRpcStaticWeavedInvokerAttribute](#)
 Network Rpc Static Weaved Invoker Attribute Contains info about a static weaved RPC Method
 - class [NetworkRpcWeavedInvokerAttribute](#)
 Network Rpc Weaved Invoker Attribute Contains info about a weaved RPC Method
 - class [NetworkRunner](#)
 Host Migration related code in order to get a copy of the Simulation State
 - class [NetworkRunnerCallbackArgs](#)
 Stores data types used on the INetworkRunnerCallbacks interface
 - class [NetworkRunnerUpdaterDefault](#)
 Default implementation of INetworkRunnerUpdater that uses the Unity PlayerLoop.
 - struct [NetworkRunnerUpdaterDefaultInvokeSettings](#)
 Settings for the NetworkRunnerUpdaterDefault.
 - struct [NetworkSceneAsyncOp](#)
 A wrapper for async scene operations.
 - struct [NetworkSceneInfo](#)
 Can store up to 8 active scenes and allows for duplicates. Each write increases Version which can be used to generate unique scene objects ids for when a scene is supposed to be reloaded.
 - struct [NetworkSceneLoadId](#)
 A unique identifier for a scene load operation.
 - struct [NetworkSceneObjectId](#)
 A unique identifier for a scene object.
 - class [NetworkSerializeMethodAttribute](#)

- Network Serialize Method Attribute*
- class [NetworkSimulationConfiguration](#)
 Configuration for network conditions simulation (induced latency and loss).
- struct [NetworkSpawnOp](#)
 Spawn Operation
- class [NetworkString](#)
 Fixed-size UTF32 string. All operations are alloc-free, except for converting to System.String.
- class [NetworkStructUtils](#)
 Utility methods for INetworkStruct
- class [NetworkStructWeavedAttribute](#)
 Describes the total number of WORDs a INetworkStruct uses.
- class [NetworkTransform](#)
 Add to any NetworkObject Transform, or its associated child Transforms to automatically synchronize TRSP (Position/Rotation/Scale/Parent).
- class [NetworkTRSP](#)
 Base class for spatial (Position/Rotation/Scale/Parent) synchronization component, such as NetworkTransform. Provides the base logic for render interpolation, parenting synchronization, and teleport, that can be used in components derived from this class.
- struct [NetworkTRSPData](#)
 Data structure storing spatial (Position/Rotation/Scale/Parent) synchronization data for spatial synchronization components, NetworkTRSP and its subclass NetworkTransform.
- class [NormalizedRectAttribute](#)
 Enables a special inspector drawer for Unity Rect type, specially designed for editing RectTransforms using normalized values.
- class [OnChangedRenderAttribute](#)
 OnChangedRender Attribute
- struct [PlayerRef](#)
 Represents a Fusion player.
- class [PreserveInPluginAttribute](#)
 Preserve In Plugin Attribute
- struct [Ptr](#)
 Ptr
- struct [QuaternionCompressed](#)
 Represents a compressed Quaternion value for network transmission.
- class [RangeExAttribute](#)
 Represents an attribute that specifies a range of values for a field or property.
- class [ReadOnlyAttribute](#)
 Attribute used to mark a field as read-only.
- class [ReadWriteUtils](#)
 Provides utility methods for reading and writing data.
- class [ReadWriteUtilsForWeaver](#)
 Provides utility methods for reading and writing data.
- class [ReflectionUtils](#)
 Provides utility methods for reflection.
- class [RenderAttribute](#)
 Override default render settings for [Networked] properties.
- struct [RenderTimeline](#)
 Can be used to acquire interpolated data for different points in time.
- class [RenderWeavedAttribute](#)
 Render Weaved Attribute
- class [ResolveNetworkPrefabSourceAttribute](#)
 Resolve Network Prefab Source Attribute

- class [RpcAttribute](#)

Flags a method as being a networked Remote Procedure Call. Only usable in a [NetworkBehaviour](#). Calls to this method (from the indicated allowed [RpcSources](#)) will generate a network message, which will execute the method remotely on the indicated [RpcTargets](#). The RPC method can include an empty [RpclInfo](#) argument, that will include meta information about the RPC on the receiving peer.
- struct [RpcHeader](#)

Header for RPC messages.
- struct [RpclInfo](#)

[RpclInfo](#) is a struct that contains information about the RPC message.
- struct [RpclInvokeData](#)

Represents the data required to invoke an RPC message.
- struct [RpclInvokeInfo](#)

May be used as an optional [RpcAttribute](#) return value. Contains meta data about the RPC send, such as failure to send reasons, culling, message size, etc.
- struct [RpcSendResult](#)

RPC send operation result information.
- class [RpcTargetAttribute](#)

RPC attribute used to indicate a specific target player for an RPC when sending from one player to another. RPC is sent to the server, and then is forwarded to the specified player. Usage:
- struct [SceneLoadDoneArgs](#)

Struct that contains information about a scene after it has been loaded.
- class [ScenePathAttribute](#)

Specifies that a string field represents a scene path.
- struct [SceneRef](#)

Scene reference struct. Can be used to reference a scene by index or by path.
- class [ScriptHelpAttribute](#)

Defines the appearance of the script header in the Unity inspector.
- class [SerializableDictionary](#)

A serializable dictionary.
- struct [SerializableType](#)

A System.Type wrapper that can be serialized.
- class [SerializableTypeAttribute](#)

Specifies that either a string field represents a type name or sets additional options for [SerializableType](#) field.
- class [SerializeReferenceTypePickerAttribute](#)

Attribute used to show a type picker for a field with [SerializeReference].
- class [SessionInfo](#)

Holds information about the Game Session
- class [Simulation](#)

Main simulation class
- class [SimulationBehaviour](#)

Base class for a Fusion aware Behaviour (derived from UnityEngine.MonoBehaviour). If a [SimulationBehaviour](#) is found on a [NetworkRunner](#) game object during the runner initialisation, the [SimulationBehaviour](#) is automatically registered. Objects derived from this object can be associated with a [NetworkRunner](#) and [Simulation](#) using [NetworkRunner.AddGlobal\(\)](#).
- class [SimulationBehaviourAttribute](#)

Attribute for specifying which [SimulationStages](#) and [SimulationModes](#) this [SimulationBehaviour](#) will execute in. Can be used to limit execution to only Host, Server or Client peers, or to only execute on Resimulation or Forward ticks. Usage:
- struct [SimulationBehaviourListScope](#)

Provides a scope for a [SimulationBehaviourUpdater.BehaviourList](#), incrementing its lock count on creation and decrementing it on disposal. If the lock count reaches zero on disposal, all pending removals in the list are performed.
- class [SimulationConfig](#)

Project configuration settings specific to how the [Simulation](#) class behaves.

- class [SimulationInput](#)
Simulation Input
- struct [SimulationInputHeader](#)
Simulation Input Header
- struct [SimulationMessage](#)
Simulation Message
- struct [SimulationMessagePtr](#)
Simulation Message Pointer
- struct [SimulationRuntimeConfig](#)
Stores the runtime configuration of the simulation
- struct [StartGameArgs](#)
Fusion Start Arguments, used to configure the simulation mode and other settings
- class [StartGameResult](#)
Represents the result of starting the Fusion Simulation
- struct [Tick](#)
A tick is a 32-bit integer that represents a frame number.
- struct [TickAccumulator](#)
A tick accumulator.
- struct [TickRate](#)
A tick rate is a collection of tick rates.
- struct [TickTimer](#)
A timer that is based on ticks instead of seconds.
- class [TimeSyncConfiguration](#)
Time Synchronization Configuration
- class [ToggleLeftAttribute](#)
Specifies that the bool field should be drawn as the toggle on the left side of the label.
- class [UnitAttribute](#)
Unit Attribute class. Used to mark a field with the respective Units
- class [UnityAddressablesRuntimeKeyAttribute](#)
Specifies that the string field represents a key for Unity Addressables.
- class [UnityAssetGuidAttribute](#)
Specifies that the string field represents a GUID of an asset.
- class [UnityContextMenuMenuItemAttribute](#)
Unity ContextMenuItemAttribute
- class [UnityDelayedAttribute](#)
Unity DelayedAttribute
- class [UnityFormerlySerializedAsAttribute](#)
Unity FormerlySerializedAsAttribute
- class [UnityHeaderAttribute](#)
Unity HeaderAttribute
- class [UnityMinAttribute](#)
Unity MinAttribute
- class [UnityMultilineAttribute](#)
Unity MultilineAttribute
- class [UnityNonReorderableAttribute](#)
Unity NonReorderableAttribute
- class [UnityNonSerializedAttribute](#)
Unity NonSerializedAttribute
- class [UnityRangeAttribute](#)
Unity RangeAttribute
- class [UnityResourcePathAttribute](#)

- *Specifies that the string field represents a path to a Unity resource.*
- class [UnitySerializeField](#)
Unity SerializeField
- class [UnitySerializeReference](#)
Unity SerializeReference
- class [UnitySpaceAttribute](#)
Unity SpaceAttribute
- class [UnityTooltipAttribute](#)
Unity TooltipAttribute
- struct [Vector2Compressed](#)
Represents a compressed Vector2 value for network transmission.
- struct [Vector3Compressed](#)
Represents a compressed Vector3 value for network transmission.
- struct [Vector4Compressed](#)
Represents a compressed Vector4 value for network transmission.
- class [WarnIfAttribute](#)
Editor attribute for adding notices to fields if the condition member evaluates as true. Condition member can be a property, field or method (with a return value).
- class [WeaverGeneratedAttribute](#)
Weaver Generated Attribute

Enumerations

- enum class [AnimatorSyncSettings](#)
- enum class [CompareOperator](#)
Comparison method for evaluating condition member value against compareToValues.
- enum class [ConnectionType](#)
Defines the type of the current connection with the Remote Peer, either the Server or a Client
- enum class [DrawIfMode](#)
Mode for the DrawIf attribute. If the condition is not met, should the field be hidden or just read-only?
- enum class [EditorButtonVisibility](#)
Specifies the visibility options for an editor button.
- enum class [GameMode](#)
Fusion Game Mode.
- enum class [HitboxTypes](#)
Defines the collision geometry type of a Hitbox.
- enum class [HitOptions](#)
- enum class [NetworkObjectAcquireResult](#)
Enum representing the possible results of acquiring a prefab instance for a network object.
- enum class [NetworkObjectConnectionDataStatus](#)
- enum class [NetworkObjectDestroyFlags](#)
- enum class [NetworkObjectFlags](#) : int
Enum representing the flags for network objects in the Fusion system. This enum is marked with the Flags attribute, allowing it to be treated as a bit field.
- enum class [NetworkObjectHeaderFlags](#) : int
Enum representing various flags for a network object header. Each flag represents a different state or property of a network object.
- enum class [NetworkObjectHeaderPlayerDataFlags](#) : int
- enum class [NetworkObjectPacketFlags](#)
- enum class [NetworkObjectRuntimeFlags](#) : int
- enum class [NetworkPrefabTableGetPrefabResult](#)

- Enum representing the possible results of attempting to get a prefab from the NetworkPrefabTable.*
- enum class [NetworkSceneInfoChangeSource](#)

What has contributed to the observed change in the scene info.
 - enum class [NetworkSceneInfoDefaultFlags](#) : uint

Network Scene Info Default Flags
 - enum class [NetworkSpawnFlags](#) : short

Network Spawn Flags
 - enum class [NetworkSpawnStatus](#) : int

Network Spawn Status
 - enum class [NetworkTypeIdKind](#)

Enum representing the type of a NetworkObject.
 - enum class [PageSizes](#)

Page Bit Shift Lookup Table
 - enum class [PriorityLevel](#)

Enum representing the priority levels for network objects
 - enum class [RenderSource](#)

Indicates how available snapshot data should be used to render networked properties (in the chosen Render-<→ Timeframe).
 - enum class [RenderTimeframe](#)

Indicates which point in time (or "timeframe") networked properties should be rendered in.
 - enum class [RpcChannel](#)

Flags for the RPC channel.
 - enum class [RpcHostMode](#)

Options for when the game is run in SimulationModes.Host mode and RPC is invoked by the host.
 - enum class [RpcLocalInvokeResult](#)

Results for the local RPC Invocation of the RPC method.
 - enum class [RpcSendCullResult](#)

Results for the RPC message send operation. Note: Some individual targets may be culled even if the send operation succeeds. Information about culled targets can be found in RpcInvokeInfo.SendResult.
 - enum class [RpcSendMessageResult](#)

Result flags for the RPC message send operation.
 - enum class [RpcSources](#)

Enum representing the sources of an RPC message.
 - enum class [RpcTargets](#)

Enum representing the targets of an RPC message.
 - enum class [RpcTargetStatus](#)

Enum representing the status of an RPC target.
 - enum class [ScriptHeaderBackColor](#)

Color of the component graphic header in the Unity inspector. None indicates no header graphic should be used.
 - enum class [ScriptHeaderIcon](#)

Icon to be rendered on the component graphic header in the Unity inspector.
 - enum class [ScriptHeaderStyle](#)

Style of the script header in the Unity inspector.
 - enum class [SessionLobby](#)

Session Lobby Type
 - enum class [ShutdownReason](#)

Describes a list of Reason why the [Fusion](#) Runner was Shutdown
 - enum class [SimulationBehaviourRuntimeFlags](#)
 - enum class [SimulationMessageInternalTypes](#)
 - enum class [SimulationModes](#)

Flags for The type of network peer a simulation represents.

- enum class [SimulationStages](#)
Flags for which stage the simulation currently running. Forward is when a tick is being simulated for the first time. Resimulate is when a tick is being simulated again with corrections.
- enum class [Topologies](#)
- enum class [Units](#)
Unit Type for a certain field. This helps to identify the unit that a certain value represents, like Seconds or Percentage
- enum class [UnityPlayerLoopSystemAddMode](#)
Enum representing the possible modes for adding a system to the Unity player loop.

Functions

- delegate void [FusionGlobalScriptableObjectUnloadDelegate](#) ([FusionGlobalScriptableObject](#) instance)
A delegate that can be used to unload a FusionGlobalScriptableObject.
- delegate void [NetworkObjectSpawnDelegate](#) ([NetworkSpawnOp](#) result)
Network Object Spawn Delegate
- unsafe delegate void [RpclnvokeDelegate](#) ([NetworkBehaviour](#) behaviour, [SimulationMessage](#) *message)
Represents a delegate that can be invoked by an RPC message.
- unsafe delegate void [RpcStaticInvokeDelegate](#) ([NetworkRunner](#) runner, [SimulationMessage](#) *message)
Represents a delegate that can be invoked by an RPC message.

5.1.1 Enumeration Type Documentation

5.1.1.1 CompareOperator

```
enum CompareOperator [strong]
```

Comparison method for evaluating condition member value against compareToValues.

Enumerator

Equal	true if condition member value equals compareToValue.
NotEqual	true if condition member value is not equal to compareToValue.
Less	true if condition member value is less than compareToValue.
LessOrEqual	true if condition member value is less than or equal to compareToValue.
GreaterOrEqual	true if condition member value is greater than or equal to compareToValue.
Greater	true if condition member value is greater than compareToValue.
NotZero	Returns true if the condition member evaluates to anything other than zero. In the case of object references, this means true for any non-null value.
IsZero	Returns true if the condition member evaluates to zero. In the case of object references, this means true for any null value.
BitwiseAndNotEqualZero	Returns true if the bitwise AND of the condition member and compareToValue is not zero.

5.1.1.2 ConnectionType

```
enum ConnectionType [strong]
```

Defines the type of the current connection with the Remote Peer, either the Server or a Client

Enumerator

None	No connection is currently active
Relayed	Connection was accomplished using the Photon Relay Services
Direct	Connection was accomplished directly with the remote peer

5.1.1.3 DrawIfMode

```
enum DrawIfMode [strong]
```

Mode for the DrawIf attribute. If the condition is not met, should the field be hidden or just read-only?

Enumerator

ReadOnly	Field is read-only if the condition is not met.
Hide	Field is hidden if the condition is not met.

5.1.1.4 EditorButtonVisibility

```
enum EditorButtonVisibility [strong]
```

Specifies the visibility options for an editor button.

Enumerator

EditMode	The button is only visible in Edit Mode.
Always	The button is always visible.

5.1.1.5 GameMode

```
enum GameMode [strong]
```

Fusion Game Mode.

Used to select how the local simulation will act.

Enumerator

Single	Single Player Mode: it works very similar to Host Mode, but don't accept any connections.
Shared	Shared Mode: starts a Game Client, which will connect to a Game Server running in the Photon Cloud using the Fusion Plugin.
Server	Server Mode: starts a Dedicated Game Server with no local player.
Host	Host Mode: starts a Game Server and allows a local player.
Client	Client Mode: starts a Game Client, which will connect to a peer in either Server or Host Modes.
AutoHostOrClient	Automatically start as Host or Client. The first peer to connect to a room will be started as a Host, all others will connect as clients.

5.1.1.6 HitboxTypes

```
enum HitboxTypes [strong]
```

Defines the collision geometry type of a [Hitbox](#).

Enumerator

None	[Future Use] to represent a disabled Hitbox .
Box	Geometry is a box, fill in Extents and (optional) Offset.
Sphere	Geometry is a sphere, fill in Radius and (optional) Offset.
Capsule	Geometry is a capsule, fill in capsule Radius, capsule Height and (optional) Offset.

5.1.1.7 HitOptions

```
enum HitOptions [strong]
```

Per-query options for lag compensation (both raycast and overlap).

Enumerator

None	Default, no extra options.
IncludePhysX	Add this to include checks against PhysX colliders.
IncludeBox2D	Add this to include checks against Box2D colliders. If PhysX flag is set, it will be used instead.
SubtickAccuracy	Subtick accuracy query (exactly like seen by player).
IgnoreInputAuthority	If the HitboxRoot objects which the player performing the query (if specified) has input authority over should be ignored by the query.

5.1.1.8 NetworkObjectAcquireResult

```
enum NetworkObjectAcquireResult [strong]
```

Enum representing the possible results of acquiring a prefab instance for a network object.

Enumerator

Success	Indicates that the prefab instance was successfully acquired.
Failed	Indicates that the acquisition of the prefab instance failed.
Retry	Indicates that the acquisition of the prefab instance should be retried.
Ignore	Indicates that the acquisition of the prefab instance should be ignored.

5.1.1.9 NetworkObjectFlags

```
enum NetworkObjectFlags : int [strong]
```

Enum representing the flags for network objects in the [Fusion](#) system. This enum is marked with the Flags attribute, allowing it to be treated as a bit field.

Enumerator

None	Represents a state where no flags are set.
MaskVersion	Mask for isolating the version part of the flags.
V1	Represents the first version of the network object flags.
Ignore	Flag indicating that the network object should be ignored.
MasterClientObject	Flag indicating that the network object is a master client object.
DestroyWhenStateAuthorityLeaves	Flag indicating that the network object should be destroyed when the state authority leaves.
AllowStateAuthorityOverride	Flag indicating that the network object allows state authority override.

5.1.1.10 NetworkObjectHeaderFlags

```
enum NetworkObjectHeaderFlags : int [strong]
```

Enum representing various flags for a network object header. Each flag represents a different state or property of a network object.

Enumerator

GlobalObjectInterest	Flag indicating that the object is of global interest.
DestroyWhenStateAuthorityLeaves	Flag indicating that the object should be destroyed when the state authority leaves.
SpawnedByClient	Flag indicating that the object was spawned by a client.

Enumerator

AllowStateAuthorityOverride	Flag indicating that the state authority override is allowed.
Struct	Flag indicating that the object is a struct.
StructArray	Flag indicating that the object is an array of structs.
DontDestroyOnLoad	Flag indicating that the object should not be destroyed on load.
HasMainNetworkTRSP	Flag indicating that the object has a main network TRSP.
AreaOfInterest	Flag indicating that the object is in an area of interest.

5.1.1.11 NetworkPrefabTableGetPrefabResult

```
enum NetworkPrefabTableGetPrefabResult [strong]
```

Enum representing the possible results of attempting to get a prefab from the [NetworkPrefabTable](#).

Enumerator

Success	The prefab was successfully retrieved.
InProgress	The retrieval of the prefab is in progress.
NotFound	The prefab was not found in the NetworkPrefabTable .
LoadError	There was an error in loading the prefab.

5.1.1.12 NetworkSceneInfoChangeSource

```
enum NetworkSceneInfoChangeSource [strong]
```

What has contributed to the observed change in the scene info.

Enumerator

None	No change.
Initial	The initial local scene has changed.
Remote	The remote scene has changed.

5.1.1.13 NetworkSceneInfoDefaultFlags

```
enum NetworkSceneInfoDefaultFlags : uint [strong]
```

Network Scene Info Default Flags

Enumerator

SceneCountMask	The scene count mask
ConterMask	The counter mask

5.1.1.14 NetworkSpawnFlags

```
enum NetworkSpawnFlags : short [strong]
```

Network Spawn Flags

Enumerator

DontDestroyOnLoad	Object get spawned as DontDestroyOnLoad on all clients.
SharedModeStateAuthMasterClient	In shared mode, override the state authority to PlayerRef.MasterClient , ignoring "Is Master Client Object" inspector setting. If used by a non-master client, object will be spawned with local player authority and an error message will be logged.
SharedModeStateAuthLocalPlayer	In shared mode, override the state authority to local player, ignoring "Is Master Client Object" inspector setting.

5.1.1.15 NetworkSpawnStatus

```
enum NetworkSpawnStatus : int [strong]
```

Network Spawn Status

Enumerator

Queued	Spawn is queued and will be spawned when the prefab is loaded.
Spawned	Spawned successfully.
FailedToLoadPrefabSynchronously	Failed to Load Prefab Synchronously.
FailedToCreateInstance	Failed to create instance.
FailedClientCantSpawn	Failed to spawn because the client can't spawn.
FailedLocalPlayerNotYetSet	Failed to spawn because the local player is not yet set.

5.1.1.16 NetworkTypeIdKind

```
enum NetworkTypeIdKind [strong]
```

Enum representing the type of a [NetworkObject](#).

Enumerator

Prefab	Represents a NetworkObject that is a Prefab.
Custom	Represents a NetworkObject that is a Custom type.
InternalStruct	Represents a NetworkObject that is an InternalStruct.
SceneObject	Represents a NetworkObject that is a SceneObject.
Invalid	Represents an Invalid NetworkObject type.

5.1.1.17 PageSizes

```
enum PageSizes [strong]
```

Page Bit Shift Lookup Table

5.1.1.18 PriorityLevel

```
enum PriorityLevel [strong]
```

Enum representing the priority levels for network objects

Enumerator

Player	Priority level assigned to player-related network objects.
High	High priority level, typically assigned to network objects that require frequent updates.
Medium	Medium priority level, typically assigned to network objects that require regular updates.
Low	Low priority level, typically assigned to network objects that require less frequent updates.
Lowest	Lowest priority level, typically assigned to network objects that require minimal updates.

5.1.1.19 RenderSource

```
enum RenderSource [strong]
```

Indicates how available snapshot data should be used to render networked properties (in the chosen [RenderTimeframe](#)).

Enumerator

Interpolated	The rendered value will come from interpolating the values at From and To to the desired point in time.
From	The rendered value will come from the nearest available snapshot at or before the point in time being rendered.
To	The rendered value will come from the nearest available snapshot ahead of the point in time being rendered.
Latest	The rendered value will come from the latest snapshot.

5.1.1.20 RenderTimeframe

enum `RenderTimeframe` [strong]

Indicates which point in time (or "timeframe") networked properties should be rendered in.

Enumerator

Local	The default timeframe for owned and predicted objects.
Remote	The default timeframe for proxied objects.

5.1.1.21 RpcChannel

enum `RpcChannel` [strong]

Flags for the RPC channel.

Enumerator

Reliable	Rpc order preserved, delivery verified, resend in case of a failed delivery.
Unreliable	Rpc order preserved, delivery not verified, no resend attempts.

5.1.1.22 RpcHostMode

enum `RpcHostMode` [strong]

Options for when the game is run in `SimulationModes.Host` mode and RPC is invoked by the host.

Enumerator

SourcesServer	If host invokes RPC <code>RpcInfo.Source</code> will be set to <code>PlayerRef.None</code> (default).
SourcesHostPlayer	If host invokes RPC <code>RpcInfo.Source</code> will be set to the host's local player.

5.1.1.23 RpcLocalInvokeResult

enum `RpcLocalInvokeResult` [strong]

Results for the local RPC Invocation of the RPC method.

Enumerator

Invoked	RPC has been invoked locally.
NotInvokableLocally	Not invoked locally because RpcAttribute.InvokeLocal is false.
NotInvokableDuringResim	Not invoked locally because InvokeResim is false and simulation stage is SimulationStages.Resimulate
InsufficientSourceAuthority	Not invoked because source NetworkObject current authority does not match flags set in RpcAttribute.Sources
InsufficientTargetAuthority	Not invoked because target player is local and this NetworkObject current authority does not match flags set in RpcAttribute.Targets
TargetPlayerIsNotLocal	Not invoked because target player is not local.
PayloadSizeExceeded	RPC is too large. See RpcAttribute.MaxPayloadSize for the maximum allowed size.
TagetPlayerIsNotLocal	TargetPlayerIsNotLocal

5.1.1.24 RpcSendCullResult

```
enum RpcSendCullResult [strong]
```

Results for the RPC message send operation. Note: Some individual targets may be culled even if the send operation succeeds. Information about culled targets can be found in [RpclInvokeInfo.SendResult](#).

Enumerator

NotCulled	RPC has been sent. Check RpclInvokeInfo.SendResult for details.
NotInvokableDuringResim	Send culled because RpcAttribute.InvokeLocal is false.
InsufficientSourceAuthority	Send culled because source NetworkObject current authority does not match flags set in RpcAttribute.Sources
NoActiveConnections	Send culled because there are no active connections.
TargetPlayerUnreachable	Send culled because target player does not exist.
TargetPlayerIsLocalButRpclsNotInvokableLocally	Send culled because target player is local and RpcAttribute.InvokeLocal is false.
PayloadSizeExceeded	RPC message size is too large to be sent. See RpcAttribute.MaxPayloadSize for the maximum allowed size.

5.1.1.25 RpcSendMessageResult

```
enum RpcSendMessageResult [strong]
```

Result flags for the RPC message send operation.

Enumerator

	None	Invalid result.
--	------	-----------------

Enumerator

SentToServerForForwarding	Client sent to the server, server will send to the target client.
SentToTargetClient	Server sent to a specific client (a targeted message).
SentBroadcast	Server attempted to send to all the clients and at least one succeeded.
NotSentTargetObjectNotConfirmed	Target object not confirmed on the client.
NotSentTargetObjectNotInPlayerInterest	Target object not in client's interest. Likely due to being outside of player's AOI region, or needs to be explicitly set as always interested.
NotSentTargetClientNotAvailable	Target client not connected (a targeted message).
NotSentBroadcastNoActiveConnections	Server attempted to send to all the clients, but none was connected.
NotSentBroadcastNoConfirmedNorInterestedClients	Server attempted to send to all the clients, but the target object is not confirmed/not in Object Interest for all target clients.
MaskSent	Mask for sent messages.
MaskNotSent	Mask for not sent messages.
MaskBroadcast	Mask for broadcast messages.
MaskCulled	Mask for culled messages.

5.1.1.26 RpcSources

```
enum RpcSources [strong]
```

Enum representing the sources of an RPC message.

Enumerator

StateAuthority	Represents the state authority source of an RPC message.
InputAuthority	Represents the input authority source of an RPC message.
Proxies	Represents the proxy source of an RPC message.
All	Represents all possible sources of an RPC message.

5.1.1.27 RpcTargets

```
enum RpcTargets [strong]
```

Enum representing the targets of an RPC message.

Enumerator

StateAuthority	Represents the state authority target of an RPC message.
InputAuthority	Represents the input authority target of an RPC message.
Proxies	Represents the proxy target of an RPC message.
Generated by Doxygen	Represents all possible targets of an RPC message.

5.1.1.28 RpcTargetStatus

```
enum RpcTargetStatus [strong]
```

Enum representing the status of an RPC target.

Enumerator

Unreachable	Represents an unreachable RPC target.
Self	Represents the RPC target as self.
Remote	Represents a remote RPC target.

5.1.1.29 ScriptHeaderBackColor

```
enum ScriptHeaderBackColor [strong]
```

Color of the component graphic header in the Unity inspector. None indicates no header graphic should be used.

5.1.1.30 ScriptHeaderIcon

```
enum ScriptHeaderIcon [strong]
```

Icon to be rendered on the component graphic header in the Unity inspector.

5.1.1.31 ScriptHeaderStyle

```
enum ScriptHeaderStyle [strong]
```

Style of the script header in the Unity inspector.

Enumerator

Unity	Use the default Unity header style.
Photon	Use the Photon header style.

5.1.1.32 SessionLobby

```
enum SessionLobby [strong]
```

Session Lobby Type

Enumerator

Invalid	Invalid Session Lobby Type
ClientServer	ClientServer Lobby
Shared	Shared Lobby
Custom	Custom Lobby - works in conjunction with a Lobby Name/ID

5.1.1.33 ShutdownReason

```
enum ShutdownReason [strong]
```

Describes a list of Reason why the [Fusion](#) Runner was Shutdown

Enumerator

Ok	OK Reason means Fusion was Shutdown by request
Error	Shutdown was caused by some internal error
IncompatibleConfiguration	Raised when the peer tries to Join a Room with a mismatching type between ClientServer Mode and Shared Mode.
ServerInRoom	Raised when the local peer started as a Server and tried to join a Room that already has a Server peer.
DisconnectedByPluginLogic	Raised when the Peer is disconnected or kicked by a Plugin Logic.
GameClosed	Raised when the Game the Peer is trying to Join is Closed
GameNotFound	Raised when the Game the Peer is trying to Join does not exist
MaxCcuReached	Raised when all CCU available for the Photon Application are in use
InvalidRegion	Raised when the peer is trying to connect to an unavailable or non-existent Region
GameIdAlreadyExists	Raised when a Session with the same name was already created
GameIsFull	Raised when a peer is trying to join a Room with already the max capacity of players
InvalidAuthentication	Raised when the Authentication Values are invalid
CustomAuthenticationFailed	Raised when the Custom Authentication has failed for some other reason
AuthenticationTicketExpired	Raised when the Authentication Ticket has expired
PhotonCloudTimeout	Timeout on the Connection with the Photon Cloud
AlreadyRunning	Raised when Fusion is already running and the StartGame is invoked again
InvalidArguments	Raised when any of the StartGame arguments does not meet the requirements
HostMigration	Signal this Runner is shutting down because of a Host Migration is about to happen
ConnectionTimeout	Connection with a remote server failed by timeout
ConnectionRefused	Connection with a remote server failed because it was refused
OperationTimeout	The current operation has timed out
OperationCanceled	The current operation was canceled

5.1.1.34 SimulationModes

```
enum SimulationModes [strong]
```

Flags for The type of network peer a simulation represents.

Enumerator

Server	Simulation represents a server peer, with no local player.
Host	Simulation represents a server peer, with a local player.
Client	Simulation represents a client peer, with a local player.

5.1.1.35 SimulationStages

```
enum SimulationStages [strong]
```

Flags for which stage the simulation currently running. Forward is when a tick is being simulated for the first time. Resimulate is when a tick is being simulated again with corrections.

Enumerator

Forward	Currently simulating a tick for the first time.
Resimulate	Currently simulating a previously simulated tick again, with state corrections.

5.1.1.36 Topologies

```
enum Topologies [strong]
```

Enumerator

ClientServer	Classic server and client model
Shared	Relay based shared world model

5.1.1.37 Units

```
enum Units [strong]
```

Unit Type for a certain field. This helps to identify the unit that a certain value represents, like Seconds or Percentage

Enumerator

None	
Ticks	ticks
Seconds	seconds - secs
MilliSecs	millisecs - ms
Kilobytes	kilobytes - kB
Megabytes	megabytes - MB
Normalized	normalized - norm
Multiplier	multiplier - mult
Percentage	%
NormalizedPercentage	normalized % - n%
Degrees	degrees - \u00B0
PerSecond	per sec - /sec
DegreesPerSecond	\u00B0 / sec - \u00B0/sec
Radians	radians - rad
RadiansPerSecond	radian / sec - rad/s
TicksPerSecond	ticks / sec - tck/s
Units	units - units
Bytes	bytes - bytes
Count	count - count
Packets	packets - packets
Frames	frames - frames
FramesPerSecond	fps - fps
SquareMagnitude	sqrMagnitude - sqrMag

5.1.1.38 UnityPlayerLoopSystemAddMode

```
enum UnityPlayerLoopSystemAddMode [strong]
```

Enum representing the possible modes for adding a system to the Unity player loop.

Enumerator

FirstChild	Add the system as the first child of the parent system.
LastChild	Add the system as the last child of the parent system.
Before	Add the system before the parent system in the player loop.
After	Add the system after the parent system in the player loop.

5.1.2 Function Documentation

5.1.2.1 FusionGlobalScriptableObjectUnloadDelegate()

```
delegate void Fusion.FusionGlobalScriptableObjectUnloadDelegate (
    FusionGlobalScriptableObject instance )
```

A delegate that can be used to unload a [FusionGlobalScriptableObject](#).

5.1.2.2 NetworkObjectSpawnDelegate()

```
delegate void Fusion.NetworkObjectSpawnDelegate (
    NetworkSpawnOp result )
```

Network Object Spawn Delegate

5.1.2.3 RpcInvokeDelegate()

```
unsafe delegate void Fusion.RpcInvokeDelegate (
    NetworkBehaviour behaviour,
    SimulationMessage * message )
```

Represents a delegate that can be invoked by an RPC message.

Parameters

<i>behaviour</i>	The NetworkBehaviour associated with the RPC message.
<i>message</i>	The SimulationMessage associated with the RPC message.

The [RpcInvokeDelegate](#) is used to invoke an RPC message. The delegate is invoked by the [RpcSystem](#) when an RPC message is received. The delegate is invoked with the [NetworkBehaviour](#) associated with the RPC message and the [SimulationMessage](#) associated with the RPC message.

5.1.2.4 RpcStaticInvokeDelegate()

```
unsafe delegate void Fusion.RpcStaticInvokeDelegate (
    NetworkRunner runner,
    SimulationMessage * message )
```

Represents a delegate that can be invoked by an RPC message.

Parameters

<i>runner</i>	The NetworkRunner associated with the RPC message.
<i>message</i>	The SimulationMessage associated with the RPC message.

The [RpcInvokeDelegate](#) is used to invoke an RPC message. The delegate is invoked by the [RpcSystem](#) when an

RPC message is received. The delegate is invoked with the [NetworkRunner](#) associated with the RPC message and the [SimulationMessage](#) associated with the RPC message.

5.2 Fusion.Analyzer Namespace Reference

Enumerations

- enum class **StaticFieldResetMode**

5.3 Fusion.Async Namespace Reference

Classes

- class [TaskManager](#)
Task Factory is used to create new Tasks and Schedule long running Tasks

5.4 Fusion.Encryption Namespace Reference

Classes

- class [DataEncryptor](#)
Responsible for encrypting and decrypting data buffers
- interface [IDataEncryption](#)
Interface for classes that manage the encryption/decryption of byte arrays

5.5 Fusion.Internal Namespace Reference

Classes

- interface [IUnitySurrogate](#)
Represents an interface for Unity surrogates. This interface provides methods for reading and writing data.
- interface [IUnityValueSurrogate](#)
Represents an interface for Unity value surrogates. This interface provides a property for accessing and modifying the data.
- class [UnityArraySurrogate](#)
A base class for Unity array surrogates.
- class [UnityDictionarySurrogate](#)
A surrogate for serializing a dictionary.
- class [UnityLinkedListSurrogate](#)
A surrogate for serializing a linked list.
- class [UnitySurrogateBase](#)
Represents a base class for Unity surrogates. This class is serializable and provides abstract methods for reading, writing, and initializing data.
- class [UnityValueSurrogate](#)
Represents a base class for Unity value surrogates. This class is serializable and provides methods for reading, writing, and initializing data.

5.6 Fusion.LagCompensation Namespace Reference

Classes

- struct [AABB](#)
Represents an Axis-Aligned Bounding Box ([AABB](#)).
- class [BoxOverlapQuery](#)
Class that represents a box overlap query. Used to query against the [NetworkRunner.LagCompensation](#) API.
- struct [BoxOverlapQueryParams](#)
Base parameters needed to execute a box overlap query
- class [BVHDraw](#)
Provide a way to iterate over BVH and return a [BVHNodeDrawInfo](#) for each node.
- class [BVHNodeDrawInfo](#)
Container class to provide the necessary info to draw nodes from the BVH
- class [ColliderDrawInfo](#)
Container class to provide the necessary information to draw a hitbox collider
- class [HitboxColliderContainerDraw](#)
Provide a way to iterate over the HitboxBuffer.HitboxSnapshot and return the [ColliderDrawInfo](#) for each collider on the snapshot.
- class [LagCompensatedExt](#)
LagCompensated Extension methods
- class [LagCompensationDraw](#)
Provide access to iterate over the lag compensation system components and give the necessary information to draw them.
- struct [PositionRotationQueryParams](#)
Query parameters for position rotation query
- class [Query](#)
Base class for all Lag Compensation queries
- struct [QueryParams](#)
Base parameters needed to execute a query.
- class [RaycastAllQuery](#)
Class that represents a raycast all query. Used to query against the [NetworkRunner.LagCompensation](#) API.
- class [RaycastQuery](#)
Class that represents a raycast query. Used to query against the [NetworkRunner.LagCompensation](#) API.
- struct [RaycastQueryParams](#)
Base parameters needed to execute a raycast query
- class [SnapshotHistoryDraw](#)
Provide a way to iterate over the HitboxBuffer and return the [HitboxColliderContainerDraw](#) container for each snapshot on the buffer.
- class [SphereOverlapQuery](#)
Class that represents a sphere overlap query. Used to query against the [NetworkRunner.LagCompensation](#) API.
- struct [SphereOverlapQueryParams](#)
Base parameters needed to execute a sphere overlap query

Enumerations

- enum class [HitType](#)
Queries can hit either fusion's custom Hitbox or Unity's standard Physx/Box2D colliders.

Functions

- delegate void [PreProcessingDelegate](#) ([Query](#) query, HashSet< [HitboxRoot](#) > rootCandidates, HashSet< int > processedColliderIndices)
Pre-processing delegate for queries.

5.6.1 Enumeration Type Documentation

5.6.1.1 HitType

enum [HitType](#) [strong]

Queries can hit either fusion's custom [Hitbox](#) or Unity's standard Physx/Box2D colliders.

Enumerator

None	Used when a raycast does not hit anything. Not used on overlaps.
Hitbox	LagCompensatedHit is a Fusion Hitbox .
PhysX	LagCompensatedHit is a Unity PhysX Collider.
Box2D	LagCompensatedHit is a Unity Box2D Collider.

5.6.2 Function Documentation

5.6.2.1 PreProcessingDelegate()

```
delegate void Fusion.LagCompensation.PreProcessingDelegate (
    Query query,
    HashSet< HitboxRoot > rootCandidates,
    HashSet< int > processedColliderIndices )
```

Pre-processing delegate for queries.

Parameters

<i>query</i>	The query to be performed.
<i>rootCandidates</i>	The root candidates to be used for the query.
<i>processedColliderIndices</i>	The indices of the colliders that have been processed.

5.7 Fusion.Protocol Namespace Reference

Classes

- interface [IMessage](#)
Represents a [Protocol](#) Message
- class [Versioning](#)
Versioning Information

Enumerations

- enum class [DisconnectReason](#) : byte
List all Disconnect reason used by the Plugin to remove an Actor from the Room

5.7.1 Enumeration Type Documentation

5.7.1.1 [DisconnectReason](#)

```
enum DisconnectReason : byte [strong]
```

List all Disconnect reason used by the Plugin to remove an Actor from the Room

Enumerator

ServerLogic	Abstract disconnect reason
InvalidEventCode	Used when an event with other code other then the treated ones is received by the plugin
InvalidJoinMsgType	When the Join Message is not of the Request Type
InvalidJoinGameMode	When the Join Message does not contain a valid Game Mode
IncompatibleConfiguration	When any of the major settings of a message does not align with the current settings, like GameMode, Protocol Version, Serialization Version and Peer Mode
ServerAlreadyInRoom	When there is already a Server running on the current Room
Error	An error occurred on the Plugin

5.8 Fusion.Runtime Namespace Reference

5.9 Fusion.Runtime.Unity Namespace Reference

5.10 Fusion.Sockets Namespace Reference

Classes

- struct [NetAddress](#)

- struct [NetBitBufferList](#)
Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address
- struct [NetCommandAccepted](#)
Accepted Command, sent by the server when a remote client connection is accepted
- struct [NetCommandConnect](#)
Connect Command used to signal a remote server that a client is trying to connect to it
- struct [NetCommandDisconnect](#)
Disconnect Command, it can be used by either side of the connection
- struct [NetCommandHeader](#)
Network Command Header Describe its type and usual settings for all commands
- struct [NetCommandRefused](#)
Refuse Command, sent by the server when the connection was refused. This happens when the server has reached its max connection capacity.
- struct [NetConfig](#)
General configuration used to drive the behavior of the Socket library

Enumerations

- enum class [NetCommands](#) : byte
Describe the Type of a Command Packet
- enum class [NetConnectFailedReason](#) : byte
The reason a connection with a remote server has failed
- enum class [NetConnectionStatus](#)
- enum class [NetDisconnectReason](#) : byte
Disconnect Reason Flag
- enum class [NetPacketType](#) : byte
Describe the type of a Networked Packet
- enum class [OnConnectionRequestReply](#)

5.10.1 Enumeration Type Documentation

5.10.1.1 [NetCommands](#)

```
enum NetCommands : byte [strong]
```

Describe the Type of a Command Packet

5.10.1.2 [NetConnectFailedReason](#)

```
enum NetConnectFailedReason : byte [strong]
```

The reason a connection with a remote server has failed

Enumerator

Timeout	Server is not responding.
ServerFull	Server has accepted the max allowed Players
ServerRefused	Server refused the connection

5.10.1.3 NetDisconnectReason

```
enum NetDisconnectReason : byte [strong]
```

Disconnect Reason Flag

5.10.1.4 NetPacketType

```
enum NetPacketType : byte [strong]
```

Describe the type of a Networked Packet

5.11 Fusion.Sockets.Stun Namespace Reference**Enumerations**

- enum class **NATType** : byte

Specifies UDP network type.

5.11.1 Enumeration Type Documentation**5.11.1.1 NATType**

```
enum NATType : byte [strong]
```

Specifies UDP network type.

Enumerator

Invalid	Invalid NAT Type
UdpBlocked	UDP is always blocked.
OpenInternet	No NAT, public IP, no firewall.
FullCone	A full cone NAT is one where all requests from the same internal IP address and port are mapped to the same external IP address and port. Furthermore, any external host can send a packet to the internal host, by sending a packet to the mapped external address. <small>Generated by Doxygen</small>
Symmetric	A symmetric NAT is one where all requests from the same internal IP address and port, to a specific destination IP address and port, are mapped to the same external IP address and port. If the same host sends a packet with the same source address and port, but to a different destination, a different mapping is used. Furthermore, only the external host that receives a

5.12 Fusion.Statistics Namespace Reference

Classes

- class [BehaviourStatisticsManager](#)
Behaviour statistics manager will provide access to the behaviour statistics snapshots.
- class [BehaviourStatisticsSnapshot](#)
Represents a snapshot of statistics related to [Fusion Behaviour](#) type execution.
- class [FusionStatisticsManager](#)
Represents a fusion statistics manager.
- class [FusionStatisticsSnapshot](#)
Represents a snapshot of [Fusion](#) statistics.
- class [LagCompensationStatisticsSnapshot](#)
Represents a snapshot of lag compensation statistics.
- struct [MemoryStatisticsSnapshot](#)
Represents a snapshot of specific memory statistics. For basic total allocated memory and total free memory check the [Fusion Statistics](#).
- class [NetworkObjectStatisticsManager](#)
Manages network object statistics for monitored network objects.
- class [NetworkObjectStatisticsSnapshot](#)
Represents a snapshot of network object statistics.

5.13 UnityEngine Namespace Reference

5.14 UnityEngine.SceneManagement Namespace Reference

5.15 UnityEngine.Scripting Namespace Reference

5.16 UnityEngine.Serialization Namespace Reference

Chapter 6

Class Documentation

6.1 `_128` Struct Reference

A [FixedStorage](#) that can hold up to 128 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint `Data` [128]
The data of the FixedStorage.

Static Public Attributes

- const int `SIZE` = 512
The size of the FixedStorage in bytes.

6.1.1 Detailed Description

A [FixedStorage](#) that can hold up to 128 words.

6.1.2 Member Data Documentation

6.1.2.1 Data

```
fixed uint Data[128]
```

The data of the [FixedStorage](#).

6.1.2.2 SIZE

```
const int SIZE = 512 [static]
```

The size of the [FixedStorage](#) in bytes.

6.2 _16 Struct Reference

A [FixedStorage](#) that can hold up to 16 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [16]
The data of the FixedStorage.

Static Public Attributes

- const int [SIZE](#) = 64
The size of the FixedStorage in bytes.

6.2.1 Detailed Description

A [FixedStorage](#) that can hold up to 16 words.

6.2.2 Member Data Documentation

6.2.2.1 Data

```
fixed uint Data[16]
```

The data of the [FixedStorage](#).

6.2.2.2 SIZE

```
const int SIZE = 64 [static]
```

The size of the [FixedStorage](#) in bytes.

6.3 _2 Struct Reference

A [FixedStorage](#) that can hold up to 2 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [2]
The data of the FixedStorage.

Static Public Attributes

- const int [SIZE](#) = 8
The size of the FixedStorage in bytes.

6.3.1 Detailed Description

A [FixedStorage](#) that can hold up to 2 words.

6.3.2 Member Data Documentation

6.3.2.1 Data

```
fixed uint Data[2]
```

The data of the [FixedStorage](#).

6.3.2.2 SIZE

```
const int SIZE = 8 [static]
```

The size of the [FixedStorage](#) in bytes.

6.4 _256 Struct Reference

A [FixedStorage](#) that can hold up to 256 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [256]
The data of the [FixedStorage](#).

Static Public Attributes

- const int [SIZE](#) = 1024
The size of the [FixedStorage](#) in bytes.

6.4.1 Detailed Description

A [FixedStorage](#) that can hold up to 256 words.

6.4.2 Member Data Documentation

6.4.2.1 Data

```
fixed uint Data[256]
```

The data of the [FixedStorage](#).

6.4.2.2 SIZE

```
const int SIZE = 1024 [static]
```

The size of the [FixedStorage](#) in bytes.

6.5 _32 Struct Reference

A [FixedStorage](#) that can hold up to 32 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [32]
The data of the [FixedStorage](#).

Static Public Attributes

- const int **SIZE** = 128

The size of the [FixedStorage](#) in bytes.

6.5.1 Detailed Description

A [FixedStorage](#) that can hold up to 32 words.

6.5.2 Member Data Documentation

6.5.2.1 Data

```
fixed uint Data[32]
```

The data of the [FixedStorage](#).

6.5.2.2 SIZE

```
const int SIZE = 128 [static]
```

The size of the [FixedStorage](#) in bytes.

6.6 _4 Struct Reference

A [FixedStorage](#) that can hold up to 4 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint **Data** [4]

The data of the [FixedStorage](#).

Static Public Attributes

- const int **SIZE** = 16

The size of the [FixedStorage](#) in bytes.

6.6.1 Detailed Description

A [FixedStorage](#) that can hold up to 4 words.

6.6.2 Member Data Documentation

6.6.2.1 Data

```
fixed uint Data[4]
```

The data of the [FixedStorage](#).

6.6.2.2 SIZE

```
const int SIZE = 16 [static]
```

The size of the [FixedStorage](#) in bytes.

6.7 _512 Struct Reference

A [FixedStorage](#) that can hold up to 512 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [512]
The data of the [FixedStorage](#).

Static Public Attributes

- const int [SIZE](#) = 2048
The size of the [FixedStorage](#) in bytes.

6.7.1 Detailed Description

A [FixedStorage](#) that can hold up to 512 words.

6.7.2 Member Data Documentation

6.7.2.1 Data

```
fixed uint Data[512]
```

The data of the [FixedStorage](#).

6.7.2.2 SIZE

```
const int SIZE = 2048 [static]
```

The size of the [FixedStorage](#) in bytes.

6.8 _64 Struct Reference

A [FixedStorage](#) that can hold up to 64 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [64]
The data of the [FixedStorage](#).

Static Public Attributes

- const int [SIZE](#) = 256
The size of the [FixedStorage](#) in bytes.

6.8.1 Detailed Description

A [FixedStorage](#) that can hold up to 64 words.

6.8.2 Member Data Documentation

6.8.2.1 Data

```
fixed uint Data[64]
```

The data of the [FixedStorage](#).

6.8.2.2 SIZE

```
const int SIZE = 256 [static]
```

The size of the [FixedStorage](#) in bytes.

6.9 _8 Struct Reference

A [FixedStorage](#) that can hold up to 8 words.

Inherits [INetworkStruct](#), and [IFixedStorage](#).

Public Attributes

- fixed uint [Data](#) [8]

The data of the FixedStorage.

Static Public Attributes

- const int [SIZE](#) = 32

The size of the FixedStorage in bytes.

6.9.1 Detailed Description

A [FixedStorage](#) that can hold up to 8 words.

6.9.2 Member Data Documentation

6.9.2.1 Data

```
fixed uint Data[8]
```

The data of the [FixedStorage](#).

6.9.2.2 SIZE

```
const int SIZE = 32 [static]
```

The size of the [FixedStorage](#) in bytes.

6.10 Allocator Struct Reference

Memory [Allocator](#)

Classes

- struct [Config](#)
Memory Allocator Configuration

Public Member Functions

- Block * **GetBlock** (int index)
- Block * **GetBlock** (long index)
- int **GetBlockBucket** (long index)
- Block * **GetBlockForPointer** (void *ptr)
- int **GetBlockIndexForPointer** (void *ptr)
- byte * **GetBlockMemory** (Block *block)
- byte * **GetBlockMemory** (long blockIndex)
- Bucket * **GetBucket** (int index)
- Bucket * **GetBucketForBlock** (Block *block)
- BlockList * **GetBucketList** (int index)
- bool **IsPointerInMeta** (void *p)
- void * **Meta** ([Ptr](#) ptr)
- [Ptr](#) **Meta** (void *p)

Static Public Member Functions

- static void **DebugVerifyBucketIntegrity** ([Allocator](#) *a, int index)
- static void * **TryAllocateSegmentFromBlock** ([Allocator](#) *a, Bucket *bucket, Block *block, int size)
- static int **WordCount** (int size)

Public Attributes

- Block * **_blocks**
- BlockList * **_blocksFreeList**
- Bucket * **_buckets**
- BlockList * **_bucketsLists**
- byte * **_bucketsMap**
- void * **_checksum**
- int **_checksumByteLength**
- [Config](#) **_config**
- void * **_globals**
- byte * **_heap**
- int **_maxBlockIndexUsed**
- byte * **_meta**
- void * **_replicate**
- int **_replicateByteLength**
- byte * **_root**

Static Public Attributes

- const int **BUCKET_COUNT** = 57
Bucket Count
- const byte **BUCKET_INVALID** = 255
Bucket Invalid
- const int **HEAP_ALIGNMENT** = 8
Heap Alignment
- const int **PTR_SIZE** = 8
- const int **REPLICATE_WORD_ALIGN** = **REPLICATE_WORD_SIZE**
Replicate Word Align
- const int **REPLICATE_WORD_SHIFT** = 2
Replicate Word Shift
- const int **REPLICATE_WORD_SIZE** = 1 << **REPLICATE_WORD_SHIFT**
Replicate Word Size
- const int **SIZE** = 112
- const int **WORD_BYTE_SIZE** = 1 << 3
- const int **WORD_SHIFT** = 3

6.10.1 Detailed Description

Memory [Allocator](#)

6.10.2 Member Data Documentation

6.10.2.1 BUCKET_COUNT

```
const int BUCKET_COUNT = 57 [static]
```

Bucket Count

6.10.2.2 BUCKET_INVALID

```
const byte BUCKET_INVALID = 255 [static]
```

Bucket Invalid

6.10.2.3 HEAP_ALIGNMENT

```
const int HEAP_ALIGNMENT = 8 [static]
```

Heap Alignment

6.10.2.4 REPLICATE_WORD_ALIGN

```
const int REPLICATE_WORD_ALIGN = REPLICATE_WORD_SIZE [static]
```

Replicate Word Align

6.10.2.5 REPLICATE_WORD_SHIFT

```
const int REPLICATE_WORD_SHIFT = 2 [static]
```

Replicate Word Shift

6.10.2.6 REPLICATE_WORD_SIZE

```
const int REPLICATE_WORD_SIZE = 1 << REPLICATE_WORD_SHIFT [static]
```

Replicate Word Size

6.11 Allocator.Config Struct Reference

Memory [Allocator](#) Configuration

Public Member Functions

- [Config \(PageSizes shift, int count, int globalsSize\)](#)
Config Constructor
- bool [Equals \(Config other\)](#)
Check Config equality
- override bool [Equals \(object obj\)](#)
Check Config equality
- override int [GetHashCode \(\)](#)
Get Hash Code
- override string [ToString \(\)](#)
Config ToString

Public Attributes

- int [BlockCount](#)
Block Count Config value
- int [BlockShift](#)
Block Shift Config value
- int [GlobalsSize](#)
Globals Size Config value

Static Public Attributes

- const int **DEFAULT_BLOCK_COUNT** = 256
Default Block Count
- const **PageSizes DEFAULT_BLOCK_SHIFT** = PageSizes._32Kb
Default Block Shift
- const int **SIZE** = 12
Config Size

Properties

- int **BlockByteSize** [get]
Block Size in Bytes
- int **BlockWordCount** [get]
Block Size in Words
- int **HeapSizeAllocated** [get]
Heap Size Allocated in Bytes
- int **HeapSizeUsable** [get]
Heap Size in Bytes

6.11.1 Detailed Description

Memory [Allocator](#) Configuration

6.11.2 Constructor & Destructor Documentation

6.11.2.1 Config()

```
Config (
    PageSizes shift,
    int count,
    int globalsSize )
```

[Config](#) Constructor

Parameters

<i>shift</i>	Block Shift
<i>count</i>	Block Count
<i>globalsSize</i>	Globals Size

6.11.3 Member Function Documentation

6.11.3.1 Equals() [1/2]

```
bool Equals (
    Config other )
```

Check [Config](#) equality

Parameters

<i>other</i>	Config Ref
--------------	----------------------------

Returns

True if has the same values

6.11.3.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Check [Config](#) equality

Parameters

<i>obj</i>	Any object reference
------------	----------------------

Returns

True if obj is a [Config](#) and has the same values

6.11.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Get Hash Code

Returns

Hash Code

6.11.3.4 `ToString()`

```
override string ToString ( )
```

Config `ToString`

6.11.4 Member Data Documentation

6.11.4.1 `BlockCount`

```
int BlockCount
```

Block Count Config value

6.11.4.2 `BlockShift`

```
int BlockShift
```

Block Shift Config value

6.11.4.3 `DEFAULT_BLOCK_COUNT`

```
const int DEFAULT_BLOCK_COUNT = 256 [static]
```

Default Block Count

6.11.4.4 `DEFAULT_BLOCK_SHIFT`

```
const PageSizes DEFAULT_BLOCK_SHIFT = PageSizes._32Kb [static]
```

Default Block Shift

6.11.4.5 `GlobalsSize`

```
int GlobalsSize
```

Globals Size Config value

6.11.4.6 SIZE

```
const int SIZE = 12 [static]
```

[Config Size](#)

6.11.5 Property Documentation

6.11.5.1 BlockByteSize

```
int BlockByteSize [get]
```

Block Size in Bytes

6.11.5.2 BlockWordCount

```
int BlockWordCount [get]
```

Block Size in Words

6.11.5.3 HeapSizeAllocated

```
int HeapSizeAllocated [get]
```

Heap Size Allocated in Bytes

6.11.5.4 HeapSizeUsable

```
int HeapSizeUsable [get]
```

Heap Size in Bytes

6.12 Angle Struct Reference

A Networked fusion type for degrees. This can be used with the [NetworkedAttribute](#), in RPCs, or in [NetworkInput](#) structs.

Inherits [INetworkStruct](#), and [IEquatable<Angle>](#).

Public Member Functions

- void [Clamp \(Angle min, Angle max\)](#)
Clamps the current value to the supplied min-max range.
- bool [Equals \(Angle other\)](#)
Checks equality with another [Angle](#).
- override bool [Equals \(object obj\)](#)
Checks equality with an object.
- override int [GetHashCode \(\)](#)
Gets the hash code.
- override string [ToString \(\)](#)
String representation of the [Angle](#).

Static Public Member Functions

- static Angle [Clamp \(Angle value, Angle min, Angle max\)](#)
Returns a the value, clamped to the min-max range.
- static Angle [Lerp \(Angle a, Angle b, float t\)](#)
Lerps between two angle values.
- static Angle [Max \(Angle a, Angle b\)](#)
Returns the larger of two supplied angles.
- static Angle [Min \(Angle a, Angle b\)](#)
Returns the smaller of two supplied angles.
- static implicit operator Angle (double value)
Converts double to [Angle](#).
- static implicit operator Angle (float value)
Converts float to [Angle](#).
- static implicit operator Angle (int value)
Converts int to [Angle](#).
- static operator double (Angle value)
Converts [Angle](#) to double.
- static operator float (Angle value)
Converts [Angle](#) to float.
- static bool [operator!= \(Angle a, Angle b\)](#)
Inequality operator for [Angle](#) struct.
- static Angle [operator+ \(Angle a, Angle b\)](#)
Addition operator for [Angle](#) struct.
- static Angle [operator- \(Angle a, Angle b\)](#)
Subtraction operator for [Angle](#) struct.
- static bool [operator< \(Angle a, Angle b\)](#)
Less than operator for [Angle](#) struct.
- static bool [operator<= \(Angle a, Angle b\)](#)
Less than or equal to operator for [Angle](#) struct.
- static bool [operator== \(Angle a, Angle b\)](#)
Equality operator for [Angle](#) struct.
- static bool [operator> \(Angle a, Angle b\)](#)
Greater than operator for [Angle](#) struct.
- static bool [operator>= \(Angle a, Angle b\)](#)
Greater than or equal to operator for [Angle](#) struct.

Public Attributes

- int `_value`

Static Public Attributes

- const int `_360` = 360 * ACCURACY
- const int `ACCURACY` = 10000
- const int `DECIMALS` = 4
- const int `SIZE` = 4

Size of this struct in bytes.

6.12.1 Detailed Description

A Networked fusion type for degrees. This can be used with the [NetworkedAttribute](#), in RPCs, or in [NetworkInput](#) structs.

6.12.2 Member Function Documentation

6.12.2.1 Clamp() [1/2]

```
void Clamp (
    Angle min,
    Angle max )
```

Clamps the current value to the supplied min-max range.

6.12.2.2 Clamp() [2/2]

```
static Angle Clamp (
    Angle value,
    Angle min,
    Angle max ) [static]
```

Returns a the value, clamped to the min-max range.

6.12.2.3 Equals() [1/2]

```
bool Equals (
    Angle other )
```

Checks equality with another `Angle`.

Parameters

<i>other</i>	Other Angle .
--------------	-------------------------------

Returns

Equality result.

6.12.2.4 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Checks equality with an object.

Parameters

<i>obj</i>	Object to compare.
------------	--------------------

Returns

Equality result.

6.12.2.5 GetHashCode()

```
override int GetHashCode ( )
```

Gets the hash code.

Returns

Hash code.

6.12.2.6 Lerp()

```
static Angle Lerp (
    Angle a,
    Angle b,
    float t ) [static]
```

Lerps between two angle values.

6.12.2.7 Max()

```
static Angle Max (
    Angle a,
    Angle b ) [static]
```

Returns the larger of two supplied angles.

6.12.2.8 Min()

```
static Angle Min (
    Angle a,
    Angle b ) [static]
```

Returns the smaller of two supplied angles.

6.12.2.9 operator Angle() [1/3]

```
static implicit operator Angle (
    double value ) [static]
```

Converts double to [Angle](#).

Parameters

<code>value</code>	Double value.
--------------------	---------------

Returns

[Angle](#) instance with the value of the double.

6.12.2.10 operator Angle() [2/3]

```
static implicit operator Angle (
    float value ) [static]
```

Converts float to [Angle](#).

Parameters

<code>value</code>	Float value.
--------------------	--------------

Returns

[Angle](#) instance with the value of the float.

6.12.2.11 operator Angle() [3/3]

```
static implicit operator Angle (
    int value ) [static]
```

Converts int to [Angle](#).

Parameters

<code>value</code>	Integer value.
--------------------	----------------

Returns

[Angle](#) instance with the value of the integer.

6.12.2.12 operator double()

```
static operator double (
    Angle value ) [explicit], [static]
```

Converts [Angle](#) to double.

Parameters

<code>value</code>	Angle instance.
--------------------	---------------------------------

Returns

Double representation of the [Angle](#).

6.12.2.13 operator float()

```
static operator float (
    Angle value ) [explicit], [static]
```

Converts [Angle](#) to float.

Parameters

<code>value</code>	<code>Angle</code> instance.
--------------------	------------------------------

Returns

Float representation of the `Angle`.

6.12.2.14 operator"!=()

```
static bool operator!= (
    Angle a,
    Angle b ) [static]
```

Inequality operator for `Angle` struct.

Parameters

<code>a</code>	First <code>Angle</code> instance.
<code>b</code>	Second <code>Angle</code> instance.

Returns

True if the value of the first `Angle` instance is not equal to the value of the second `Angle` instance, otherwise false.

6.12.2.15 operator+()

```
static Angle operator+ (
    Angle a,
    Angle b ) [static]
```

Addition operator for `Angle` struct.

Parameters

<code>a</code>	First <code>Angle</code> instance.
<code>b</code>	Second <code>Angle</code> instance.

Returns

A new `Angle` instance that is the sum of the first and second `Angle` instances, wrapped around at 360 degrees if necessary.

6.12.2.16 operator-()

```
static Angle operator- (
    Angle a,
    Angle b ) [static]
```

Subtraction operator for [Angle](#) struct.

Parameters

a	First Angle instance.
b	Second Angle instance.

Returns

A new [Angle](#) instance that is the difference of the first and second [Angle](#) instances, wrapped around at 360 degrees if necessary.

6.12.2.17 operator<()

```
static bool operator< (
    Angle a,
    Angle b ) [static]
```

Less than operator for [Angle](#) struct.

Parameters

a	First Angle instance.
b	Second Angle instance.

Returns

True if the value of the first [Angle](#) instance is less than the value of the second [Angle](#) instance, otherwise false.

6.12.2.18 operator<=()

```
static bool operator<= (
    Angle a,
    Angle b ) [static]
```

Less than or equal to operator for [Angle](#) struct.

Parameters

<i>a</i>	First Angle instance.
<i>b</i>	Second Angle instance.

Returns

True if the value of the first [Angle](#) instance is less than or equal to the value of the second [Angle](#) instance, otherwise false.

6.12.2.19 operator==()

```
static bool operator== (
    Angle a,
    Angle b ) [static]
```

Equality operator for [Angle](#) struct.

Parameters

<i>a</i>	First Angle instance.
<i>b</i>	Second Angle instance.

Returns

True if the value of the first [Angle](#) instance is equal to the value of the second [Angle](#) instance, otherwise false.

6.12.2.20 operator>()

```
static bool operator> (
    Angle a,
    Angle b ) [static]
```

Greater than operator for [Angle](#) struct.

Parameters

<i>a</i>	First Angle instance.
<i>b</i>	Second Angle instance.

Returns

True if the value of the first [Angle](#) instance is greater than the value of the second [Angle](#) instance, otherwise false.

6.12.2.21 operator>=()

```
static bool operator>= (
    Angle a,
    Angle b ) [static]
```

Greater than or equal to operator for [Angle](#) struct.

Parameters

a	First Angle instance.
b	Second Angle instance.

Returns

True if the value of the first [Angle](#) instance is greater than or equal to the value of the second [Angle](#) instance, otherwise false.

6.12.2.22 ToString()

```
override string ToString ( )
```

String representation of the [Angle](#).

6.12.3 Member Data Documentation

6.12.3.1 SIZE

```
const int SIZE = 4 [static]
```

Size of this struct in bytes.

6.13 ArrayLengthAttribute Class Reference

Editor attribute for selecting the minimum and maximum length constraints for an array field.

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [ArrayLengthAttribute \(int length\)](#)
Initializes a new instance of the `ArrayLengthAttribute` class with the specified length.
- [ArrayLengthAttribute \(int minLength, int maxLength\)](#)
Initializes a new instance of the `ArrayLengthAttribute` class with the specified minimum and maximum lengths.

Properties

- int [MaxLength \[get\]](#)
Gets the maximum length of the array.
- int [MinLength \[get\]](#)
Gets the minimum length of the array.

Additional Inherited Members

6.13.1 Detailed Description

Editor attribute for selecting the minimum and maximum length constraints for an array field.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 `ArrayLengthAttribute()` [1/2]

```
ArrayLengthAttribute (
    int length )
```

Initializes a new instance of the `ArrayLengthAttribute` class with the specified length.

Parameters

<code>length</code>	The length of the array.
---------------------	--------------------------

6.13.2.2 `ArrayLengthAttribute()` [2/2]

```
ArrayLengthAttribute (
    int minLength,
    int maxLength )
```

Initializes a new instance of the `ArrayLengthAttribute` class with the specified minimum and maximum lengths.

Parameters

<i>minLength</i>	The minimum length of the array.
<i>maxLength</i>	The maximum length of the array.

6.13.3 Property Documentation

6.13.3.1 MaxLength

```
int MaxLength [get]
```

Gets the maximum length of the array.

6.13.3.2 MinLength

```
int MinLength [get]
```

Gets the minimum length of the array.

6.14 AssemblyNameAttribute Class Reference

Specifies that the attributed field represents the name of an assembly.

Inherits [DrawerPropertyAttribute](#).

Properties

- bool [RequiresUnsafeCode](#) [get, set]
Gets or sets a value indicating whether the assembly requires unsafe code.

6.14.1 Detailed Description

Specifies that the attributed field represents the name of an assembly.

6.14.2 Property Documentation

6.14.2.1 RequiresUnsafeCode

```
bool RequiresUnsafeCode [get], [set]
```

Gets or sets a value indicating whether the assembly requires unsafe code.

6.15 AssetObject Class Reference

Base class for all [Fusion](#) assets.

Inherits [ScriptableObject](#).

6.15.1 Detailed Description

Base class for all [Fusion](#) assets.

6.16 TaskManager Class Reference

Task Factory is used to create new Tasks and Schedule long running Tasks

Static Public Member Functions

- static Task [ContinueWhenAll](#) (Task[] precedingTasks, Func< CancellationToken, Task > action, CancellationToken cancellationToken)
Run a continuation Task after all other Tasks have completed
- static Task [Run](#) (Func< CancellationToken, Task > action, CancellationToken cancellationToken, Task<-CreationOptions options=TaskCreationOptions.None)
Run an Action asynchronously
- static Task [Service](#) (Action recurringAction, CancellationToken cancellationToken, int interval, string serviceName=null)
Start a Service Task that will invoke a Recurring Action every each interval in millis
- static void [Setup](#) ()
Setup a new TaskFactory tailored to work with Unity

6.16.1 Detailed Description

Task Factory is used to create new Tasks and Schedule long running Tasks

6.16.2 Member Function Documentation

6.16.2.1 ContinueWhenAll()

```
static Task ContinueWhenAll (
    Task[] precedingTasks,
    Func< CancellationToken, Task > action,
    CancellationToken cancellationToken ) [static]
```

Run a continuation Task after all other Tasks have completed

Parameters

<i>precedingTasks</i>	List of pending tasks to wait
<i>action</i>	Action to run after the Tasks
<i>cancellationToken</i>	CancellationToken used to stop the Action

Returns[Async](#) Task based on the Action**6.16.2.2 Run()**

```
static Task Run (
    Func< CancellationToken, Task > action,
    CancellationToken cancellationToken,
    TaskCreationOptions options = TaskCreationOptions.None ) [static]
```

Run an Action asynchronously

Parameters

<i>action</i>	Action to be invoked
<i>cancellationToken</i>	CancellationToken used to stop the Action
<i>options</i>	Extra Task Creation options

Returns[Async](#) Task based on the Action**6.16.2.3 Service()**

```
static Task Service (
    Func< Task< bool >> recurringAction,
    CancellationToken cancellationToken,
    int interval,
    string serviceName = null ) [static]
```

Start a Service Task that will invoke a Recurring Action every each interval in millis

Parameters

<i>recurringAction</i>	Action invoked every interval. It can return false to stop the service
<i>cancellationToken</i>	CancellationToken used to stop the service
<i>interval</i>	Interval between action invoke
<i>serviceName</i>	Custom id name for the Service

Returns

Service Task

6.16.2.4 Setup()

```
static void Setup ( ) [static]
```

Setup a new TaskFactory tailored to work with Unity

6.17 AuthorityMasks Class Reference

Provides constants and methods for managing authority masks.

Static Public Attributes

- const int **ALL** = **STATE** | **INPUT** | **PROXY**
Constant representing all authorities.
- const int **INPUT** = 1 << 1
Constant representing the input authority mask.
- const int **NONE** = 0
Constant representing no authority.
- const int **PROXY** = 1 << 2
Constant representing the proxy authority mask.
- const int **STATE** = 1 << 0
Constant representing the state authority mask.

6.17.1 Detailed Description

Provides constants and methods for managing authority masks.

6.17.2 Member Data Documentation

6.17.2.1 ALL

```
const int ALL = STATE | INPUT | PROXY [static]
```

Constant representing all authorities.

6.17.2.2 INPUT

```
const int INPUT = 1 << 1 [static]
```

Constant representing the input authority mask.

6.17.2.3 NONE

```
const int NONE = 0 [static]
```

Constant representing no authority.

6.17.2.4 PROXY

```
const int PROXY = 1 << 2 [static]
```

Constant representing the proxy authority mask.

6.17.2.5 STATE

```
const int STATE = 1 << 0 [static]
```

Constant representing the state authority mask.

6.18 Behaviour Class Reference

Alternative base class to Unity's MonoBehaviour. This allows for components that work both in Unity, as well as the Photon relays.

Inherits MonoBehaviour, ILogSource, and ILogDumpable.

Inherited by [Hitbox](#), [NetworkEvents](#), [NetworkObject](#), [NetworkObjectPrefabData](#), [NetworkRunner](#), and [SimulationBehaviour](#).

Public Member Functions

- T [AddBehaviour< T >\(\)](#)
Wrapper for Unity's GameObject.AddComponent()
- T [GetBehaviour< T >\(\)](#)
Wrapper for Unity's GameObject.GetComponentInChildren()
- bool [TryGetBehaviour< T >\(out T behaviour\)](#)
Wrapper for Unity's GameObject.TryGetComponent()

Static Public Member Functions

- static void [DestroyBehaviour](#) ([Behaviour](#) behaviour)
Wrapper for Unity's GameObject.Destroy()

6.18.1 Detailed Description

Alternative base class to Unity's MonoBehaviour. This allows for components that work both in Unity, as well as the Photon relays.

6.18.2 Member Function Documentation

6.18.2.1 AddBehaviour< T >()

`T AddBehaviour< T > ()`

Wrapper for Unity's `GameObject.AddComponent()`

Type Constraints

T : Behaviour

6.18.2.2 DestroyBehaviour()

```
static void DestroyBehaviour (
    Behaviour behaviour ) [static]
```

Wrapper for Unity's `GameObject.Destroy()`

6.18.2.3 GetBehaviour< T >()

`T GetBehaviour< T > ()`

Wrapper for Unity's `GameObject.GetComponentInChildren()`

Type Constraints

T : Behaviour

6.18.2.4 TryGetBehaviour< T >()

```
bool TryGetBehaviour< T > (
    out T behaviour )
```

Wrapper for Unity's GameObject.TryGetComponent()

Type Constraints

T : Behaviour

6.19 BinaryDataAttribute Class Reference

Specifies that the field represents binary data.

Inherits [DrawerPropertyAttribute](#).

6.19.1 Detailed Description

Specifies that the field represents binary data.

6.20 BitSetAttribute Class Reference

Represents an attribute that specifies the number of bits in a bit set.

Inherits [DrawerPropertyAttribute](#).

Public Member Functions

- [BitSetAttribute](#) (int bitCount)

Initializes a new instance of the [BitSetAttribute](#) class with the specified number of bits.

Properties

- int [BitCount](#) [get]
Gets the number of bits in the bit set.

6.20.1 Detailed Description

Represents an attribute that specifies the number of bits in a bit set.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 BitSetAttribute()

```
BitSetAttribute (
    int bitCount )
```

Initializes a new instance of the [BitSetAttribute](#) class with the specified number of bits.

Parameters

<i>bitCount</i>	The number of bits in the bit set.
-----------------	------------------------------------

6.20.3 Property Documentation

6.20.3.1 BitCount

int BitCount [get]

Gets the number of bits in the bit set.

6.21 CapacityAttribute Class Reference

Capacity Attribute

Inherits Attribute.

Public Member Functions

- [CapacityAttribute \(int length\)](#)
CapacityAttribute Constructor

Properties

- int [Length \[get\]](#)
Total Capacity

6.21.1 Detailed Description

Capacity Attribute

6.21.2 Constructor & Destructor Documentation

6.21.2.1 CapacityAttribute()

```
CapacityAttribute (
    int length )
```

[CapacityAttribute Constructor](#)

Parameters

<i>length</i>	Length
---------------	------------------------

6.21.3 Property Documentation

6.21.3.1 Length

```
int Length [get]
```

Total Capacity

6.22 DecoratingPropertyAttribute Class Reference

A base class for property attributes that decorate other property attributes.

Inherits [PropertyAttribute](#).

Inherited by [ArrayLengthAttribute](#), [DisplayNameAttribute](#), [DolfAttributeBase](#), [FieldEditorButtonAttribute](#), [HideArrayElementLabelAttribute](#), [InlineHelpAttribute](#), [ReadOnlyAttribute](#), [SerializeReferenceTypePickerAttribute](#), and [UnitAttribute](#).

Static Public Attributes

- const int [DefaultOrder](#) = -10000

The default order of the attribute.

Protected Member Functions

- [DecoratingPropertyAttribute \(\)](#)
Initializes a new instance with the default order.
- [DecoratingPropertyAttribute \(int order\)](#)
Initializes a new instance with the specified order.

6.22.1 Detailed Description

A base class for property attributes that decorate other property attributes.

6.22.2 Constructor & Destructor Documentation

6.22.2.1 DecoratingPropertyAttribute() [1/2]

```
DecoratingPropertyAttribute ( ) [protected]
```

Initializes a new instance with the default order.

6.22.2.2 DecoratingPropertyAttribute() [2/2]

```
DecoratingPropertyAttribute ( int order ) [protected]
```

Initializes a new instance with the specified order.

Parameters

order	
-------	--

6.22.3 Member Data Documentation

6.22.3.1 DefaultOrder

```
const int DefaultOrder = -10000 [static]
```

The default order of the attribute.

6.23 DefaultForPropertyAttribute Class Reference

Default For Property Attribute

Inherits PropertyAttribute.

Public Member Functions

- [DefaultForPropertyAttribute](#) (string propertyName, int wordOffset, int wordCount)
DefaultForPropertyAttribute Constructor

Properties

- String [PropertyName](#) [get]
Property Name
- int [WordCount](#) [get]
Property Word Count
- int [WordOffset](#) [get]
Property Word Offset

6.23.1 Detailed Description

Default For Property Attribute

For non-serialized properties

6.23.2 Constructor & Destructor Documentation

6.23.2.1 DefaultForPropertyAttribute()

```
DefaultForPropertyAttribute (
    string propertyName,
    int wordOffset,
    int wordCount )
```

[DefaultForPropertyAttribute](#) Constructor

Parameters

<i>propertyName</i>	PropertyName
<i>wordOffset</i>	WordOffset
<i>wordCount</i>	WordCount

6.23.3 Property Documentation

6.23.3.1 PropertyName

```
String PropertyName [get]
```

Property Name

6.23.3.2 WordCount

```
int WordCount [get]
```

Property Word Count

6.23.3.3 WordOffset

```
int WordOffset [get]
```

Property Word Offset

6.24 DisplayAsEnumAttribute Class Reference

Casts an enum or int value in the inspector to specific enum type for rendering of its popup list. Supplying a method name rather than a type allows a property with the type Type to be used to dynamically get the enum type.

Inherits [DrawerPropertyAttribute](#).

Public Member Functions

- [DisplayAsEnumAttribute](#) (string enumTypeMemberName)
Initializes a new instance of the [DisplayAsEnumAttribute](#) class with the specified enum type member name.
- [DisplayAsEnumAttribute](#) (Type enumType)
Initializes a new instance of the [DisplayAsEnumAttribute](#) class with the specified enum type.

Properties

- Type [EnumType](#) [get]
Gets the type of the enum.
- string [EnumTypeMemberName](#) [get]
Gets the name of the member that returns the enum type.

6.24.1 Detailed Description

Casts an enum or int value in the inspector to specific enum type for rendering of its popup list. Supplying a method name rather than a type allows a property with the type Type to be used to dynamically get the enum type.

6.24.2 Constructor & Destructor Documentation

6.24.2.1 [DisplayAsEnumAttribute\(\)](#) [1/2]

```
DisplayAsEnumAttribute (
    Type enumType )
```

Initializes a new instance of the [DisplayAsEnumAttribute](#) class with the specified enum type.

Parameters

<i>enumType</i>	The type of the enum.
-----------------	-----------------------

6.24.2.2 DisplayAsEnumAttribute() [2/2]

```
DisplayAsEnumAttribute (
    string enumTypeMemberName )
```

Initializes a new instance of the [DisplayAsEnumAttribute](#) class with the specified enum type member name.

Parameters

<i>enumTypeMemberName</i>	The name of the member that returns the enum type.
---------------------------	--

6.24.3 Property Documentation**6.24.3.1 EnumType**

Type `EnumType` [get]

Gets the type of the enum.

6.24.3.2 EnumTypeMemberName

`string EnumTypeMemberName [get]`

Gets the name of the member that returns the enum type.

6.25 DisplayNameAttribute Class Reference

Specifies the display name for a field.

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [DisplayNameAttribute](#) (string name)

Initializes a new instance of the [DisplayNameAttribute](#) class with the specified name.

Public Attributes

- readonly string [Name](#)

Field name to display.

Additional Inherited Members

6.25.1 Detailed Description

Specifies the display name for a field.

6.25.2 Constructor & Destructor Documentation

6.25.2.1 DisplayNameAttribute()

```
DisplayNameAttribute (
    string name )
```

Initializes a new instance of the [DisplayNameAttribute](#) class with the specified name.

Parameters

<i>name</i>	The display name.
-------------	-------------------

6.25.3 Member Data Documentation

6.25.3.1 Name

```
readonly string Name
```

Field name to display.

6.26 DolfAttributeBase Class Reference

Editor attribute for selective editor rendering. Condition member can be a property, field or method (with a return value).

Inherits [DecoratingPropertyAttribute](#).

Inherited by [DrawIfAttribute](#), [ErrorIfAttribute](#), and [WarnIfAttribute](#).

Public Attributes

- double `_doubleValue`
The double value to compare against.
- bool `_isDouble`
Is the value to compare against a double?
- long `_longValue`
The long value to compare against.
- [CompareOperator Compare](#)
The comparison operator to use.
- string `ConditionMember`
Condition member to evaluate.
- bool `ErrorOnConditionMemberNotFound` = true
If true, an error will be thrown if the condition member is not found.

Protected Member Functions

- [DoIfAttributeBase \(string conditionMember, bool compareToValue, CompareOperator compare\)](#)
Initializes a new instance with a boolean value to compare against.
- [DoIfAttributeBase \(string conditionMember, double compareToValue, CompareOperator compare\)](#)
Initializes a new instance with a double value to compare against.
- [DoIfAttributeBase \(string conditionMember, long compareToValue, CompareOperator compare\)](#)
Initializes a new instance with a long value to compare against.

Additional Inherited Members

6.26.1 Detailed Description

Editor attribute for selective editor rendering. Condition member can be a property, field or method (with a return value).

Value of condition method is converted to a long or a double. null = 0, false = 0, true = 1, Unity Object = InstanceId

6.26.2 Constructor & Destructor Documentation

6.26.2.1 DoIfAttributeBase() [1/3]

```
DoIfAttributeBase (
    string conditionMember,
    double compareToValue,
    CompareOperator compare ) [protected]
```

Initializes a new instance with a double value to compare against.

Parameters

<i>conditionMember</i>	
<i>compareToValue</i>	
<i>compare</i>	

6.26.2.2 DolfAttributeBase() [2/3]

```
DoIfAttributeBase (
    string conditionMember,
    long compareToValue,
    CompareOperator compare ) [protected]
```

Initializes a new instance with a long value to compare against.

Parameters

<i>conditionMember</i>	
<i>compareToValue</i>	
<i>compare</i>	

6.26.2.3 DolfAttributeBase() [3/3]

```
DoIfAttributeBase (
    string conditionMember,
    bool compareToValue,
    CompareOperator compare ) [protected]
```

Initializes a new instance with a boolean value to compare against.

Parameters

<i>conditionMember</i>	
<i>compareToValue</i>	
<i>compare</i>	

6.26.3 Member Data Documentation**6.26.3.1 _doubleValue**

```
double _doubleValue
```

The double value to compare against.

6.26.3.2 `_isDouble`

```
bool _isDouble
```

Is the value to compare against a double?

6.26.3.3 `_longValue`

```
long _longValue
```

The long value to compare against.

6.26.3.4 `Compare`

```
CompareOperator Compare
```

The comparison operator to use.

6.26.3.5 `ConditionMember`

```
string ConditionMember
```

Condition member to evaluate.

6.26.3.6 `ErrorOnConditionMemberNotFound`

```
bool ErrorOnConditionMemberNotFound = true
```

If true, an error will be thrown if the condition member is not found.

6.27 DrawerPropertyAttribute Class Reference

A base class for property attributes that are used to draw properties in the inspector.

Inherits PropertyAttribute.

Inherited by [AssemblyNameAttribute](#), [BinaryDataAttribute](#), [BitSetAttribute](#), [DisplayAsEnumAttribute](#), [DrawInlineAttribute](#), [ExpandableEnumAttribute](#), [LayerAttribute](#), [LayerMatrixAttribute](#), [MaxStringByteCountAttribute](#), [RangeExAttribute](#), [ScenePathAttribute](#), [ToggleLeftAttribute](#), [UnityAssetGuidAttribute](#), and [UnityResourcePathAttribute](#).

6.27.1 Detailed Description

A base class for property attributes that are used to draw properties in the inspector.

6.28 DrawIfAttribute Class Reference

Editor attribute for selectively drawing/hiding fields. Condition member can be a property, field or method (with a return value).

Inherits [DolfAttributeBase](#).

Public Member Functions

- [DrawIfAttribute](#) (string conditionMember)
Initializes a new instance that will hide the field if the condition member is not equal to zero.
- [DrawIfAttribute](#) (string conditionMember, bool compareToValue, [CompareOperator](#) compare=[CompareOperator.Equal](#), [DrawIfMode](#) mode=[DrawIfMode.ReadOnly](#))
- [DrawIfAttribute](#) (string conditionMember, double compareToValue, [CompareOperator](#) compare=[CompareOperator.Equal](#), [DrawIfMode](#) mode=[DrawIfMode.ReadOnly](#))
- [DrawIfAttribute](#) (string conditionMember, long compareToValue, [CompareOperator](#) compare=[CompareOperator.Equal](#), [DrawIfMode](#) mode=[DrawIfMode.ReadOnly](#))

Public Attributes

- [DrawIfMode Mode](#)
Instructs the attribute completely hide the field if not draw, rather than the default of just disabling it.

Properties

- bool? [Hide](#) [get, set]
Should the field be hidden if the condition is not met?

Additional Inherited Members

6.28.1 Detailed Description

Editor attribute for selectively drawing/hiding fields. Condition member can be a property, field or method (with a return value).

Value of condition method is converted to a long. null = 0, false = 0, true = 1, Unity Object = InstanceId

6.28.2 Constructor & Destructor Documentation

6.28.2.1 DrawIfAttribute()

```
DrawIfAttribute (
    string conditionMember )
```

Initializes a new instance that will hide the field if the condition member is not equal to zero.

Parameters

<code>conditionMember</code>	<input type="checkbox"/>
------------------------------	--------------------------

6.28.3 Member Data Documentation

6.28.3.1 Mode

[DrawIfMode](#) Mode

Instructs the attribute completely hide the field if not draw, rather than the default of just disabling it.

6.28.4 Property Documentation

6.28.4.1 Hide

`bool? Hide [get], [set]`

Should the field be hidden if the condition is not met?

6.29 DrawInlineAttribute Class Reference

Specifies that a field should be drawn inline in the inspector.

Inherits [DrawerPropertyAttribute](#).

6.29.1 Detailed Description

Specifies that a field should be drawn inline in the inspector.

6.30 DynamicHeap Struct Reference

A dynamic heap for allocating and tracking unmanaged objects.

Classes

- class [Ignore](#)

Ignore this field when scanning for pointers.

Public Types

- enum class **ObjectFlags** : byte

Public Member Functions

- delegate void **CollectGarbageDelegate** (DynamicHeap *heap, void **dynamicRoots, int dynamicRootsLength)

Collect garbage delegate

Static Public Member Functions

- static void **AllocateBlock** (DynamicHeap *heap)
- static byte * **AllocateInternal** (DynamicHeap *heap, int size, out byte block)
- static Page * **AllocatePage** (DynamicHeap *heap)
- static Page * **AllocatePage_Internal** (DynamicHeap *heap, bool mustSucceed)
- static int **BitScan** (uint v)
- static int **BlocksWithAvailablePages** (DynamicHeap *heap)
- static void **CollectGarbage** (DynamicHeap *heap, void **dynamicRoots, int dynamicRootsLength)

Collect garbage

- static void **Destroy** (Block *block)
- static void **ExpandStack** (DynamicHeap *heap)
- static void **Free** (DynamicHeap *heap, void *ptr)

Free up an object

- static void **FreeInternal** (DynamicHeap *heap, void *ptr, Object objData)
- static int **GetBin** (int size)
- static Bin * **GetBinByIndex** (DynamicHeap *heap, int binIndex)
- static int **GetBinIndexForSize** (DynamicHeap *heap, int size)
- static Page * **GetPageForPtr** (DynamicHeap *heap, Block *block, void *ptr)
- static int **GetPageOffset** (Page *page, ObjectFree *obj)
- static ushort **GetTypeOffset**< T > ()
- static void **InitObj** (DynamicHeap *heap, Object *obj, ushort type, ushort array, byte block)
- static void **InitRoot** (DynamicHeap *heap, Object *obj)
- static bool **IsPtrInBlock** (DynamicHeap *heap, Block *block, void *p)
- static ushort **NextGen** (DynamicHeap *heap)
- static int **ObjectsFreeCount** (Page *p)
- static int **PagesWithAvailableObjectsInBin** (Bin *bin)
- static ObjectFree * **ResolvePageOffset** (Page *page, int offset)
- static T * **SetForcedAlive**< T > (T *ptr)

Mark an object with ObjectFlags.ForceAlive

- static void **ThrowHeapCorrupted** ()
- static byte * **TryAllocateFromPage** (DynamicHeap *heap, Page *page, int size, out byte block)
- static int **WordCount** (int size)

Public Attributes

- `Bin * _bins`
- `Block ** _blocks`
- `BlockList _blocksFreePages`
- `int _blocksUsed`
- `Config _config`
- `int _gcBlock`
- `int _gcBlockPage`
- `ushort _gcGen`
- `Phase _gcPhase`
- `Object ** _gcStack`
- `int _gcStackCapacity`
- `int _gcStackCount`
- `int _memoryAllocated`
- `int _objectsAllocated`
- `Object ** _rootList`
- `int _rootListCapacity`
- `int _rootListCount`
- `int * _typeMap`
- `int _typeMapLength`
- `int * _typeMapStrides`

Static Public Attributes

- `static byte[] _debruijnTable`
- `static Dictionary< Type, TypeData > _types = null`
- `static Dictionary< ushort, TypeData > _typesByOffset = null`

6.30.1 Detailed Description

A dynamic heap for allocating and tracking unmanaged objects.

6.30.2 Member Function Documentation

6.30.2.1 CollectGarbage()

```
static void CollectGarbage (
    DynamicHeap * heap,
    void ** dynamicRoots,
    int dynamicRootsLength ) [static]
```

Collect garbage

Parameters

<code>heap</code>	Dynamic heap to collect from
<code>dynamicRoots</code>	Dynamic roots
<code>dynamicRootsLength</code>	Dynamic roots length

6.30.2.2 CollectGarbageDelegate()

```
delegate void CollectGarbageDelegate (
    DynamicHeap * heap,
    void ** dynamicRoots,
    int dynamicRootsLength )
```

Collect garbage delegate

6.30.2.3 Free()

```
static void Free (
    DynamicHeap * heap,
    void * ptr ) [static]
```

Free up an object

Parameters

<i>heap</i>	Heap to free from
<i>ptr</i>	Pointer to object

Exceptions

<i>InvalidOperationException</i>	Thrown if <i>ptr</i> is not a tracked object
----------------------------------	--

6.30.2.4 SetForcedAlive< T >()

```
static T* SetForcedAlive< T > (
    T * ptr ) [static]
```

Mark an object with ObjectFlags.ForceAlive

Parameters

<i>ptr</i>	Pointer Object to mark
------------	------------------------

Template Parameters

<i>T</i>	Type of object
----------	----------------

Returns

Pointer to object

Type Constraints

T : unmanaged

6.30.3 Member Data Documentation

6.30.3.1 _debruijnTable

byte [] _debruijnTable [static]

Initial value:

```
= {
    0, 9, 1, 10, 13, 21, 2, 29, 11, 14, 16, 18, 22, 25, 3, 30,
    8, 12, 20, 28, 15, 17, 24, 7, 19, 27, 23, 6, 26, 5, 4, 31
}
```

6.31 DynamicHeap.Ignore Class Reference

[Ignore](#) this field when scanning for pointers.

Inherits Attribute.

6.31.1 Detailed Description

[Ignore](#) this field when scanning for pointers.

6.32 DynamicHeapInstance Class Reference

Dynamic heap instance.

Public Member Functions

- void * [Allocate](#) (int size)
Allocate a pointer.
- void * [AllocateArray< T >](#) (int length)
Allocate a pointer array.
- void * [AllocateArrayPointers< T >](#) (int length)
Allocate an array of pointers.
- void * [AllocateTracked< T >](#) (bool root=false)
Allocate a tracked pointer.
- void * [AllocateTrackedArray< T >](#) (int length, bool root=false)
Allocate a tracked pointer array.
- void * [AllocateTrackedArrayPointers< T >](#) (int length, bool root=false)
Allocate a tracked array of pointers.
- [DynamicHeapInstance](#) (params Type[] types)
Create a dynamic heap instance.
- void [Free](#) (void *ptr)
Free a pointer.

6.32.1 Detailed Description

Dynamic heap instance.

6.32.2 Constructor & Destructor Documentation

6.32.2.1 DynamicHeapInstance()

```
DynamicHeapInstance (
    params Type[ ] types )
```

Create a dynamic heap instance.

Parameters

<i>types</i>	Types to allocate.
--------------	--------------------

6.32.3 Member Function Documentation

6.32.3.1 Allocate()

```
void* Allocate (
    int size )
```

Allocate a pointer.

Parameters

<i>size</i>	Size to allocate.
-------------	-------------------

Returns

Pointer to allocated memory.

6.32.3.2 AllocateArray< T >()

```
void* AllocateArray< T > (
    int length )
```

Allocate a pointer array.

Parameters

<i>length</i>	Length of array.
---------------	------------------

Template Parameters

<i>T</i>	Type of array.
----------	----------------

Returns

Pointer to allocated memory.

Type Constraints

T : *unmanaged*

6.32.3.3 AllocateArrayPointers< T >()

```
void* AllocateArrayPointers< T > (
    int length )
```

Allocate an array of pointers.

Parameters

<i>length</i>	Length of array.
---------------	------------------

Template Parameters

<i>T</i>	Type of array.
----------	----------------

Returns

Pointer to allocated memory.

Type Constraints

T : *unmanaged*

6.32.3.4 AllocateTracked< T >()

```
void* AllocateTracked< T > (
    bool root = false )
```

Allocate a tracked pointer.

Parameters

<i>root</i>	Signal if the pointer is a root.
-------------	----------------------------------

Template Parameters

<i>T</i>	Type of pointer.
----------	------------------

Returns

Pointer to allocated memory.

Type Constraints

T : *unmanaged*

6.32.3.5 AllocateTrackedArray< T >()

```
void* AllocateTrackedArray< T > (
    int length,
    bool root = false )
```

Allocate a tracked pointer array.

Parameters

<i>length</i>	Length of array.
<i>root</i>	Signal if the pointer is a root.

Template Parameters

<i>T</i>	Type of array.
----------	----------------

Returns

Pointer to allocated memory.

Type Constraints

T : *unmanaged*

6.32.3.6 AllocateTrackedArrayPointers< T >()

```
void* AllocateTrackedArrayPointers< T > (
    int length,
    bool root = false )
```

Allocate a tracked array of pointers.

Parameters

<i>length</i>	Length of array.
<i>root</i>	Signal if the pointer is a root.

Template Parameters

<i>T</i>	Type of array.
----------	----------------

Returns

Pointer to allocated memory.

Type Constraints

T : **unmanaged**

6.32.3.7 Free()

```
void Free (
    void * ptr )
```

Free a pointer.

Parameters

<i>ptr</i>	Pointer to free.
------------	------------------

6.33 EditorButtonAttribute Class Reference

Specifies that a method should be displayed as a button in the Unity editor.

Inherits Attribute.

Public Member Functions

- [EditorButtonAttribute \(EditorButtonVisibility visibility=EditorButtonVisibility.Always, int priority=0, bool dirtyObject=false\)](#)

Initializes a new instance of the [EditorButtonAttribute](#) class with the specified visibility, priority, and dirty object flag.
- [EditorButtonAttribute \(string label, EditorButtonVisibility visibility=EditorButtonVisibility.Always, int priority=0, bool dirtyObject=false\)](#)

Initializes a new instance of the [EditorButtonAttribute](#) class with the specified label, visibility, priority, and dirty object flag.

Public Attributes

- bool [AllowMultipleTargets](#)
Determines whether multiple targets are supported.
- bool [DirtyObject](#)
Determines whether the object should be marked as dirty after clicking the button.
- string [Label](#)
The label text to display on the button.
- int [Priority](#)
The priority of the button. Buttons with higher priority are displayed first.
- [EditorButtonVisibility Visibility](#)
The visibility of the button in the Unity editor.

6.33.1 Detailed Description

Specifies that a method should be displayed as a button in the Unity editor.

6.33.2 Constructor & Destructor Documentation

6.33.2.1 EditorButtonAttribute() [1/2]

```
EditorButtonAttribute (
    string label,
    EditorButtonVisibility visibility = EditorButtonVisibility.Always,
    int priority = 0,
    bool dirtyObject = false )
```

Initializes a new instance of the [EditorButtonAttribute](#) class with the specified label, visibility, priority, and dirty object flag.

Parameters

<i>label</i>	The label text to display on the button.
<i>visibility</i>	The visibility of the button in the Unity editor.
<i>priority</i>	The priority of the button. Buttons with higher priority are displayed first.
<i>dirtyObject</i>	Determines whether the object should be marked as dirty after clicking the button.

6.33.2.2 EditorButtonAttribute() [2/2]

```
EditorButtonAttribute (
    EditorButtonVisibility visibility = EditorButtonVisibility.Always,
    int priority = 0,
    bool dirtyObject = false )
```

Initializes a new instance of the [EditorButtonAttribute](#) class with the specified visibility, priority, and dirty object flag.

Parameters

<code>visibility</code>	The visibility of the button in the Unity editor.
<code>priority</code>	The priority of the button. Buttons with higher priority are displayed first.
<code>dirtyObject</code>	Determines whether the object should be marked as dirty after clicking the button.

6.33.3 Member Data Documentation

6.33.3.1 AllowMultipleTargets

```
bool AllowMultipleTargets
```

Determines whether multiple targets are supported.

6.33.3.2 DirtyObject

```
bool DirtyObject
```

Determines whether the object should be marked as dirty after clicking the button.

6.33.3.3 Label

```
string Label
```

The label text to display on the button.

6.33.3.4 Priority

```
int Priority
```

The priority of the button. Buttons with higher priority are displayed first.

6.33.3.5 Visibility

```
EditorButtonVisibility Visibility
```

The visibility of the button in the Unity editor.

6.34 DataEncryptor Class Reference

Responsible for encrypting and decrypting data buffers

Inherits [IDataEncryption](#).

Public Member Functions

- bool [ComputeHash](#) (byte *buffer, ref int bufferLength, int capacity)
Compute the Buffer hash and append it to the buffer itself
- bool [DecryptData](#) (byte *buffer, ref int bufferLength, int capacity)
Decrypt data in place and update it's length.
- void [Dispose](#) ()
- bool [EncryptData](#) (byte *buffer, ref int bufferLength, int capacity)
Encrypts the data in the provided buffer.
- byte[] [GenerateKey](#) ()
Generate the key used used by the encryption implementation
- void [Setup](#) (byte[] key)
Setup the encryption implementation with the right key
- bool [VerifyHash](#) (byte *buffer, ref int bufferLength, int capacity)
Verify the buffer hash that was appended to the buffer

6.34.1 Detailed Description

Responsible for encrypting and decrypting data buffers

6.34.2 Member Function Documentation

6.34.2.1 EncryptData()

```
bool EncryptData (
    byte * buffer,
    ref int bufferLength,
    int capacity )
```

Encrypts the data in the provided buffer.

Parameters

<i>buffer</i>	The buffer containing the data to be encrypted.
<i>bufferLength</i>	The length of the data in the buffer.
<i>capacity</i>	The total capacity of the buffer.

Returns

Returns true if the encryption was successful, false otherwise.

Exceptions

<i>InvalidOperationException</i>	Thrown when the encryption provider is not initialized.
<i>ArgumentException</i>	Thrown when the original buffer cannot hold the encrypted data.

Implements [IDataEncryption](#).

6.35 IDataEncryption Interface Reference

Interface for classes that manage the encryption/decryption of byte arrays

Inherits [IDisposable](#).

Inherited by [DataEncryptor](#).

Public Member Functions

- bool [ComputeHash](#) (byte *buffer, ref int bufferLength, int capacity)
Compute the Buffer hash and append it to the buffer itself
- bool [DecryptData](#) (byte *buffer, ref int bufferLength, int capacity)
Decrypt data in place and update it's length.
- bool [EncryptData](#) (byte *buffer, ref int bufferLength, int capacity)
Encrypt data in place and update it's length.
- byte[] [GenerateKey](#) ()
Generate the key used used by the encryption implementation
- void [Setup](#) (byte[] key)
Setup the encryption implementation with the right key
- bool [VerifyHash](#) (byte *buffer, ref int bufferLength, int capacity)
Verify the buffer hash that was appended to the buffer

6.35.1 Detailed Description

Interface for classes that manage the encryption/decryption of byte arrays

6.35.2 Member Function Documentation

6.35.2.1 ComputeHash()

```
bool ComputeHash (
    byte * buffer,
    ref int bufferLength,
    int capacity )
```

Compute the Buffer hash and append it to the buffer itself

Parameters

<i>buffer</i>	Data to compute the hash
<i>bufferLength</i>	Length of the data to hash
<i>capacity</i>	Buffer total capacity

Returns

True if the hash was properly computed, false otherwise

Implemented in [DataEncryptor](#).

6.35.2.2 DecryptData()

```
bool DecryptData (
    byte * buffer,
    ref int bufferLength,
    int capacity )
```

Decrypt data in place and update it's length.

Parameters

<i>buffer</i>	Data to decrypt
<i>bufferLength</i>	Length of the data to decrypt
<i>capacity</i>	Buffer total capacity

Returns

True if the decryption was completed, false otherwise

Implemented in [DataEncryptor](#).

6.35.2.3 EncryptData()

```
bool EncryptData (
    byte * buffer,
    ref int bufferLength,
    int capacity )
```

Encrypt data in place and update it's length.

Parameters

<i>buffer</i>	Data to encrypt
<i>bufferLength</i>	Length of the data to encrypt
<i>capacity</i>	Buffer total capacity

Returns

True if the encryption was completed, false otherwise

Implemented in [DataEncryptor](#).

6.35.2.4 GenerateKey()

```
byte [ ] GenerateKey ( )
```

Generate the key used used by the encryption implementation

Returns

Key used to setup the encryption implementation

Implemented in [DataEncryptor](#).

6.35.2.5 Setup()

```
void Setup (
    byte[ ] key )
```

Setup the encryption implementation with the right key

Implemented in [DataEncryptor](#).

6.35.2.6 VerifyHash()

```
bool VerifyHash (
    byte * buffer,
    ref int bufferLength,
    int capacity )
```

Verify the buffer hash that was appended to the buffer

Parameters

<i>buffer</i>	Buffer to check the hash
<i>bufferLength</i>	Length of the data to hash
<i>capacity</i>	Buffer total capacity

Returns

True if the hash was properly verified, false otherwise

Implemented in [DataEncryptor](#).

6.36 ErrorIfAttribute Class Reference

Editor attribute for adding notices to fields if the condition member evaluates as `true`. Condition member can be a property, field or method (with a return value).

Inherits [DolfAttributeBase](#).

Public Member Functions

- [ErrorIfAttribute \(string conditionMember, bool compareToValue, string message, CompareOperator compare=CompareOperator.Equal\)](#)
- [ErrorIfAttribute \(string conditionMember, double compareToValue, string message, CompareOperator compare=CompareOperator.Equal\)](#)
- [ErrorIfAttribute \(string conditionMember, long compareToValue, string message, CompareOperator compare=CompareOperator.Equal\)](#)

Public Attributes

- bool [AsBox](#)
Should the error be shown as a box?
- string [Message](#)
The default error text, when an error is shown.

Additional Inherited Members

6.36.1 Detailed Description

Editor attribute for adding notices to fields if the condition member evaluates as `true`. Condition member can be a property, field or method (with a return value).

6.36.2 Member Data Documentation

6.36.2.1 AsBox

bool [AsBox](#)

Should the error be shown as a box?

6.36.2.2 Message

```
string Message
```

The default error text, when an error is shown.

6.37 ExpandableEnumAttribute Class Reference

Editor attribute that shows an enum as an expandable list of options in the inspector.

Inherits [DrawerPropertyAttribute](#).

Properties

- bool [AlwaysExpanded](#) = false [get, set]
Always expand the enum in the inspector (no foldout)
- bool [ShowFlagsButtons](#) = true [get, set]
Show the enum flags as buttons in the inspector.
- bool [ShowInlineHelp](#) = false [get, set]
Show inline help for enum values in the inspector.

6.37.1 Detailed Description

Editor attribute that shows an enum as an expandable list of options in the inspector.

6.37.2 Property Documentation

6.37.2.1 AlwaysExpanded

```
bool AlwaysExpanded = false [get], [set]
```

Always expand the enum in the inspector (no foldout)

6.37.2.2 ShowFlagsButtons

```
bool ShowFlagsButtons = true [get], [set]
```

Show the enum flags as buttons in the inspector.

6.37.2.3 ShowInlineHelp

```
bool ShowInlineHelp = false [get], [set]
```

Show inline help for enum values in the inspector.

6.38 FieldEditorButtonAttribute Class Reference

Editor attribute to add a button that invokes a custom method in the inspector.

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [FieldEditorButtonAttribute](#) (string label, string targetMethod)

Initializes a new instance class with the specified label and target method.

Public Attributes

- bool [AllowMultipleTargets](#)

Is it allowed to select multiple targets for the button?

- string [Label](#)

Button label.

- string [TargetMethod](#)

The method to invoke when the button is clicked.

Additional Inherited Members

6.38.1 Detailed Description

Editor attribute to add a button that invokes a custom method in the inspector.

6.38.2 Constructor & Destructor Documentation

6.38.2.1 FieldEditorButtonAttribute()

```
FieldEditorButtonAttribute (
    string label,
    string targetMethod )
```

Initializes a new instance class with the specified label and target method.

Parameters

<i>label</i>	The label of the button
<i>targetMethod</i>	The method to invoke when the button is clicked

6.38.3 Member Data Documentation**6.38.3.1 AllowMultipleTargets**

```
bool AllowMultipleTargets
```

Is it allowed to select multiple targets for the button?

6.38.3.2 Label

```
string Label
```

Button label.

6.38.3.3 TargetMethod

```
string TargetMethod
```

The method to invoke when the button is clicked.

6.39 FieldsMask< T > Class Template Reference

Base class for [FieldsMask<T>](#).

Inherits [FieldsMask](#).

Public Member Functions

- [**FieldsMask \(\)**](#)
Constructor for [FieldsMask<T>](#).
- [**FieldsMask \(Func< Mask256 > getDefaultsDelegate\)**](#)
Constructor for [FieldsMask](#).
- [**FieldsMask \(long maskA, long maskB=0, long maskC=0, long maskD=0\)**](#)
Constructor for [FieldsMask](#).
- [**FieldsMask \(Mask256 mask\)**](#)
Constructor for [FieldsMask<T>](#).

Static Public Member Functions

- static implicit [operator Mask256 \(FieldsMask mask\)](#)
Implicitly convert [FieldsMask](#) to its long mask value.

Public Attributes

- [Mask256 Mask](#)
The internal mask value.

Protected Member Functions

- [FieldsMask \(\)](#)
Constructor for [FieldsMask](#).
- [FieldsMask \(long a, long b, long c, long d\)](#)
Constructor for [FieldsMask](#).
- [FieldsMask \(Mask256 mask\)](#)
Constructor for [FieldsMask](#).

6.39.1 Detailed Description

Base class for [FieldsMask<T>](#).

Associates and displays a 64 bit mask which represents the field members of a struct. Makes it possible to treat a Struct like an Flags Enum. NOTE: A [FieldsMask<T>](#) attribute is required for proper rendering in the Inspector.

6.39.2 Constructor & Destructor Documentation

6.39.2.1 FieldsMask() [1/7]

```
FieldsMask (
    Mask256 mask )  [protected]
```

Constructor for [FieldsMask](#).

6.39.2.2 FieldsMask() [2/7]

```
FieldsMask (
    long a,
    long b,
    long c,
    long d )  [protected]
```

Constructor for [FieldsMask](#).

6.39.2.3 FieldsMask() [3/7]

```
FieldsMask ( ) [protected]
```

Constructor for [FieldsMask](#).

6.39.2.4 FieldsMask() [4/7]

```
FieldsMask ( Mask256 mask )
```

Constructor for [FieldsMask<T>](#).

6.39.2.5 FieldsMask() [5/7]

```
FieldsMask ( long maskA,  
            long maskB = 0,  
            long maskC = 0,  
            long maskD = 0 )
```

Constructor for [FieldsMask](#).

6.39.2.6 FieldsMask() [6/7]

```
FieldsMask ( )
```

Constructor for [FieldsMask<T>](#).

6.39.2.7 FieldsMask() [7/7]

```
FieldsMask ( Func< Mask256 > getDefaultsDelegate )
```

Constructor for [FieldsMask](#).

6.39.3 Member Function Documentation

6.39.3.1 operator Mask256()

```
static implicit operator Mask256 (
    FieldsMask< T > mask ) [static]
```

Implicitly convert [FieldsMask](#) to its long mask value.

6.39.4 Member Data Documentation

6.39.4.1 Mask

[Mask256](#) Mask

The internal mask value.

6.40 FixedArray< T > Class Template Reference

A fixed size array that can be used in structs.

Inherits [IEnumerable< T >](#).

Classes

- struct [Enumerator](#)

Enumerator for the [FixedArray](#) struct.

Public Member Functions

- void [Clear \(\)](#)

Sets all elements in the array to their default value.
- void [CopyFrom \(List< T > source, int sourceOffset, int sourceCount\)](#)

Copies a range of elements from a source list into the FixedArray.
- void [CopyFrom \(T\[\] source, int sourceOffset, int sourceCount\)](#)

Copies a range of elements from a source array into the FixedArray.
- void [CopyTo \(List< T > list\)](#)

Adds each value to the supplied List. This does not clear the list, so values will be appended to the existing list.
- void [CopyTo \(T\[\] array, bool throwIfOverflow=true\)](#)

Copies values to the supplied array.
- [FixedArray \(T *array, int length\)](#)

NetworkArray constructor.
- [Enumerator GetEnumerator \(\)](#)

Returns an enumerator that iterates through the FixedArray.
- [IEnumerator< T > IEnumerable< T >. GetEnumerator \(\)](#)
- [IEnumerator IEnumerable. GetEnumerator \(\)](#)
- [T\[\] ToArray \(\)](#)

Allocates a new array and copies values from this array. For a non-alloc alternative use [CopyTo\(List< T >\)](#).
- string [ToString \(\)](#)

Returns the elements of this array as a string, with value separated by characters. Specifically for use in the Unity inspector. This is private and only is found by NetworkBehaviourEditor using reflection, so do not rename this method.
- override string [ToString \(\)](#)

Returns a string that represents the current object.

Static Public Member Functions

- static unsafe `FixedArray< T > Create< T >` (ref T firstField, int length)
Creates a `FixedArray` with a specified length.
- static unsafe `FixedArray< TAdapted > Create< TActual, TAdapted >` (ref TActual firstField, int length)
Creates a `FixedArray` with a specified length and adapts the type of the elements.
- static unsafe `FixedArray< T > CreateFromFieldSequence< T >` (ref T firstField, ref T lastField)
Creates a `FixedArray` from a sequence of fields.
- static int `IndexOf< T >` (this `FixedArray< T >` array, T elem)
Returns the index of the first occurrence of a value in the `FixedArray`.

Public Attributes

- `T * _array`
- `int _length`

Static Public Attributes

- static `StringBuilder _stringBuilderCached`

Properties

- int `Length` [get]
The fixed size of the array.
- ref T `this[int index]` [get]
Indexer of array elements.

6.40.1 Detailed Description

A fixed size array that can be used in structs.

Helper methods for `FixedArray`.

Template Parameters

<code>T</code>	
----------------	--

Type Constraints

`T : unmanaged`

6.40.2 Constructor & Destructor Documentation

6.40.2.1 FixedArray()

```
FixedArray (
    T * array,
    int length )
```

[NetworkArray](#) constructor.

6.40.3 Member Function Documentation

6.40.3.1 Clear()

```
void Clear ( )
```

Sets all elements in the array to their default value.

6.40.3.2 CopyFrom() [1/2]

```
void CopyFrom (
    List< T > source,
    int sourceOffset,
    int sourceCount )
```

Copies a range of elements from a source list into the [FixedArray](#).

Parameters

<i>source</i>	The source list from which to copy elements. Must not be null.
<i>sourceOffset</i>	The zero-based index in the source list at which copying begins.
<i>sourceCount</i>	The number of elements to copy from the source list.

Exceptions

<i>ArgumentNullException</i>	Thrown when the provided source list is null.
<i>ArgumentException</i>	Thrown when the number of elements to copy is greater than the length of the FixedArray .
<i>ArgumentOutOfRangeException</i>	Thrown when the sum of sourceOffset and sourceCount is greater than the length of the source list.

6.40.3.3 CopyFrom() [2/2]

```
void CopyFrom (
```

```
T[ ] source,
int sourceOffset,
int sourceCount )
```

Copies a range of elements from a source array into the [FixedArray](#).

Parameters

<i>source</i>	The source array from which to copy elements. Must not be null.
<i>sourceOffset</i>	The zero-based index in the source array at which copying begins.
<i>sourceCount</i>	The number of elements to copy from the source array.

Exceptions

<i>ArgumentNullException</i>	Thrown when the provided source array is null.
<i>ArgumentException</i>	Thrown when the number of elements to copy is greater than the length of the FixedArray .
<i>ArgumentOutOfRangeException</i>	Thrown when the sum of sourceOffset and sourceCount is greater than the length of the source array.

6.40.3.4 CopyTo() [1/2]

```
void CopyTo (
    List< T > list )
```

Adds each value to the supplied List. This does not clear the list, so values will be appended to the existing list.

6.40.3.5 CopyTo() [2/2]

```
void CopyTo (
    T[ ] array,
    bool throwIfOverflow = true )
```

Copies values to the supplied array.

Parameters

<i>array</i>	The array to copy values to. Must not be null.
<i>throwIfOverflow</i>	If true, this method will throw an error if the supplied array is smaller than this <code>NetworkArray<T></code> . If false, will only copy as many elements as the target array can hold.

6.40.3.6 Create< T >()

```
static unsafe FixedArray<T> Create< T > (
    ref T firstField,
    int length ) [static]
```

Creates a [FixedArray](#) with a specified length.

Parameters

<i>firstField</i>	Reference to the first field in the array.
<i>length</i>	The length of the array.

Returns

A new [FixedArray](#) instance with the specified length.

Type Constraints

T : unmanaged

6.40.3.7 Create< TActual, TAdapted >()

```
static unsafe FixedArray<TAdapted> Create< TActual, TAdapted > (
    ref TActual firstField,
    int length ) [static]
```

Creates a [FixedArray](#) with a specified length and adapts the type of the elements.

Parameters

<i>firstField</i>	Reference to the first field in the array.
<i>length</i>	The length of the array.

Returns

A new [FixedArray](#) instance with the specified length and adapted type of elements.

Type Constraints

TActual : unmanaged

TAdapted : unmanaged

6.40.3.8 CreateFromFieldSequence< T >()

```
static unsafe FixedArray<T> CreateFromFieldSequence< T > (
    ref T firstField,
    ref T lastField ) [static]
```

Creates a [FixedArray](#) from a sequence of fields.

Parameters

<i>firstField</i>	Reference to the first field in the sequence.
<i>lastField</i>	Reference to the last field in the sequence.

Returns

A new [FixedArray](#) instance with the values from the sequence of fields.

Type Constraints

T : *unmanaged*

6.40.3.9 GetEnumerator()

```
Enumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the [FixedArray](#).

6.40.3.10 IndexOf< T >()

```
static int IndexOf< T > (
    this FixedArray< T > array,
    T elem ) [static]
```

Returns the index of the first occurrence of a value in the [FixedArray](#).

Parameters

<i>array</i>	The FixedArray to search.
<i>elem</i>	The value to locate in the FixedArray .

Returns

The zero-based index of the first occurrence of elem within the entire [FixedArray](#), if found; otherwise, -1.

Type Constraints

T : *unmanaged*

T : IEquatable<T>

6.40.3.11 ToArray()

```
T [ ] ToArray ( )
```

Allocates a new array and copies values from this array. For a non-alloc alternative use [CopyTo\(List<T>\)](#).

6.40.3.12 ToStringString()

```
string ToStringString ( )
```

Returns the elements of this array as a string, with value separated by characters. Specifically for use in the Unity inspector. This is private and only is found by NetworkBehaviourEditor using reflection, so do not rename this method.

6.40.3.13 ToString()

```
override string ToString ( )
```

Returns a string that represents the current object.

6.40.4 Property Documentation

6.40.4.1 Length

```
int Length [get]
```

The fixed size of the array.

6.40.4.2 this[int index]

```
ref T this[int index] [get]
```

Indexer of array elements.

6.41 FixedArray< T >.Enumerator Struct Reference

Enumerator for the [FixedArray](#) struct.

Inherits [IEnumerator< T >](#).

Public Member Functions

- void [Dispose](#) ()
Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.
- [Enumerator](#) ([FixedArray< T >](#) array)
Initializes a new instance of the [Enumerator](#) struct.
- bool [MoveNext](#) ()
Advances the enumerator to the next element of the collection.
- void [Reset](#) ()
Sets the enumerator to its initial position, which is before the first element in the collection.

Public Attributes

- [FixedArray< T >](#) _array
- int _index

Properties

- T [Current](#) [get]
Gets the current element in the collection.
- object [IEnumerator](#).[Current](#) [get]

6.41.1 Detailed Description

Enumerator for the [FixedArray](#) struct.

6.41.2 Constructor & Destructor Documentation

6.41.2.1 Enumerator()

```
Enumerator (
    FixedArray< T > array )
```

Initializes a new instance of the [Enumerator](#) struct.

Parameters

<code>array</code>	The FixedArray instance to enumerate.
--------------------	---

6.41.3 Member Function Documentation

6.41.3.1 Dispose()

```
void Dispose ( )
```

Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.

6.41.3.2 MoveNext()

```
bool MoveNext ( )
```

Advances the enumerator to the next element of the collection.

Returns

True if the enumerator was successfully advanced to the next element; false if the enumerator has passed the end of the collection.

6.41.3.3 Reset()

```
void Reset ( )
```

Sets the enumerator to its initial position, which is before the first element in the collection.

6.41.4 Property Documentation

6.41.4.1 Current

```
T Current [get]
```

Gets the current element in the collection.

6.42 FixedBufferPropertyAttribute Class Reference

Fixed Buffer Property Attribute

Inherits PropertyAttribute.

Public Member Functions

- `FixedBufferPropertyAttribute (Type fieldType, Type surrogateType, int capacity)`
FixedBufferPropertyAttribute Constructor

Properties

- `int Capacity [get]`
Fixed Buffer Capacity
- `Type SurrogateType [get]`
Fixed Buffer Surrogate Type
- `Type Type [get]`
Fixed Buffer Type

6.42.1 Detailed Description

Fixed Buffer Property Attribute

6.42.2 Constructor & Destructor Documentation

6.42.2.1 FixedBufferPropertyAttribute()

```
FixedBufferPropertyAttribute (
    Type fieldType,
    Type surrogateType,
    int capacity )
```

`FixedBufferPropertyAttribute` Constructor

Parameters

<code>fieldType</code>	<code>Type</code>
<code>surrogateType</code>	<code>SurrogateType</code>
<code>capacity</code>	<code>Capacity</code>

6.42.3 Property Documentation

6.42.3.1 Capacity

int Capacity [get]

Fixed Buffer Capacity

6.42.3.2 SurrogateType

Type SurrogateType [get]

Fixed Buffer Surrogate Type

6.42.3.3 Type

Type Type [get]

Fixed Buffer Type

6.43 FixedStorage Class Reference

Provides utility methods for fixed storage types.

Static Public Member Functions

- static int [GetWordCount< T >\(\)](#)
Gets the word count of a fixed storage type.

6.43.1 Detailed Description

Provides utility methods for fixed storage types.

6.43.2 Member Function Documentation

6.43.2.1 GetWordCount< T >()

static int GetWordCount< T > () [static]

Gets the word count of a fixed storage type.

Template Parameters

<i>T</i>	The type of the fixed storage.
----------	--------------------------------

Returns

The word count of the fixed storage type.

Type Constraints

T : unmanaged

T : IFixedStorage

6.44 FloatCompressed Struct Reference

Represents a compressed float value for network transmission.

Inherits [INetworkStruct](#), and [IEquatable<FloatCompressed>](#).

Public Member Functions

- bool [Equals \(FloatCompressed other\)](#)
Checks if the current `FloatCompressed` instance is equal to the other `FloatCompressed` instance.
- override bool [Equals \(object obj\)](#)
Checks if the provided object is a `FloatCompressed` instance and if it's equal to the current `FloatCompressed` instance.
- override int [GetHashCode \(\)](#)
Returns the hash code for the current `FloatCompressed` instance.

Static Public Member Functions

- static implicit operator float ([FloatCompressed q](#))
Implicit conversion from `FloatCompressed` to float.
- static implicit operator [FloatCompressed](#) (float v)
Implicit conversion from float to `FloatCompressed`.
- static bool [operator!= \(FloatCompressed left, FloatCompressed right\)](#)
Inequality operator for `FloatCompressed` struct.
- static bool [operator== \(FloatCompressed left, FloatCompressed right\)](#)
Equality operator for `FloatCompressed` struct.

Public Attributes

- int [valueEncoded](#)
Encoded value of the float.

6.44.1 Detailed Description

Represents a compressed float value for network transmission.

6.44.2 Member Function Documentation

6.44.2.1 Equals() [1/2]

```
bool Equals (
    FloatCompressed other )
```

Checks if the current [FloatCompressed](#) instance is equal to the other [FloatCompressed](#) instance.

Parameters

<i>other</i>	The other FloatCompressed instance to compare with the current FloatCompressed instance.
--------------	--

Returns

True if the values of both [FloatCompressed](#) instances are equal, otherwise false.

6.44.2.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Checks if the provided object is a [FloatCompressed](#) instance and if it's equal to the current [FloatCompressed](#) instance.

Parameters

<i>obj</i>	The object to compare with the current FloatCompressed instance.
------------	--

Returns

True if the provided object is a [FloatCompressed](#) instance and it's equal to the current [FloatCompressed](#) instance, otherwise false.

6.44.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [FloatCompressed](#) instance.

Returns

A hash code for the current [FloatCompressed](#) instance.

6.44.2.4 operator float()

```
static implicit operator float (
    FloatCompressed q ) [static]
```

Implicit conversion from [FloatCompressed](#) to float.

Parameters

<i>q</i>	The FloatCompressed instance to convert.
----------	--

Returns

The decompressed float value of the [FloatCompressed](#) instance.

6.44.2.5 operator [FloatCompressed](#)()

```
static implicit operator FloatCompressed (
    float v ) [static]
```

Implicit conversion from float to [FloatCompressed](#).

Parameters

<i>v</i>	The float value to convert.
----------	-----------------------------

Returns

A new [FloatCompressed](#) instance with the compressed value of the float.

6.44.2.6 operator"!=()

```
static bool operator!= (
    FloatCompressed left,
    FloatCompressed right ) [static]
```

Inequality operator for [FloatCompressed](#) struct.

Parameters

<i>left</i>	First FloatCompressed instance.
<i>right</i>	Second FloatCompressed instance.

Returns

True if the value of the first `FloatCompressed` instance is not equal to the value of the second `FloatCompressed` instance, otherwise false.

6.44.2.7 `operator==()`

```
static bool operator== (
    FloatCompressed left,
    FloatCompressed right ) [static]
```

Equality operator for `FloatCompressed` struct.

Parameters

<i>left</i>	First <code>FloatCompressed</code> instance.
<i>right</i>	Second <code>FloatCompressed</code> instance.

Returns

True if the value of the first `FloatCompressed` instance is equal to the value of the second `FloatCompressed` instance, otherwise false.

6.44.3 Member Data Documentation

6.44.3.1 `valueEncoded`

```
int valueEncoded
```

Encoded value of the float.

6.45 FloatUtils Class Reference

Provides utility methods for compressing and decompressing float values.

Static Public Member Functions

- static unsafe int `Compress` (float f, int accuracy=`DEFAULT_ACCURACY`)
Compresses a float value.
- static unsafe float `Decompress` (int value, float accuracy=`DEFAULT_ACCURACY`)
Decompresses a compressed float value.

Static Public Attributes

- const int `DEFAULT_ACCURACY` = 1 << 10
Default accuracy for float compression and decompression.

6.45.1 Detailed Description

Provides utility methods for compressing and decompressing float values.

6.45.2 Member Function Documentation

6.45.2.1 Compress()

```
static unsafe int Compress (
    float f,
    int accuracy = DEFAULT_ACCURACY) [static]
```

Compresses a float value.

Parameters

<code>f</code>	The float value to compress.
<code>accuracy</code>	The accuracy to use for compression. Defaults to <code>DEFAULT_ACCURACY</code> .

Returns

The compressed float value.

6.45.2.2 Decompress()

```
static unsafe float Decompress (
    int value,
    float accuracy = DEFAULT_ACCURACY) [static]
```

Decompresses a compressed float value.

Parameters

<code>value</code>	The compressed float value to decompress.
<code>accuracy</code>	The accuracy to use for decompression. Defaults to <code>DEFAULT_ACCURACY</code> .

Returns

The decompressed float value.

6.45.3 Member Data Documentation

6.45.3.1 DEFAULT_ACCURACY

```
const int DEFAULT_ACCURACY = 1 << 10 [static]
```

Default accuracy for float compression and decompression.

6.46 FusionGlobalScriptableObject< T > Class Template Reference

A base class for ScriptableObjects that are meant to be globally accessible, at edit-time and runtime. The way such objects are loaded is driven by usages of [FusionGlobalScriptableObjectSourceAttribute](#) attributes.

Inherits [FusionScriptableObject](#), and [FusionGlobalScriptableObject](#).

Protected Member Functions

- virtual void [OnDisable](#) ()
If the current instance is global, unsets [IsGlobal](#) and calls [OnUnloadedAsGlobal](#)
- virtual void [OnLoadedAsGlobal](#) ()
Invoked when the instance is loaded as global.
- virtual void [OnUnloadedAsGlobal](#) (bool destroyed)
Invoked when the instance is unloaded as global.

Static Protected Member Functions

- static bool [TryGetGlobalInternal](#) (out T global)
Loads or returns the current global instance. Returns null if loading an instance failed.
- static bool [UnloadGlobalInternal](#) ()
Unloads the global instance if it is loaded.

Properties

- static T [GlobalInternal](#) [get, set]
A singleton instance-like property. Loads or returns the current global instance. Derived classes can package it in a property with a different name. Throws if loading an instance failed.
- bool [IsGlobal](#) [get]
Is this instance a global instance.
- static bool [IsGlobalLoadedInternal](#) [get]
Returns true if a global instance is loaded. Compared to [GlobalInternal](#), it does not attempt to load an instance.

6.46.1 Detailed Description

A base class for ScriptableObjects that are meant to be globally accessible, at edit-time and runtime. The way such objects are loaded is driven by usages of [FusionGlobalScriptableObjectSourceAttribute](#) attributes.

Type Constraints

$T : \text{FusionGlobalScriptableObject} < T >$

6.46.2 Member Function Documentation

6.46.2.1 OnDisable()

```
virtual void OnDisable () [protected], [virtual]
```

If the current instance is global, unsets [IsGlobal](#) and calls [OnUnloadedAsGlobal](#)

Reimplemented in [NetworkProjectConfigAsset](#).

6.46.2.2 OnLoadedAsGlobal()

```
virtual void OnLoadedAsGlobal () [protected], [virtual]
```

Invoked when the instance is loaded as global.

6.46.2.3 OnUnloadedAsGlobal()

```
virtual void OnUnloadedAsGlobal (
    bool destroyed ) [protected], [virtual]
```

Invoked when the instance is unloaded as global.

Parameters

<i>destroyed</i>	<input type="button" value=""/>
------------------	---------------------------------

6.46.2.4 TryGetGlobalInternal()

```
static bool TryGetGlobalInternal (
    out T global ) [static], [protected]
```

Loads or returns the current global instance. Returns `null` if loading an instance failed.

Parameters

global

Returns

6.46.2.5 UnloadGlobalInternal()

```
static bool UnloadGlobalInternal () [static], [protected]
```

Unloads the global instance if it is loaded.

Returns

`true` if an instance was unloaded

6.46.3 Property Documentation

6.46.3.1 GlobalInternal

```
T GlobalInternal [static], [get], [set], [protected]
```

A singleton instance-like property. Loads or returns the current global instance. Derived classes can package it in a property with a different name. Throws if loading an instance failed.

Exceptions

InvalidOperationException

6.46.3.2 IsGlobal

```
bool IsGlobal [get]
```

Is this instance a global instance.

6.46.3.3 IsGlobalLoadedInternal

```
bool IsGlobalLoadedInternal [static], [get], [protected]
```

Returns true if a global instance is loaded. Compared to [GlobalInternal](#), it does not attempt to load an instance.

6.47 FusionGlobalScriptableObjectAttribute Class Reference

Provides additional information for a global scriptable object.

Inherits Attribute.

Public Member Functions

- [FusionGlobalScriptableObjectAttribute \(string defaultPath\)](#)

Creates a new instance.

Properties

- string [DefaultContents](#) [get, set]
The default contents for the asset, if it is a TextAsset.
- string [DefaultContentsGeneratorMethod](#) [get, set]
Name of the method that is used to generate the default contents for the asset.
- string [DefaultPath](#) [get]
The default path for the asset.

6.47.1 Detailed Description

Provides additional information for a global scriptable object.

6.47.2 Constructor & Destructor Documentation

6.47.2.1 FusionGlobalScriptableObjectAttribute()

```
FusionGlobalScriptableObjectAttribute (
    string defaultPath )
```

Creates a new instance.

Parameters

<code>defaultPath</code>	The default path for the asset.
--------------------------	---------------------------------

6.47.3 Property Documentation

6.47.3.1 DefaultContents

```
string DefaultContents [get], [set]
```

The default contents for the asset, if it is a TextAsset.

6.47.3.2 DefaultContentsGeneratorMethod

```
string DefaultContentsGeneratorMethod [get], [set]
```

Name of the method that is used to generate the default contents for the asset.

6.47.3.3 DefaultPath

```
string DefaultPath [get]
```

The default path for the asset.

6.48 FusionGlobalScriptableObjectLoadResult Struct Reference

The result of [FusionGlobalScriptableObjectSourceAttribute.Load](#). Contains the loaded object and an optional unloader delegate.

Public Member Functions

- [FusionGlobalScriptableObjectLoadResult \(FusionGlobalScriptableObject obj, FusionGlobalScriptableObjectUnloadDelegate unloader=null\)](#)

Static Public Member Functions

- static implicit [operator FusionGlobalScriptableObjectLoadResult \(FusionGlobalScriptableObject result\)](#)
Implicitly converts a FusionGlobalScriptableObject to a FusionGlobalScriptableObjectLoadResult.

Public Attributes

- readonly [FusionGlobalScriptableObject Object](#)
Object instance.
- readonly [FusionGlobalScriptableObjectUnloadDelegate Unloader](#)
An optional delegate that is used to unload Object.

6.48.1 Detailed Description

The result of `FusionGlobalScriptableObjectSourceAttribute.Load`. Contains the loaded object and an optional unloader delegate.

6.48.2 Constructor & Destructor Documentation

6.48.2.1 `FusionGlobalScriptableObjectLoadResult()`

```
FusionGlobalScriptableObjectLoadResult (
    FusionGlobalScriptableObject obj,
    FusionGlobalScriptableObjectUnloadDelegate unloader = null )
```

Parameters

<i>obj</i>	Object instance.
<i>unloader</i>	An optional delegate that is used to unload <i>obj</i> .

6.48.3 Member Function Documentation

6.48.3.1 `operator FusionGlobalScriptableObjectLoadResult()`

```
static implicit operator FusionGlobalScriptableObjectLoadResult (
    FusionGlobalScriptableObject result ) [static]
```

Implicitly converts a `FusionGlobalScriptableObject` to a `FusionGlobalScriptableObjectLoadResult`.

6.48.4 Member Data Documentation

6.48.4.1 `Object`

```
readonly FusionGlobalScriptableObject Object
```

Object instance.

6.48.4.2 Unloader

```
readonly FusionGlobalScriptableObjectUnloadDelegate Unloader
```

An optional delegate that is used to unload [Object](#).

6.49 FusionGlobalScriptableObjectSourceAttribute Class Reference

Base class for all attributes that can be used to load [FusionGlobalScriptableObject](#). Attributes need to be registered at the assembly level. For instance, this snippet is used to register a default loader, that attempts to load from Resources based on [FusionGlobalScriptableObjectAttribute.DefaultPath](#):

Inherits Attribute.

Public Member Functions

- [FusionGlobalScriptableObjectSourceAttribute](#) ([Type objectType](#))

Parameters

objectType	Type or the base type of FusionGlobalScriptableObject that this loader supports.
----------------------------	--

- abstract [FusionGlobalScriptableObjectLoadResult Load](#) ([Type type](#))

Attempt to load the object of the specified type. Return `default` if the object cannot be loaded.

Properties

- bool [AllowEditMode](#) = false [get, set]
Can this loader be used in edit mode.
- bool [AllowFallback](#) = false [get, set]
Does this loader allow fallback to the next loader?
- [Type ObjectType](#) [get]
Type or the base type of [FusionGlobalScriptableObject](#) that this loader supports.
- int [Order](#) [get, set]
Order in which this loader will be executed. Lower values are executed first.

6.49.1 Detailed Description

Base class for all attributes that can be used to load [FusionGlobalScriptableObject](#). Attributes need to be registered at the assembly level. For instance, this snippet is used to register a default loader, that attempts to load from Resources based on [FusionGlobalScriptableObjectAttribute.DefaultPath](#):

```
[assembly: Fusion.FusionGlobalScriptableObjectResource(typeof(Fusion.FusionGlobalScriptableObjectAttribute), Order = 2000, AllowFallback = true)]
```

6.49.2 Member Function Documentation

6.49.2.1 Load()

```
abstract FusionGlobalScriptableObjectLoadResult Load (
    Type type ) [pure virtual]
```

Attempt to load the object of the specified type. Return default if the object cannot be loaded.

Parameters

<i>type</i>	The requested type
-------------	--------------------

6.49.3 Property Documentation

6.49.3.1 AllowEditMode

```
bool AllowEditMode = false [get], [set]
```

Can this loader be used in edit mode.

6.49.3.2 AllowFallback

```
bool AllowFallback = false [get], [set]
```

Does this loader allow fallback to the next loader?

6.49.3.3 ObjectType

```
Type ObjectType [get]
```

Type or the base type of [FusionGlobalScriptableObject](#) that this loader supports.

6.49.3.4 Order

```
int Order [get], [set]
```

Order in which this loader will be executed. Lower values are executed first.

6.50 FusionMonoBehaviour Class Reference

Base class for all [Fusion](#) MonoBehaviours.

Inherits MonoBehaviour.

6.50.1 Detailed Description

Base class for all [Fusion](#) MonoBehaviours.

6.51 FusionScriptableObject Class Reference

Base class for all [Fusion](#) scriptable objects.

Inherits ScriptableObject.

Inherited by [FusionGlobalScriptableObject< NetworkProjectConfigAsset >](#), and [FusionGlobalScriptableObject< T >](#).

6.51.1 Detailed Description

Base class for all [Fusion](#) scriptable objects.

6.52 HeapConfiguration Class Reference

Memory Heap Settings

Public Member Functions

- [HeapConfiguration Init](#) (int globalsSize)
Initializes and creates a new [HeapConfiguration](#) based on the Global Size
- override string [ToString](#) ()
ToString

Public Attributes

- int [GlobalsSize](#)
Heap Global Size
- int [PageCount](#) = [Allocator.Config.DEFAULT_BLOCK_COUNT](#)
Default number of Heap Pages
- [PageSizes PageShift](#) = [Allocator.Config.DEFAULT_BLOCK_SHIFT](#)
Default size of each Heap Page

6.52.1 Detailed Description

Memory Heap Settings

6.52.2 Member Function Documentation

6.52.2.1 Init()

```
HeapConfiguration Init (
    int globalSize )
```

Initializes and creates a new [HeapConfiguration](#) based on the Global Size

6.52.2.2 ToString()

```
override string ToString ( )
```

ToString

6.52.3 Member Data Documentation

6.52.3.1 GlobalsSize

```
int GlobalsSize
```

Heap Global Size

6.52.3.2 PageCount

```
int PageCount = Allocator.Config.DEFAULT_BLOCK_COUNT
```

Default number of Heap Pages

6.52.3.3 PageShift

```
PageSizes PageShift = Allocator.Config.DEFAULT_BLOCK_SHIFT
```

Default size of each Heap Page

6.53 HideArrayElementLabelAttribute Class Reference

Attribute used to hide the label of an array element in the inspector.

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [HideArrayElementLabelAttribute \(\)](#)

Initializes a new instance of the [HideArrayElementLabelAttribute](#) class.

Additional Inherited Members

6.53.1 Detailed Description

Attribute used to hide the label of an array element in the inspector.

6.53.2 Constructor & Destructor Documentation

6.53.2.1 HideArrayElementLabelAttribute()

```
HideArrayElementLabelAttribute ()
```

Initializes a new instance of the [HideArrayElementLabelAttribute](#) class.

6.54 Hitbox Class Reference

Represents a single lag-compensated collider. Multiple component instances can be added anywhere in the hierarchy of a [NetworkObject](#) which includes a [HitboxRoot](#).

Inherits [Behaviour](#).

Public Member Functions

- void [OnDrawGizmos \(\)](#)
Draws this hitbox gizmo on Unity editor.
- void [SetLayer \(int layer\)](#)
Set a layer mask for this [Hitbox](#) gameobject. This method caches the layer mask.

Public Attributes

- Vector3 [BoxExtents](#)
When [Type](#) is set to [HitboxTypes.Box](#), this defines the local-space geometry for narrow-phase checks.
- float [CapsuleExtents](#)
When [Type](#) is set to [HitboxTypes.Capsule](#), this defines the local-space geometry for narrow-phase checks.
- float [CapsuleRadius](#)
When [Type](#) is set to [HitboxTypes.Capsule](#), this defines the local-space geometry for narrow-phase checks.
- Color [GizmosColor](#) = Color.yellow
Color used when drawing gizmos for this hitbox.
- Vector3 [Offset](#)
This [Hitbox](#)'s local-space offset from its [GameObject](#) position.
- HitboxRoot [Root](#)
Reference to the top-level [HitboxRoot](#) component for this [NetworkObject](#).
- float [SphereRadius](#)
When [Type](#) is set to [HitboxTypes.Sphere](#), this defines the local-space geometry for narrow-phase checks.
- HitboxTypes [Type](#)
The collision geometry type for this [Hitbox](#).

Protected Member Functions

- virtual void [DrawGizmos \(Color color, ref Matrix4x4 localToWorldMatrix\)](#)
Draw the [Hitbox](#) gizmos.

Properties

- int [ColliderIndex](#) [get]
Index assigned to the collider of this hitbox on the lag-compensated snapshots.
- bool [HitboxActive](#) [get, set]
Get or set the state of this [Hitbox](#). If a hitbox or its [HitboxRoot](#) are not active, it will not be hit by lag-compensated queries.
- Int32 [HitboxIndex](#) [get]
The index of this hitbox in the [HitboxRoot.Hitboxes](#) array on [Root](#). The value is set by the root when initializing the nested hitboxes with [HitboxRoot.InitHitboxes](#).
- Vector3 [Position](#) [get]
World-space position (includes Offset) of this [Hitbox](#).

Additional Inherited Members

6.54.1 Detailed Description

Represents a single lag-compensated collider. Multiple component instances can be added anywhere in the hierarchy of a [NetworkObject](#) which includes a [HitboxRoot](#).

6.54.2 Member Function Documentation

6.54.2.1 DrawGizmos()

```
virtual void DrawGizmos (
    Color color,
    ref Matrix4x4 localToWorldMatrix ) [protected], [virtual]
```

Draw the [Hitbox](#) gizmos.

6.54.2.2 OnDrawGizmos()

```
void OnDrawGizmos ( )
```

Draws this hitbox gizmo on Unity editor.

6.54.2.3 SetLayer()

```
void SetLayer (
    int layer )
```

Set a layer mask for this [Hitbox](#) gameobject. This method caches the layer mask.

Parameters

<i>layer</i>	<input type="text"/>
--------------	----------------------

6.54.3 Member Data Documentation

6.54.3.1 BoxExtents

```
Vector3 BoxExtents
```

When [Type](#) is set to [HitboxTypes.Box](#), this defines the local-space geometry for narrow-phase checks.

6.54.3.2 CapsuleExtents

```
float CapsuleExtents
```

When [Type](#) is set to [HitboxTypes.Capsule](#), this defines the local-space geometry for narrow-phase checks.

6.54.3.3 CapsuleRadius

```
float CapsuleRadius
```

When [Type](#) is set to [HitboxTypes.Capsule](#), this defines the local-space geometry for narrow-phase checks.

6.54.3.4 GizmosColor

```
Color GizmosColor = Color.yellow
```

Color used when drawing gizmos for this hitbox.

6.54.3.5 Offset

```
Vector3 Offset
```

This [Hitbox](#)'s local-space offset from its [GameObject](#) position.

6.54.3.6 Root

```
HitboxRoot Root
```

Reference to the top-level [HitboxRoot](#) component for this [NetworkObject](#).

6.54.3.7 SphereRadius

```
float SphereRadius
```

When [Type](#) is set to [HitboxTypes.Sphere](#), this defines the local-space geometry for narrow-phase checks.

6.54.3.8 Type

`HitboxTypes` Type

The collision geometry type for this [Hitbox](#).

6.54.4 Property Documentation

6.54.4.1 ColliderIndex

`int ColliderIndex [get]`

Index assigned to the collider of this hitbox on the lag-compensated snapshots.

6.54.4.2 HitboxActive

`bool HitboxActive [get], [set]`

Get or set the state of this [Hitbox](#). If a hitbox or its [HitboxRoot](#) are not active, it will not be hit by lag-compensated queries.

6.54.4.3 HitboxIndex

`Int32 HitboxIndex [get]`

The index of this hitbox in the [HitboxRoot.Hitboxes](#) array on [Root](#). The value is set by the root when initializing the nested hitboxes with [HitboxRoot.InitHitboxes](#).

6.54.4.4 Position

`Vector3 Position [get]`

World-space position (includes Offset) of this [Hitbox](#).

6.55 HitboxManager Class Reference

Entry point for lag compensated [Hitbox](#) queries, which maintains a history buffer, and provides lag compensated raycast and overlap methods. Singleton instance is accessible through the property [Runner.LagCompensation](#).

Inherits [SimulationBehaviour](#), [IAfterTick](#), [IBeforeSimulation](#), and [ISpawned](#).

Public Member Functions

- void [GetPlayerTickAndAlpha](#) ([PlayerRef](#) player, out int? tickFrom, out int? tickTo, out float? alpha)
Gets the tick and alpha interpolate values for a player.
- [LagCompensationStatisticsSnapshot GetStatisticsSnapshot \(\)](#)
Gets a snapshot of the lag compensation statistics.
- int [OverlapBox](#) ([BoxOverlapQuery](#) query, List<[LagCompensatedHit](#)> hits, bool clearHits=true)
Performs a lag-compensated box overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [OverlapBox](#) (Vector3 center, Vector3 extents, Quaternion orientation, int tick, int? tickTo, float? alpha, List<[LagCompensatedHit](#)> hits, int layerMask=-1, [HitOptions](#) options=[HitOptions.None](#), bool clearHits=true, QueryTriggerInteraction queryTriggerInteraction=QueryTriggerInteraction.UseGlobal, PreProcessingDelegate preProcessRoots=null)
Performs a lag-compensated box overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [OverlapBox](#) (Vector3 center, Vector3 extents, Quaternion orientation, [PlayerRef](#) player, List<[LagCompensatedHit](#)> hits, int layerMask=-1, [HitOptions](#) options=[HitOptions.None](#), bool clearHits=true, QueryTriggerInteraction queryTriggerInteraction=QueryTriggerInteraction.UseGlobal, PreProcessingDelegate preProcessRoots=null)
Performs a lag-compensated box overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [OverlapSphere](#) ([SphereOverlapQuery](#) query, List<[LagCompensatedHit](#)> hits, bool clearHits=true)
Performs a lag-compensated sphere overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [OverlapSphere](#) (Vector3 origin, float radius, int tick, int? tickTo, float? alpha, List<[LagCompensatedHit](#)> hits, int layerMask=-1, [HitOptions](#) options=[HitOptions.None](#), bool clearHits=true, QueryTriggerInteraction queryTriggerInteraction=QueryTriggerInteraction.UseGlobal, PreProcessingDelegate preProcessRoots=null)
Performs a lag-compensated overlap sphere query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [OverlapSphere](#) (Vector3 origin, float radius, [PlayerRef](#) player, List<[LagCompensatedHit](#)> hits, int layerMask=-1, [HitOptions](#) options=[HitOptions.None](#), bool clearHits=true, QueryTriggerInteraction queryTriggerInteraction=QueryTriggerInteraction.UseGlobal, PreProcessingDelegate preProcessRoots=null)
Performs a lag-compensated overlap sphere query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- void [PositionRotation](#) ([Hitbox](#) hitbox, int tick, out Vector3 position, out Quaternion rotation, bool subtickAccuracy=false, int? tickTo=null, float? alpha=null)
Performs a lag-compensated query for a specific [Hitbox](#) position and rotation.
- void [PositionRotation](#) ([Hitbox](#) hitbox, [PlayerRef](#) player, out Vector3 position, out Quaternion rotation, bool subTickAccuracy=false)
Performs a lag-compensated query for a specific [Hitbox](#) position and rotation.
- bool [Raycast](#) ([RaycastQuery](#) query, out [LagCompensatedHit](#) hit)
Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- bool [Raycast](#) (Vector3 origin, Vector3 direction, float length, int tick, int? tickTo, float? alpha, out [LagCompensatedHit](#) hit, int layerMask=-1, [HitOptions](#) options=[HitOptions.None](#), QueryTriggerInteraction queryTriggerInteraction=QueryTriggerInteraction.UseGlobal, PreProcessingDelegate preProcessRoots=null)
Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

- bool [Raycast](#) (Vector3 origin, Vector3 direction, float length, [PlayerRef](#) player, out [LagCompensatedHit](#) hit, int layerMask=-1, [HitOptions](#) options=[HitOptions.None](#), [QueryTriggerInteraction](#) queryTrigger←
Interaction=[QueryTriggerInteraction.UseGlobal](#), [PreProcessingDelegate](#) preProcessRoots=null)

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [RaycastAll](#) ([RaycastAllQuery](#) query, List<[LagCompensatedHit](#)> hits, bool clearHits=true)

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.
- int [RaycastAll](#) (Vector3 origin, Vector3 direction, float length, int tick, int? tickTo, float? alpha, List<[LagCompensatedHit](#)> hits, int layerMask=-1, bool clearHits=true, [HitOptions](#) options=[HitOptions.None](#), [QueryTriggerInteraction](#) queryTriggerInteraction=[QueryTriggerInteraction.UseGlobal](#), [PreProcessingDelegate](#) preProcessRoots=null)

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes. Important: results are NOT sorted by distance.
- int [RaycastAll](#) (Vector3 origin, Vector3 direction, float length, [PlayerRef](#) player, List<[LagCompensatedHit](#)> hits, int layerMask=-1, bool clearHits=true, [HitOptions](#) options=[HitOptions.None](#), [QueryTriggerInteraction](#) queryTriggerInteraction=[QueryTriggerInteraction.UseGlobal](#), [PreProcessingDelegate](#) preProcessRoots=null)

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes. Important: results are NOT sorted by distance.

Public Attributes

- int [BVHDepth](#)

Debug data from Broadphase BVH (tree depth).
- int [BVHNodes](#)

Debug data from Broadphase BVH (total nodes count).
- [LagCompensationDraw DrawInfo](#)

Debug data used to draw the BVH nodes and the lag compensation history.
- int [TotalHitboxes](#)

Debug data from lag compensation history (registered [Hitbox](#) count).

Additional Inherited Members

6.55.1 Detailed Description

Entry point for lag compensated [Hitbox](#) queries, which maintains a history buffer, and provides lag compensated raycast and overlap methods. Singleton instance is accessible through the property [Runner.LagCompensation](#).

Usage - Call any of the following methods:

```
HitboxManager.Raycast()
HitboxManager.RaycastAll()
HitboxManager.PositionRotation()
HitboxManager.OverlapSphere()
```

These methods use the history buffer to perform a [Hitbox](#) query against a state consistent with how the indicated [PlayerRef](#) perceived them locally.

6.55.2 Member Function Documentation

6.55.2.1 GetPlayerTickAndAlpha()

```
void GetPlayerTickAndAlpha (
    PlayerRef player,
    out int? tickFrom,
    out int? tickTo,
    out float? alpha )
```

Gets the tick and alpha interpolate values for a player.

Parameters

<i>player</i>	The player reference.
<i>tickFrom</i>	The tick value from which to interpolate.
<i>tickTo</i>	The tick value to which to interpolate.
<i>alpha</i>	The interpolation alpha value.

6.55.2.2 GetStatisticsSnapshot()

```
LagCompensationStatisticsSnapshot GetStatisticsSnapshot ()
```

Gets a snapshot of the lag compensation statistics.

Returns

The lag compensation statistics snapshot.

6.55.2.3 OverlapBox() [1/3]

```
int OverlapBox (
    BoxOverlapQuery query,
    List< LagCompensatedHit > hits,
    bool clearHits = true )
```

Performs a lag-compensated box overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>query</i>	The query containing all necessary information.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).

Returns

The total number of hits found.

6.55.2.4 OverlapBox() [2/3]

```
int OverlapBox (
    Vector3 center,
    Vector3 extents,
    Quaternion orientation,
    int tick,
    int? tickTo,
    float? alpha,
    List< LagCompensatedHit > hits,
    int layerMask = -1,
    HitOptions options = HitOptions.None,
    bool clearHits = true,
    QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
    PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated box overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>center</i>	Center of the box in world space.
<i>extents</i>	Half of the size of the box in each dimension.
<i>orientation</i>	Rotation of the box.
<i>tick</i>	The exact tick to be queried
<i>tickTo</i>	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the <i>alpha</i> parameter for interpolation between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded value of <i>alpha</i> .
<i>alpha</i>	Interpolation value when querying between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded <i>alpha</i> value.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

The total number of hits found.

6.55.2.5 OverlapBox() [3/3]

```
int OverlapBox (
    Vector3 center,
    Vector3 extents,
    Quaternion orientation,
    PlayerRef player,
    List< LagCompensatedHit > hits,
    int layerMask = -1,
    HitOptions options = HitOptions.None,
    bool clearHits = true,
    QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
    PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated box overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>center</i>	Center of the box in world space.
<i>extents</i>	Half of the size of the box in each dimension.
<i>orientation</i>	Rotation of the box.
<i>player</i>	Player who "owns" this overlap. Used by the server to find the exact hitbox snapshots to check against.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

The total number of hits found.

6.55.2.6 OverlapSphere() [1/3]

```
int OverlapSphere (
    SphereOverlapQuery query,
    List< LagCompensatedHit > hits,
    bool clearHits = true )
```

Performs a lag-compensated sphere overlap query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>query</i>	The query containing all necessary information.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).

Returns

The total number of hits found.

6.55.2.7 OverlapSphere() [2/3]

```
int OverlapSphere (
    Vector3 origin,
    float radius,
    int tick,
    int? tickTo,
    float? alpha,
    List< LagCompensatedHit > hits,
    int layerMask = -1,
    HitOptions options = HitOptions.None,
    bool clearHits = true,
    QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
    PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated overlap sphere query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>origin</i>	Sphere center, in world-space
<i>radius</i>	Sphere radius
<i>tick</i>	The tick to be queried
<i>tickTo</i>	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the <i>alpha</i> parameter for interpolation between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded value of <i>alpha</i> .

Parameters

<i>alpha</i>	Interpolation value when querying between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded alpha value.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

total number of hits

6.55.2.8 OverlapSphere() [3/3]

```
int OverlapSphere (
    Vector3 origin,
    float radius,
    PlayerRef player,
    List< LagCompensatedHit > hits,
    int layerMask = -1,
    HitOptions options = HitOptions.None,
    bool clearHits = true,
    QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
    PreProcessingDelegate preprocessRoots = null )
```

Performs a lag-compensated overlap sphere query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>origin</i>	Sphere center, in world-space
<i>radius</i>	Sphere radius
<i>player</i>	Player who "owns" this overlap. Used by the server to find the exact hitbox snapshots to check against.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).

Parameters

<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

total number of hits

6.55.2.9 PositionRotation() [1/2]

```
void PositionRotation (
    Hitbox hitbox,
    int tick,
    out Vector3 position,
    out Quaternion rotation,
    bool subtickAccuracy = false,
    int? tickTo = null,
    float? alpha = null )
```

Performs a lag-compensated query for a specific [Hitbox](#) position and rotation.

Parameters

<i>hitbox</i>	The target hitbox to be queried in the past
<i>tick</i>	The tick to be queried
<i>tickTo</i>	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the <i>alpha</i> parameter for interpolation between <i>tick</i> and <i>tickTo</i> . If <i>subtickAccuracy</i> is requested, the query will return the hitbox state interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded value of <i>alpha</i> .
<i>alpha</i>	Interpolation value when querying between <i>tick</i> and <i>tickTo</i> . If <i>subtickAccuracy</i> is requested, the query will return the hitbox state interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded alpha value.
<i>position</i>	Will be filled with the hitbox position at the time of the tick
<i>rotation</i>	Will be filled with the hitbox rotation at the time of the tick
<i>subtickAccuracy</i>	If the query should interpolate between ticks to reflect exactly what was seen on the client.

6.55.2.10 PositionRotation() [2/2]

```
void PositionRotation (
```

```
    Hitbox hitbox,
    PlayerRef player,
    out Vector3 position,
    out Quaternion rotation,
    bool subTickAccuracy = false )
```

Performs a lag-compensated query for a specific [Hitbox](#) position and rotation.

Parameters

<i>hitbox</i>	The target hitbox to be queried in the past
<i>player</i>	Player who "owns" this overlap. Used by the server to find the exact hitbox snapshots to check against.
<i>position</i>	Will be filled with the hitbox position at the time of the tick
<i>rotation</i>	Will be filled with the hitbox rotation at the time of the tick
<i>subTickAccuracy</i>	If the query should interpolate between ticks to reflect exactly what was seen on the client.

6.55.2.11 Raycast() [1/3]

```
bool Raycast (
    RaycastQuery query,
    out LagCompensatedHit hit )
```

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>query</i>	The query containing all necessary information.
<i>hit</i>	Raycast results will be filled in here.

Returns

The total number of hits found.

6.55.2.12 Raycast() [2/3]

```
bool Raycast (
    Vector3 origin,
    Vector3 direction,
    float length,
    int tick,
    int? tickTo,
    float? alpha,
    out LagCompensatedHit hit,
```

```
int layerMask = -1,
HitOptions options = HitOptions.None,
QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>origin</i>	Raycast origin, in world-space
<i>direction</i>	Raycast direction, in world-space
<i>length</i>	Raycast length
<i>tick</i>	Simulation tick number to use as the time reference for the lag compensation (use this for server AI, and similar).
<i>tickTo</i>	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the <i>alpha</i> parameter for interpolation between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded value of <i>alpha</i> .
<i>alpha</i>	Interpolation value when querying between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded alpha value.
<i>hit</i>	Raycast results will be filled in here.
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

True if something is hit

6.55.2.13 Raycast() [3/3]

```
bool Raycast (
    Vector3 origin,
    Vector3 direction,
    float length,
    PlayerRef player,
    out LagCompensatedHit hit,
    int layerMask = -1,
```

```
HitOptions options = HitOptions.None,
QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>origin</i>	Raycast origin, in world-space
<i>direction</i>	Raycast direction, in world-space
<i>length</i>	Raycast length
<i>player</i>	Player who "owns" this raycast. Used by the server to find the exact hitbox snapshots to check against.
<i>hit</i>	Raycast results will be filled in here.
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

True if something is hit

6.55.2.14 RaycastAll() [1/3]

```
int RaycastAll (
    RaycastAllQuery query,
    List< LagCompensatedHit > hits,
    bool clearHits = true )
```

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

<i>query</i>	The query containing all necessary information.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).

Returns

The total number of hits found.

6.55.2.15 RaycastAll() [2/3]

```
int RaycastAll (
    Vector3 origin,
    Vector3 direction,
    float length,
    int tick,
    int? tickTo,
    float? alpha,
    List< LagCompensatedHit > hits,
    int layerMask = -1,
    bool clearHits = true,
    HitOptions options = HitOptions.None,
    QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
    PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes. Important: results are NOT sorted by distance.

Parameters

<i>origin</i>	Raycast origin, in world-space
<i>direction</i>	Raycast direction, in world-space
<i>length</i>	Raycast length
<i>tick</i>	Simulation tick number to use as the time reference for the lag compensation (use this for server AI, and similar).
<i>tickTo</i>	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the <i>alpha</i> parameter for interpolation between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded value of <i>alpha</i> .
<i>alpha</i>	Interpolation value when querying between <i>tick</i> and <i>tickTo</i> . If HitOptions.SubtickAccuracy is included on <i>options</i> , this query will be resolved against hitbox colliders interpolated between the specified ticks. Otherwise, only one of the two ticks will be considered, according to the rounded <i>alpha</i> value.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

total number of hits

6.55.2.16 RaycastAll() [3/3]

```
int RaycastAll (
    Vector3 origin,
    Vector3 direction,
    float length,
    PlayerRef player,
    List< LagCompensatedHit > hits,
    int layerMask = -1,
    bool clearHits = true,
    HitOptions options = HitOptions.None,
    QueryTriggerInteraction queryTriggerInteraction = QueryTriggerInteraction.UseGlobal,
    PreProcessingDelegate preProcessRoots = null )
```

Performs a lag-compensated raycast query against all registered hitboxes. If the [HitOptions.IncludePhysX](#) or [HitOptions.IncludeBox2D](#) flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes. Important: results are NOT sorted by distance.

Parameters

<i>origin</i>	Raycast origin, in world-space
<i>direction</i>	Raycast direction, in world-space
<i>length</i>	Raycast length
<i>player</i>	Player who "owns" this raycast. Used by the server to find the exact hitbox snapshots to check against.
<i>hits</i>	List to be filled with hits (both hitboxes and/or static colliders, if included).
<i>layerMask</i>	Only objects with matching layers will be checked against.
<i>options</i>	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
<i>clearHits</i>	Clear list of hits before filling with new ones (defaults to true).
<i>queryTriggerInteraction</i>	Trigger interaction behavior when also querying PhysX.
<i>preProcessRoots</i>	Delegate to pre-process HitboxRoots found in the broad-phase resolution of the query. Roots removed from the list will not be processed any further. Roots that remain on the candidates collection will be normally processed and fitting colliders will be evaluated in the query narrow-phase resolution. Hitbox collider indices added to the processed set will be evaluated in the narrow-phase regardless of further root processing steps (e.g. layer mask match).

Returns

total number of hits

6.55.3 Member Data Documentation

6.55.3.1 BVHDepth

```
int BVHDepth
```

Debug data from Broadphase BVH (tree depth).

6.55.3.2 BVHNodes

```
int BVHNodes
```

Debug data from Broadphase BVH (total nodes count).

6.55.3.3 DrawInfo

```
LagCompensationDraw DrawInfo
```

Debug data used to draw the BVH nodes and the lag compensation history.

6.55.3.4 TotalHitboxes

```
int TotalHitboxes
```

Debug data from lag compensation history (registered [Hitbox](#) count).

6.56 HitboxRoot Class Reference

Root [Hitbox](#) group container. Manages registering/unregistering hitboxes with the group, and defines the broad-phase geometry for the group.

Inherits [NetworkBehaviour](#).

Public Types

- enum class [ConfigFlags](#) : int
Set of configuration options for a [Hitbox](#) Root behaviour.

Public Member Functions

- override void **Despawned** (NetworkRunner runner, bool hasState)
Called before the network object is despawned
- void **InitHitboxes** ()
Finds child Hitbox components, and adds them to the Hitboxes collection.
- bool **IsHitboxActive** (Hitbox hitbox)
Checks the state of a Hitbox instance under this root. Both the hitbox and its root must be active in order for it to be hit by lag-compensated queries.
- void **OnDrawGizmos** ()
HitboxRoot on draw gizmos.
- void **SetHitboxActive** (Hitbox hitbox, bool setActive)
Sets the state of a Hitbox instance under this root. Both the hitbox and its root must be active in order for it to be hit by lag-compensated queries.
- void **SetMinBoundingRadius** ()
Sets BroadRadius to a rough value which encompasses all Hitboxes in their current positions.

Public Attributes

- float **BroadRadius**
The radius of the broadphase bounding sphere for this Hitbox group. Used by HitboxManager to insert/update lag compensated NetworkObjects into its BVH (bounding volume hierarchy) data structure. Be sure this radius encompasses all children Hitbox components (including their full ranges of animation motion). We plan to offer an option to dynamically compute the bounding volume, but the performance trade off will still favor a hand-crafted radius.
- ConfigFlags **Config** = ConfigFlags.Default
Set of configuration options for this Hitbox Root behaviour. Check the API documentation for more details on what each flag represents.
- Color **GizmosColor** = Color.gray
Color used when drawing gizmos for this hitbox.
- Hitbox[] **Hitboxes**
All Hitbox instances in hierarchy. Auto-filled at Spawner.
- Vector3 **Offset**
Local-space offset of the broadphase bounding sphere from its transform position.

Static Public Attributes

- const Int32 **MAX_HITBOXES** = (sizeof(UInt32) * 8) - 1
The max number of hitboxes allowed under the same root.

Protected Member Functions

- virtual void **DrawGizmos** (Color color, ref Matrix4x4 localToWorldMatrix)
Draws the gizmos for the HitboxRoot

Properties

- bool **HitboxRootActive** [get, set]
Get or set the state of this HitboxRoot. For a hitbox to be hit by lag-compensated queries, both it and its HitboxRoot must be active.
- bool **InInterest** [get]
If this HitboxRoot is in interest for the local player.
- HitboxManager **Manager** [get]
Reference to associated hitbox manager (from which lag compensated queries can be performed).

Additional Inherited Members

6.56.1 Detailed Description

Root [Hitbox](#) group container. Manages registering/unregistering hitboxes with the group, and defines the broad-phase geometry for the group.

Broadphase is the initial rough query used by raycasts/overlaps/etc to find potential hit candidates, which are then used in the final **narrowphase** query.

6.56.2 Member Enumeration Documentation

6.56.2.1 ConfigFlags

```
enum ConfigFlags : int [strong]
```

Set of configuration options for a [Hitbox](#) Root behaviour.

Enumerator

ReinitializeHitboxesBeforeRegistration	If the collection of hitboxes under a given root should be re-initialized before the Root is registered in a hitbox snapshot. If disabled, the hitboxes will be used as configured in edit-time.
IncludeInactiveHitboxes	If Hitboxes on inactive Game Objects should be registered under this root upon initialization.
Legacy	Set of configuration flags that replicate the behaviour as it was before the flag options were added.
Default	Ser of configuration flags with the default behaviour, suitable for most use-cases.

6.56.3 Member Function Documentation

6.56.3.1 DrawGizmos()

```
virtual void DrawGizmos (
    Color color,
    ref Matrix4x4 localToWorldMatrix ) [protected], [virtual]
```

Draws the gizmos for the [HitboxRoot](#)

6.56.3.2 InitHitboxes()

```
void InitHitboxes ( )
```

Finds child [Hitbox](#) components, and adds them to the [Hitboxes](#) collection.

6.56.3.3 IsHitboxActive()

```
bool IsHitboxActive (
    Hitbox hitbox )
```

Checks the state of a [Hitbox](#) instance under this root. Both the hitbox and its root must be active in order for it to be hit by lag-compensated queries.

Parameters

<i>hitbox</i>	A hitbox instance under the hierarchy of this root.
---------------	---

Returns

True if the *hitbox* is part of this root and is active.

Exceptions

ArgumentOutOfRangeException	If the Hitbox.HitboxIndex of the <i>hitbox</i> is outside the valid range.
AssertException	In Debug configuration, if the <i>hitbox</i> is not part of this root.

6.56.3.4 OnDrawGizmos()

```
void OnDrawGizmos ( )
```

[HitboxRoot](#) on draw gizmos.

6.56.3.5 SetHitboxActive()

```
void SetHitboxActive (
    Hitbox hitbox,
    bool setActive )
```

Sets the state of a [Hitbox](#) instance under this root. Both the hitbox and its root must be active in order for it to be hit by lag-compensated queries.

Parameters

<i>hitbox</i>	A hitbox instance under the hierarchy of this root.
<i>setActive</i>	If the hitbox should be activated or deactivated.

Exceptions

<i>ArgumentOutOfRangeException</i>	If the Hitbox.HitboxIndex of the <i>hitbox</i> is outside the valid range.
<i>AssertException</i>	In Debug configuration, if the <i>hitbox</i> is not part of this root.

6.56.3.6 SetMinBoundingRadius()

```
void SetMinBoundingRadius ( )
```

Sets [BroadRadius](#) to a rough value which encompasses all [Hitboxes](#) in their current positions.

6.56.4 Member Data Documentation**6.56.4.1 BroadRadius**

```
float BroadRadius
```

The radius of the broadphase bounding sphere for this [Hitbox](#) group. Used by [HitboxManager](#) to insert/update lag compensated NetworkObjects into its BVH (bounding volume hierarchy) data structure. Be sure this radius encompasses all children [Hitbox](#) components (including their full ranges of animation motion). We plan to offer an option to dynamically compute the bounding volume, but the performance trade off will still favor a hand-crafted radius.

Broadphase is the initial rough query used by raycasts/overlaps/etc to find potential hit candidates, which are then used in the final **narrowphase** query.

6.56.4.2 Config

```
ConfigFlags Config = ConfigFlags.Default
```

Set of configuration options for this [Hitbox](#) Root behaviour. Check the API documentation for more details on what each flag represents.

6.56.4.3 GizmosColor

```
Color GizmosColor = Color.gray
```

Color used when drawing gizmos for this hitbox.

6.56.4.4 Hitboxes

```
Hitbox [] Hitboxes
```

All [Hitbox](#) instances in hierarchy. Auto-filled at Spawned.

6.56.4.5 MAX_HITBOXES

```
const Int32 MAX_HITBOXES = (sizeof(UInt32) * 8) - 1 [static]
```

The max number of hitboxes allowed under the same root.

6.56.4.6 Offset

```
Vector3 Offset
```

Local-space offset of the broadphase bounding sphere from its transform position.

Adjust the [BroadRadius](#) and [Offset](#) until the sphere gizmo (shown in the Unity Scene window) encompasses all children [Hitbox](#) components (including their full ranges of animation motion).

Broadphase is the initial rough query used by raycasts/overlaps/etc to find potential hit candidates, which are then used in the final **narrowphase** query.

6.56.5 Property Documentation

6.56.5.1 HitboxRootActive

```
bool HitboxRootActive [get], [set]
```

Get or set the state of this [HitboxRoot](#). For a hitbox to be hit by lag-compensated queries, both it and its [HitboxRoot](#) must be active.

6.56.5.2 InInterest

```
bool InInterest [get]
```

If this [HitboxRoot](#) is in interest for the local player.

6.56.5.3 Manager

```
HitboxManager Manager [get]
```

Reference to associated hitbox manager (from which lag compensated queries can be performed).

6.57 HostMigrationConfig Class Reference

Project configuration settings specific to how the Host Migration behaves.

Public Attributes

- bool [EnableAutoUpdate](#)
Enabled the Host Migration feature
- int [UpdateDelay](#) = 10
Delay between Host Migration Snapshot updates

6.57.1 Detailed Description

Project configuration settings specific to how the Host Migration behaves.

6.57.2 Member Data Documentation

6.57.2.1 EnableAutoUpdate

```
bool EnableAutoUpdate
```

Enabled the Host Migration feature

6.57.2.2 UpdateDelay

```
int UpdateDelay = 10
```

Delay between Host Migration Snapshot updates

6.58 HostMigrationToken Class Reference

Transitory Holder with all necessary information to restart the [Fusion](#) Runner after the Host Migration has completed

Properties

- [GameMode GameMode](#) [get]

New GameMode the local peer will assume after the Host Migration

6.58.1 Detailed Description

Transitory Holder with all necessary information to restart the [Fusion](#) Runner after the Host Migration has completed

6.58.2 Property Documentation

6.58.2.1 GameMode

[GameMode GameMode](#) [get]

New GameMode the local peer will assume after the Host Migration

6.59 IAfterAllTicks Interface Reference

Interface for [AfterAllTicks](#) callback. Called after the re-simulation loop (when applicable), and also after the forward simulation loop. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherited by [NetworkMecanimAnimator](#), and [NetworkTransform](#).

Public Member Functions

- void [AfterAllTicks](#) (bool resimulation, int tickCount)

Called after the re-simulation loop (when applicable), and also after the forward simulation loop. Only called on Updates where re-simulation or forward ticks are processed.

6.59.1 Detailed Description

Interface for [AfterAllTicks](#) callback. Called after the re-simulation loop (when applicable), and also after the forward simulation loop. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.59.2 Member Function Documentation

6.59.2.1 AfterAllTicks()

```
void AfterAllTicks (
    bool resimulation,
    int tickCount )
```

Called after the re-simulation loop (when applicable), and also after the forward simulation loop. Only called on Updates where re-simulation or forward ticks are processed.

Parameters

<i>resimulation</i>	True if this is being called during the re-simulation loop. False if during the forward simulation loop.
<i>TickCount</i>	How many re-simulation or forward ticks are going to be processed.

6.60 IAfterClientPredictionReset Interface Reference

Callback interface for [AfterClientPredictionReset](#). Called at the very start of the resimulation loop (on clients with prediction enabled), immediately after state is set to the latest server snapshot. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Public Member Functions

- void [AfterClientPredictionReset\(\)](#)

Called at the very start of the resimulation loop (on clients with prediction enabled), immediately after state is set to the latest server snapshot.

6.60.1 Detailed Description

Callback interface for [AfterClientPredictionReset](#). Called at the very start of the resimulation loop (on clients with prediction enabled), immediately after state is set to the latest server snapshot. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.60.2 Member Function Documentation

6.60.2.1 AfterClientPredictionReset()

```
void AfterClientPredictionReset ( )
```

Called at the very start of the resimulation loop (on clients with prediction enabled), immediately after state is set to the latest server snapshot.

6.61 IAfterHostMigration Interface Reference

Used to mark NetworkBehaviors that need to be react after a Host Migration process

Public Member Functions

- void [AfterHostMigration\(\)](#)

Invoked after the Host Migration happens in order to setup non-networked data on NetworkBehaviors

6.61.1 Detailed Description

Used to mark NetworkBehaviors that need to be react after a Host Migration process

6.61.2 Member Function Documentation

6.61.2.1 AfterHostMigration()

```
void AfterHostMigration ( )
```

Invoked after the Host Migration happens in order to setup non-networked data on NetworkBehaviors

6.62 IAfterRender Interface Reference

Interface for [AfterRender](#) callback. Called after the render loop.

Public Member Functions

- void [AfterRender](#) ()
Called after the render loop.

6.62.1 Detailed Description

Interface for [AfterRender](#) callback. Called after the render loop.

6.62.2 Member Function Documentation

6.62.2.1 AfterRender()

```
void AfterRender ( )
```

Called after the render loop.

6.63 IAfterSpawned Interface Reference

Interface for [AfterSpawned](#) callback. Called after the object is spawned.

Public Member Functions

- void [AfterSpawned \(\)](#)
Called after the object is spawned.

6.63.1 Detailed Description

Interface for [AfterSpawned](#) callback. Called after the object is spawned.

6.63.2 Member Function Documentation

6.63.2.1 AfterSpawned()

```
void AfterSpawned ( )
```

Called after the object is spawned.

6.64 IAfterTick Interface Reference

Interface for [AfterTick](#) callback. Called after each tick simulation completes. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherited by [HitboxManager](#).

Public Member Functions

- void [AfterTick \(\)](#)
Called after each tick simulation completes.

6.64.1 Detailed Description

Interface for [AfterTick](#) callback. Called after each tick simulation completes. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.64.2 Member Function Documentation

6.64.2.1 AfterTick()

```
void AfterTick ( )
```

Called after each tick simulation completes.

6.65 IAfterUpdate Interface Reference

Interface for the [AfterUpdate](#) callback, which is called at the end of each [Fusion](#) Update segment. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Public Member Functions

- void [AfterUpdate](#) ()

Called at the end of the Fusion Update loop, before all Unity MonoBehaviour.Update() callbacks.

6.65.1 Detailed Description

Interface for the [AfterUpdate](#) callback, which is called at the end of each [Fusion](#) Update segment. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.65.2 Member Function Documentation

6.65.2.1 AfterUpdate()

```
void AfterUpdate ( )
```

Called at the end of the [Fusion](#) Update loop, before all Unity MonoBehaviour.Update() callbacks.

6.66 IAsyncOperation Interface Reference

Defines an asynchronous operation.

Inherited by [ICoroutine](#).

Properties

- [ExceptionDispatchInfo](#) [Error](#) [get]
Gets the exception information if an error occurred during the operation.
- bool [IsDone](#) [get]
Gets a value indicating whether the operation is done.

Events

- Action<[IAsyncOperation](#)> Completed
Occurs when the operation is completed.

6.66.1 Detailed Description

Defines an asynchronous operation.

6.66.2 Property Documentation

6.66.2.1 Error

`ExceptionDispatchInfo Error [get]`

Gets the exception information if an error occurred during the operation.

6.66.2.2 IsDone

`bool IsDone [get]`

Gets a value indicating whether the operation is done.

6.66.3 Event Documentation

6.66.3.1 Completed

`Action<IAsyncOperation> Completed`

Occurs when the operation is completed.

6.67 IBeforeAllTicks Interface Reference

Interface for [BeforeAllTicks](#) callback. Called before the re-simulation loop (when applicable), and also before the forward simulation loop. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherited by [NetworkTransform](#).

Public Member Functions

- void [BeforeAllTicks](#) (bool resimulation, int tickCount)

Called before the re-simulation loop (when applicable), and also before the forward simulation loop. Only called on Updates where re-simulation or forward ticks are processed.

6.67.1 Detailed Description

Interface for [BeforeAllTicks](#) callback. Called before the re-simulation loop (when applicable), and also before the forward simulation loop. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.67.2 Member Function Documentation

6.67.2.1 BeforeAllTicks()

```
void BeforeAllTicks (
    bool resimulation,
    int tickCount )
```

Called before the re-simulation loop (when applicable), and also before the forward simulation loop. Only called on Updates where re-simulation or forward ticks are processed.

Parameters

<i>resimulation</i>	True if this is being called during the re-simulation loop. False if during the forward simulation loop.
<i>tickCount</i>	How many re-simulation or forward ticks are going to be processed.

6.68 IBeforeClientPredictionReset Interface Reference

Callback interface for [BeforeClientPredictionReset](#). Called at the very start of the re-simulation loop (on clients with prediction enabled), before state is set to the latest server snapshot. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Public Member Functions

- void [BeforeClientPredictionReset](#) ()

Called at the very start of the re-simulation loop (on clients with prediction enabled), before state is set to the latest server snapshot.

6.68.1 Detailed Description

Callback interface for [BeforeClientPredictionReset](#). Called at the very start of the re-simulation loop (on clients with prediction enabled), before state is set to the latest server snapshot. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.68.2 Member Function Documentation

6.68.2.1 BeforeClientPredictionReset()

```
void BeforeClientPredictionReset ( )
```

Called at the very start of the re-simulation loop (on clients with prediction enabled), before state is set to the latest server snapshot.

6.69 IBeforeCopyPreviousState Interface Reference

Interface for [BeforeCopyPreviousState](#) callback. Called before the copy of the previous state.

Inherited by [NetworkTransform](#).

Public Member Functions

- void [BeforeCopyPreviousState](#) ()
Called before the copy of the previous state.

6.69.1 Detailed Description

Interface for [BeforeCopyPreviousState](#) callback. Called before the copy of the previous state.

6.69.2 Member Function Documentation

6.69.2.1 BeforeCopyPreviousState()

```
void BeforeCopyPreviousState ( )
```

Called before the copy of the previous state.

6.70 IBeforeHitboxRegistration Interface Reference

Interface for [BeforeHitboxRegistration](#) callback. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Public Member Functions

- void [BeforeHitboxRegistration \(\)](#)

Called immediately before the [HitboxManager](#) registers hitboxes in a snapshot.

6.70.1 Detailed Description

Interface for [BeforeHitboxRegistration](#) callback. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.70.2 Member Function Documentation

6.70.2.1 BeforeHitboxRegistration()

```
void BeforeHitboxRegistration ( )
```

Called immediately before the [HitboxManager](#) registers hitboxes in a snapshot.

6.71 IBeforeSimulation Interface Reference

Interface for [BeforeSimulation](#) callback. Called before both the re-simulation (when applicable) and forward simulation loops. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Inherited by [HitboxManager](#).

Public Member Functions

- void [BeforeSimulation \(int forwardTickCount\)](#)

Called before both the re-simulation (when applicable) and forward simulation loops. Only called on updates that have forward ticks to process.

6.71.1 Detailed Description

Interface for [BeforeSimulation](#) callback. Called before both the re-simulation (when applicable) and forward simulation loops. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.71.2 Member Function Documentation

6.71.2.1 BeforeSimulation()

```
void BeforeSimulation (
    int forwardTickCount )
```

Called before both the re-simulation (when applicable) and forward simulation loops. Only called on updates that have forward ticks to process.

Parameters

<i>forwardTickCount</i>	How many forward ticks are going to be processed.
-------------------------	---

6.72 IBeforeTick Interface Reference

Interface for [BeforeTick](#) callback. Called before each tick is simulated. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Public Member Functions

- void [BeforeTick](#) ()
Called before each tick is simulated.

6.72.1 Detailed Description

Interface for [BeforeTick](#) callback. Called before each tick is simulated. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.72.2 Member Function Documentation

6.72.2.1 BeforeTick()

```
void BeforeTick ( )
```

Called before each tick is simulated.

6.73 IBeforeUpdate Interface Reference

Interface for the [BeforeUpdate](#) callback, which is called at the beginning of each [Fusion](#) Update segment. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Public Member Functions

- void [BeforeUpdate](#) ()
Called at the start of the [Fusion](#) Update loop, before the [Fusion](#) simulation loop.

6.73.1 Detailed Description

Interface for the [BeforeUpdate](#) callback, which is called at the beginning of each [Fusion](#) Update segment. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.73.2 Member Function Documentation

6.73.2.1 BeforeUpdate()

```
void BeforeUpdate ( )
```

Called at the start of the [Fusion](#) Update loop, before the [Fusion](#) simulation loop.

6.74 ICOROUTINE Interface Reference

Defines a coroutine.

Inherits [IAsyncOperation](#), and [IEnumerator](#).

Additional Inherited Members

6.74.1 Detailed Description

Defines a coroutine.

6.75 IDespawned Interface Reference

Interface for [Despawned](#) callback. Called when a [NetworkBehaviour](#) is despawned.

Inherited by [NetworkBehaviour](#).

Public Member Functions

- void [Despawned](#) ([NetworkRunner](#) runner, bool hasState)
Called when a [NetworkBehaviour](#) is despawned.

6.75.1 Detailed Description

Interface for [Despawned](#) callback. Called when a [NetworkBehaviour](#) is despawned.

6.75.2 Member Function Documentation

6.75.2.1 Despawned()

```
void Despawned (   
    NetworkRunner runner,   
    bool hasState )
```

Called when a [NetworkBehaviour](#) is despawned.

Parameters

<i>runner</i>	NetworkRunner that despawned the NetworkBehaviour .
<i>hasState</i>	Whether the NetworkBehaviour has state.

Implemented in [HitboxRoot](#), and [NetworkBehaviour](#).

6.76 IElementReaderWriter< T > Interface Template Reference

Defines the interface for reading and writing elements in a byte array.

Public Member Functions

- int [GetElementHashCode](#) (T element)
Calculate the hash code of an element.
- int [GetElementWordCount](#) ()
Gets the word count of an element.
- T [Read](#) (byte *data, int index)
Reads an element from the specified index in the byte array.
- ref T [ReadRef](#) (byte *data, int index)
Reads a reference to an element from the specified index in the byte array.
- void [Write](#) (byte *data, int index, T element)
Writes an element to the specified index in the byte array.

6.76.1 Detailed Description

Defines the interface for reading and writing elements in a byte array.

Template Parameters

<i>T</i>	The type of the elements.
----------	---------------------------

6.76.2 Member Function Documentation

6.76.2.1 GetElementHashCode()

```
int GetElementHashCode (
    T element )
```

Calculate the hash code of an element.

Parameters

<i>element</i>	<input type="checkbox"/>
----------------	--------------------------

Returns**6.76.2.2 GetElementWordCount()**

```
int GetElementWordCount ( )
```

Gets the word count of an element.

Returns

The word count of an element.

6.76.2.3 Read()

```
T Read (
    byte * data,
    int index )
```

Reads an element from the specified index in the byte array.

Parameters

<i>data</i>	The byte array.
<i>index</i>	The index of the element.

Returns

The element at the specified index.

6.76.2.4 ReadRef()

```
ref T ReadRef (
    byte * data,
    int index )
```

Reads a reference to an element from the specified index in the byte array.

Parameters

<i>data</i>	The byte array.
<i>index</i>	The index of the element.

Returns

A reference to the element at the specified index.

6.76.2.5 Write()

```
void Write (
    byte * data,
    int index,
    T element )
```

Writes an element to the specified index in the byte array.

Parameters

<i>data</i>	The byte array.
<i>index</i>	The index at which to write the element.
<i>element</i>	The element to write.

6.77 IFixedStorage Interface Reference

Interface for fixed storage types.

Inherited by [_128](#), [_16](#), [_2](#), [_256](#), [_32](#), [_4](#), [_512](#), [_64](#), and [_8](#).

6.77.1 Detailed Description

Interface for fixed storage types.

6.78 IIInputAuthorityGained Interface Reference

Interface for handling the event when the input authority is gained.

Public Member Functions

- void [InputAuthorityGained \(\)](#)

Method to be called when the input authority is gained.

6.78.1 Detailed Description

Interface for handling the event when the input authority is gained.

6.78.2 Member Function Documentation

6.78.2.1 InputAuthorityGained()

```
void InputAuthorityGained ( )
```

Method to be called when the input authority is gained.

6.79 IInputAuthorityLost Interface Reference

Interface for handling the event when the input authority is lost.

Public Member Functions

- void [InputAuthorityLost](#) ()
Method to be called when the input authority is lost.

6.79.1 Detailed Description

Interface for handling the event when the input authority is lost.

6.79.2 Member Function Documentation

6.79.2.1 InputAuthorityLost()

```
void InputAuthorityLost ( )
```

Method to be called when the input authority is lost.

6.80 IInterestEnter Interface Reference

Interface for handling the event when a player enters the area of interest.

Public Member Functions

- void [InterestEnter \(PlayerRef player\)](#)

Method to be called when a player enters the area of interest.

6.80.1 Detailed Description

Interface for handling the event when a player enters the area of interest.

6.80.2 Member Function Documentation

6.80.2.1 InterestEnter()

```
void InterestEnter (
    PlayerRef player )
```

Method to be called when a player enters the area of interest.

Parameters

<i>player</i>	The player who entered the area of interest.
---------------	--

6.81 IInterestExit Interface Reference

Interface for handling the event when a player exits the area of interest.

Public Member Functions

- void [InterestExit \(PlayerRef player\)](#)

Method to be called when a player exits the area of interest.

6.81.1 Detailed Description

Interface for handling the event when a player exits the area of interest.

6.81.2 Member Function Documentation

6.81.2.1 InterestExit()

```
void InterestExit (
    PlayerRef player )
```

Method to be called when a player exits the area of interest.

Parameters

<i>player</i>	The player who exited the area of interest.
---------------	---

6.82 ILocalPrefabCreated Interface Reference

Interface for handling the event when a local prefab is created.

Public Member Functions

- void [LocalPrefabCreated \(\)](#)
Method to be called when a local prefab is created.

6.82.1 Detailed Description

Interface for handling the event when a local prefab is created.

6.82.2 Member Function Documentation

6.82.2.1 LocalPrefabCreated()

```
void LocalPrefabCreated ( )
```

Method to be called when a local prefab is created.

6.83 INetworkArray Interface Reference

Defines the interface for a networked array.

Inherits [IEnumerable](#).

Inherited by [NetworkArray< T >](#).

Properties

- object [this\[int index\]](#) [get, set]
Gets or sets the element at the specified index.

6.83.1 Detailed Description

Defines the interface for a networked array.

6.83.2 Property Documentation

6.83.2.1 this[int index]

```
object this[int index] [get], [set]
```

Gets or sets the element at the specified index.

Parameters

<i>index</i>	The zero-based index of the element to get or set.
--------------	--

6.84 INetworkAssetSource< T > Interface Template Reference

Interface for a network asset source.

Public Member Functions

- void [Acquire](#) (bool synchronous)
Acquires the network asset.
- void [Release](#) ()
Releases the network asset.
- T [WaitForResult](#) ()
Waits for the result of the network asset acquisition.

Properties

- string [Description](#) [get]
Gets the description of the network asset.
- bool [IsCompleted](#) [get]
Checks if the network asset acquisition is completed.

6.84.1 Detailed Description

Interface for a network asset source.

Template Parameters

<i>T</i>	Type of the network asset.
----------	----------------------------

6.84.2 Member Function Documentation

6.84.2.1 Acquire()

```
void Acquire (
    bool synchronous )
```

Acquires the network asset.

Parameters

<code>synchronous</code>	If true, the acquisition is done synchronously.
--------------------------	---

6.84.2.2 Release()

```
void Release ( )
```

Releases the network asset.

6.84.2.3 WaitForResult()

```
T WaitForResult ( )
```

Waits for the result of the network asset acquisition.

Returns

The network asset.

6.84.3 Property Documentation**6.84.3.1 Description**

```
string Description [get]
```

Gets the description of the network asset.

6.84.3.2 IsCompleted

```
bool IsCompleted [get]
```

Checks if the network asset acquisition is completed.

6.85 INetworkDictionary Interface Reference

Defines the interface for a networked dictionary.

Inherits IEnumerable.

Inherited by [NetworkDictionary< K, V >](#).

Public Member Functions

- void [Add](#) (object item)
Adds an item to the networked dictionary.

6.85.1 Detailed Description

Defines the interface for a networked dictionary.

6.85.2 Member Function Documentation

6.85.2.1 Add()

```
void Add (
    object item )
```

Adds an item to the networked dictionary.

Parameters

<i>item</i>	The item to add to the dictionary.
-------------	------------------------------------

Implemented in [NetworkDictionary< K, V >](#).

6.86 INetworkInput Interface Reference

Flag interface for custom [NetworkInput](#) structs.

6.86.1 Detailed Description

Flag interface for custom [NetworkInput](#) structs.

6.87 INetworkLinkedList Interface Reference

Defines the interface for a networked linked list.

Inherits [IEnumerable](#).

Inherited by [NetworkLinkedList< T >](#).

Public Member Functions

- void [Add](#) (object item)
Adds an item to the networked linked list.

6.87.1 Detailed Description

Defines the interface for a networked linked list.

6.87.2 Member Function Documentation

6.87.2.1 Add()

```
void Add (object item)
```

Adds an item to the networked linked list.

Parameters

<i>item</i>	The item to add to the linked list.
-------------	-------------------------------------

Implemented in [NetworkLinkedList< T >](#).

6.88 INetworkObjectInitializer Interface Reference

Interface for initializing network objects.

Inherited by [NetworkObjectInitializerUnity](#).

Public Member Functions

- void [InitializeNetworkState](#) ([NetworkObject](#) networkObject)
Initializes the network object.

6.88.1 Detailed Description

Interface for initializing network objects.

6.88.2 Member Function Documentation

6.88.2.1 InitializeNetworkState()

```
void InitializeNetworkState (
    NetworkObject networkObject )
```

Initializes the network object.

Parameters

<code>networkObject</code>	The network object to initialize.
----------------------------	-----------------------------------

Implemented in [NetworkObjectInitializerUnity](#).

6.89 INetworkObjectProvider Interface Reference

Interface which defines the handlers for [NetworkRunner](#) `Spawn()` and `Despawn()` actions. Passing an instance of this interface to [NetworkRunner.StartGame\(StartGameArgs\)](#) as the `StartGameArgs.ObjectProvider` argument value will assign that instance as the handler for runner `Spawn()` and `Despawn()` actions. By default (if `StartGameArgs.ObjectProvider == null`) actions will use `Instantiate()`, and `Despawn()` actions will use `Destroy()`.

Inherited by [NetworkObjectProviderDummy](#).

Public Member Functions

- `NetworkObjectAcquireResult AcquirePrefabInstance (NetworkRunner runner, in NetworkPrefabAcquireContext context, out NetworkObject result)`
Acquires an instance of a prefab for a network object.
- `void ReleaseInstance (NetworkRunner runner, in NetworkObjectReleaseContext context)`
Releases an instance of a network object.

6.89.1 Detailed Description

Interface which defines the handlers for [NetworkRunner](#) `Spawn()` and `Despawn()` actions. Passing an instance of this interface to [NetworkRunner.StartGame\(StartGameArgs\)](#) as the `StartGameArgs.ObjectProvider` argument value will assign that instance as the handler for runner `Spawn()` and `Despawn()` actions. By default (if `StartGameArgs.ObjectProvider == null`) actions will use `Instantiate()`, and `Despawn()` actions will use `Destroy()`.

6.89.2 Member Function Documentation

6.89.2.1 AcquirePrefabInstance()

```
NetworkObjectAcquireResult AcquirePrefabInstance (
    NetworkRunner runner,
    in NetworkPrefabAcquireContext context,
    out NetworkObject result )
```

Acquires an instance of a prefab for a network object.

Parameters

<i>runner</i>	The NetworkRunner that manages the network objects.
<i>context</i>	The context that provides information for acquiring the prefab instance.
<i>result</i>	The acquired NetworkObject instance.

Returns

A [NetworkObjectAcquireResult](#) indicating the result of the operation.

Implemented in [NetworkObjectProviderDummy](#).

6.89.2.2 ReleaseInstance()

```
void ReleaseInstance (
    NetworkRunner runner,
    in NetworkObjectReleaseContext context )
```

Releases an instance of a network object.

Parameters

<i>runner</i>	The NetworkRunner that manages the network objects.
<i>context</i>	The context that provides information for releasing the network object instance.

Implemented in [NetworkObjectProviderDummy](#).

6.90 INetworkPrefabSource Interface Reference

Interface for a network prefab source.

Inherits [INetworkAssetSource< NetworkObject >](#).

Properties

- [NetworkObjectGuid AssetGuid](#) [get]
Gets the GUID of the network object asset.

Additional Inherited Members**6.90.1 Detailed Description**

Interface for a network prefab source.

6.90.2 Property Documentation

6.90.2.1 AssetGuid

`NetworkObjectGuid AssetGuid [get]`

Gets the GUID of the network object asset.

6.91 INetworkRunnerCallbacks Interface Reference

Interface for `NetworkRunner` callbacks. Register a class/struct instance which implements this interface with `NetworkRunner.AddCallbacks(INetworkRunnerCallbacks[])`.

Inherited by `NetworkDelegates`, and `NetworkEvents`.

Public Member Functions

- void `OnConnectedToServer (NetworkRunner runner)`
Callback when NetworkRunner successfully connects to a server or host.
- void `OnConnectFailed (NetworkRunner runner, NetAddress remoteAddress, NetConnectFailedReason reason)`
Callback when NetworkRunner fails to connect to a server or host.
- void `OnConnectRequest (NetworkRunner runner, NetworkRunnerCallbackArgs.ConnectRequest request, byte[] token)`
Callback when NetworkRunner receives a Connection Request from a Remote Client
- void `OnCustomAuthenticationResponse (NetworkRunner runner, Dictionary< string, object > data)`
Callback is invoked when the Authentication procedure returns a response from the Authentication Server
- void `OnDisconnectedFromServer (NetworkRunner runner, NetDisconnectReason reason)`
Callback when NetworkRunner disconnects from a server or host.
- void `OnHostMigration (NetworkRunner runner, HostMigrationToken hostMigrationToken)`
Callback is invoked when the Host Migration process has started
- void `OnInput (NetworkRunner runner, NetworkInput input)`
Callback from NetworkRunner that polls for user inputs. The NetworkInput that is supplied expects:
- void `OnInputMissing (NetworkRunner runner, PlayerRef player, NetworkInput input)`
Callback from NetworkRunner when an input is missing.
- void `OnObjectEnterAOI (NetworkRunner runner, NetworkObject obj, PlayerRef player)`
Callback from a NetworkRunner when a new NetworkObject has entered the Area of Interest
- void `OnObjectExitAOI (NetworkRunner runner, NetworkObject obj, PlayerRef player)`
Callback from a NetworkRunner when a new NetworkObject has exit the Area of Interest
- void `OnPlayerJoined (NetworkRunner runner, PlayerRef player)`
Callback from a NetworkRunner when a new player has joined.
- void `OnPlayerLeft (NetworkRunner runner, PlayerRef player)`
Callback from a NetworkRunner when a player has disconnected.
- void `OnReliableDataProgress (NetworkRunner runner, PlayerRef player, ReliableKey key, float progress)`
Callback is invoked when a Reliable Data Stream is being received, reporting its progress

- void `OnReliableDataReceived (NetworkRunner runner, PlayerRef player, ReliableKey key, ArraySegment< byte > data)`
Callback is invoked when a Reliable Data Stream has been received
- void `OnSceneLoadDone (NetworkRunner runner)`
Callback is invoked when a Scene Load has finished
- void `OnSceneLoadStart (NetworkRunner runner)`
Callback is invoked when a Scene Load has started
- void `OnSessionListUpdated (NetworkRunner runner, List< SessionInfo > sessionList)`
This callback is invoked when a new List of Sessions is received from Photon Cloud
- void `OnShutdown (NetworkRunner runner, ShutdownReason shutdownReason)`
Called when the runner is shutdown
- void `OnUserSimulationMessage (NetworkRunner runner, SimulationMessagePtr message)`
This callback is invoked when a manually dispatched simulation message is received from a remote peer

6.91.1 Detailed Description

Interface for `NetworkRunner` callbacks. Register a class/struct instance which implements this interface with `NetworkRunner.AddCallbacks(INetworkRunnerCallbacks[])`.

6.91.2 Member Function Documentation

6.91.2.1 OnConnectedToServer()

```
void OnConnectedToServer (
    NetworkRunner runner )
```

Callback when `NetworkRunner` successfully connects to a server or host.

6.91.2.2 OnConnectFailed()

```
void OnConnectFailed (
    NetworkRunner runner,
    NetAddress remoteAddress,
    NetConnectFailedReason reason )
```

Callback when `NetworkRunner` fails to connect to a server or host.

6.91.2.3 OnConnectRequest()

```
void OnConnectRequest (
    NetworkRunner runner,
    NetworkRunnerCallbackArgs.ConnectRequest request,
    byte[] token )
```

Callback when `NetworkRunner` receives a Connection Request from a Remote Client

Parameters

<i>runner</i>	Local NetworkRunner
<i>request</i>	Request information
<i>token</i>	Request Token

6.91.2.4 OnCustomAuthenticationResponse()

```
void OnCustomAuthenticationResponse (
    NetworkRunner runner,
    Dictionary< string, object > data )
```

Callback is invoked when the Authentication procedure returns a response from the Authentication Server

Parameters

<i>runner</i>	The runner this object exists on
<i>data</i>	Custom Authentication Reply Values

6.91.2.5 OnDisconnectedFromServer()

```
void OnDisconnectedFromServer (
    NetworkRunner runner,
    NetDisconnectReason reason )
```

Callback when [NetworkRunner](#) disconnects from a server or host.

6.91.2.6 OnHostMigration()

```
void OnHostMigration (
    NetworkRunner runner,
    HostMigrationToken hostMigrationToken )
```

Callback is invoked when the Host Migration process has started

Parameters

<i>runner</i>	The runner this object exists on
<i>hostMigrationToken</i>	Migration Token that stores all necessary information to restart the Fusion Runner

6.91.2.7 OnInput()

```
void OnInput (
    NetworkRunner runner,
    NetworkInput input )
```

Callback from [NetworkRunner](#) that polls for user inputs. The [NetworkInput](#) that is supplied expects:

```
input.Set(new CustomINetworkInput() { /* your values */ });
```

6.91.2.8 OnInputMissing()

```
void OnInputMissing (
    NetworkRunner runner,
    PlayerRef player,
    NetworkInput input )
```

Callback from [NetworkRunner](#) when an input is missing.

Parameters

<i>runner</i>	NetworkRunner reference
<i>player</i>	PlayerRef reference which the input is missing from
<i>input</i>	NetworkInput reference which is missing

6.91.2.9 OnObjectEnterAOI()

```
void OnObjectEnterAOI (
    NetworkRunner runner,
    NetworkObject obj,
    PlayerRef player )
```

Callback from a [NetworkRunner](#) when a new [NetworkObject](#) has entered the Area of Interest

Parameters

<i>runner</i>	NetworkRunner reference
<i>obj</i>	NetworkObject reference
<i>player</i>	PlayerRef reference

6.91.2.10 OnObjectExitAOI()

```
void OnObjectExitAOI (
    NetworkRunner runner,
```

```
    NetworkObject obj,
    PlayerRef player )
```

Callback from a [NetworkRunner](#) when a new [NetworkObject](#) has exit the Area of Interest

Parameters

<i>runner</i>	NetworkRunner reference
<i>obj</i>	NetworkObject reference
<i>player</i>	PlayerRef reference

6.91.2.11 OnPlayerJoined()

```
void OnPlayerJoined (
    NetworkRunner runner,
    PlayerRef player )
```

Callback from a [NetworkRunner](#) when a new player has joined.

6.91.2.12 OnPlayerLeft()

```
void OnPlayerLeft (
    NetworkRunner runner,
    PlayerRef player )
```

Callback from a [NetworkRunner](#) when a player has disconnected.

6.91.2.13 OnReliableDataProgress()

```
void OnReliableDataProgress (
    NetworkRunner runner,
    PlayerRef player,
    ReliableKey key,
    float progress )
```

Callback is invoked when a Reliable Data Stream is being received, reporting its progress

Parameters

<i>runner</i>	NetworkRunner reference
<i>player</i>	Which PlayerRef the stream is being sent from
<i>key</i>	ReliableKey reference that identifies the data stream
<i>progress</i>	Progress of the stream

6.91.2.14 OnReliableDataReceived()

```
void OnReliableDataReceived (
    NetworkRunner runner,
    PlayerRef player,
    ReliableKey key,
    ArraySegment< byte > data )
```

Callback is invoked when a Reliable Data Stream has been received

Parameters

<i>runner</i>	NetworkRunner reference
<i>player</i>	Which PlayerRef the stream was sent from
<i>key</i>	ReliableKey reference that identifies the data stream
<i>data</i>	Data received

6.91.2.15 OnSceneLoadDone()

```
void OnSceneLoadDone (
    NetworkRunner runner )
```

Callback is invoked when a Scene Load has finished

Parameters

<i>runner</i>	NetworkRunner reference
---------------	-------------------------

6.91.2.16 OnSceneLoadStart()

```
void OnSceneLoadStart (
    NetworkRunner runner )
```

Callback is invoked when a Scene Load has started

Parameters

<i>runner</i>	NetworkRunner reference
---------------	-------------------------

6.91.2.17 OnSessionListUpdated()

```
void OnSessionListUpdated (
    NetworkRunner runner,
    List< SessionInfo > sessionList )
```

This callback is invoked when a new List of Sessions is received from Photon Cloud

Parameters

<i>runner</i>	The runner this object exists on
<i>sessionList</i>	Updated list of Session

6.91.2.18 OnShutdown()

```
void OnShutdown (
    NetworkRunner runner,
    ShutdownReason shutdownReason )
```

Called when the runner is shutdown

Parameters

<i>runner</i>	The runner being shutdown
<i>shutdownReason</i>	Describes the reason Fusion was Shutdown

6.91.2.19 OnUserSimulationMessage()

```
void OnUserSimulationMessage (
    NetworkRunner runner,
    SimulationMessagePtr message )
```

This callback is invoked when a manually dispatched simulation message is received from a remote peer

Parameters

<i>runner</i>	The runner this message is for
<i>message</i>	The message pointer

6.92 INetworkRunnerUpdater Interface Reference

Interface which defines the handlers for [NetworkRunner](#) Updates. An implementation is responsible for calling [NetworkRunner.UpdateInternal\(double\)](#) and [NetworkRunner.RenderInternal](#) periodically.

Inherited by [NetworkRunnerUpdaterDefault](#), and [NetworkRunnerUpdaterDummy](#).

Public Member Functions

- void [Initialize](#) ([NetworkRunner](#) runner)
Called when the NetworkRunner is started.
- void [Shutdown](#) ([NetworkRunner](#) runner)
Called when the NetworkRunner is stopped.

6.92.1 Detailed Description

Interface which defines the handlers for [NetworkRunner](#) Updates. An implementation is responsible for calling [NetworkRunner.UpdateInternal\(double\)](#) and [NetworkRunner.RenderInternal](#) periodically.

An instance of this interface can be passed to [NetworkRunner.StartGame\(StartGameArgs\)](#) as the [StartGameArgs.Updater](#). By default (if [StartGameArgs.Updater == null](#)) [Fusion](#) will use [NetworkRunnerUpdaterDefault](#), which invokes [NetworkRunner.UpdateInternal\(double\)](#) before script's Update and [NetworkRunner.RenderInternal](#) before LateUpdate.

6.92.2 Member Function Documentation

6.92.2.1 Initialize()

```
void Initialize (
    NetworkRunner runner )
```

Called when the [NetworkRunner](#) is started.

Parameters

<i>runner</i>	The NetworkRunner instance.
---------------	---

6.92.2.2 Shutdown()

```
void Shutdown (
    NetworkRunner runner )
```

Called when the [NetworkRunner](#) is stopped.

Parameters

<i>runner</i>	The NetworkRunner instance.
---------------	---

6.93 INetworkSceneManager Interface Reference

Interface for a [NetworkRunner](#) scene manager. A scene manager is responsible for loading and unloading scenes

Inherited by [NetworkSceneManagerDummy](#).

Public Member Functions

- **[SceneRef GetSceneRef](#)** ([GameObject](#) gameObject)

Gets a [SceneRef](#) for the scene that the given [GameObject](#) belongs to.
- **[SceneRef GetSceneRef](#)** (string sceneNameOrPath)

Gets a [SceneRef](#) for the given scene name or path.
- void [Initialize](#) ([NetworkRunner](#) runner)

Callback for initialization
- bool [IsRunnerScene](#) ([Scene](#) scene)

Signals if the given scene is the main runner scene. Mostly used for Multipeer logic
- **[NetworkSceneAsyncOp LoadScene](#)** ([SceneRef](#) sceneRef, [NetworkLoadSceneParameters](#) parameters)

Loads a given scene with the specified parameters.
- void [MakeDontDestroyOnLoad](#) ([GameObject](#) obj)

Mark an object as [DontDestroyOnLoad](#).
- bool [MoveGameObjectToScene](#) ([GameObject](#) gameObject, [SceneRef](#) sceneRef)

Move a [GameObject](#) to a desired scene.
- bool [OnSceneInfoChanged](#) ([NetworkSceneInfo](#) sceneInfo, [NetworkSceneInfoChangeSource](#) changeSource)

Implement this method and return true if you want to handle scene info changes manually. Return false if the default scene info change handling should be done by the [NetworkRunner](#) instead.
- void [Shutdown](#) ()

Callback for shutdown and clean up
- bool [TryGetPhysicsScene2D](#) (out [PhysicsScene2D](#) scene2D)

Tries to get the physics scene 2D.
- bool [TryGetPhysicsScene3D](#) (out [PhysicsScene](#) scene3D)

Tries to get the physics scene 3D.
- **[NetworkSceneAsyncOp UnloadScene](#)** ([SceneRef](#) sceneRef)

Unloads a given scene.

Properties

- bool [IsBusy](#) [get]

Signals if the [INetworkSceneManager](#) instance is busy with any scene loading operations
- [Scene](#) [MainRunnerScene](#) [get]

The main scene of the [NetworkRunner](#). Mostly used for Multipeer logic

6.93.1 Detailed Description

Interface for a [NetworkRunner](#) scene manager. A scene manager is responsible for loading and unloading scenes

6.93.2 Member Function Documentation

6.93.2.1 GetSceneRef() [1/2]

```
SceneRef GetSceneRef (
    GameObject gameObject )
```

Gets a [SceneRef](#) for the scene that the given GameObject belongs to.

6.93.2.2 GetSceneRef() [2/2]

```
SceneRef GetSceneRef (
    string sceneNameOrPath )
```

Gets a [SceneRef](#) for the given scene name or path.

6.93.2.3 Initialize()

```
void Initialize (
    NetworkRunner runner )
```

Callback for initialization

6.93.2.4 IsRunnerScene()

```
bool IsRunnerScene (
    Scene scene )
```

Signals if the given scene is the main runner scene. Mostly used for Multipeer logic

6.93.2.5 LoadScene()

```
NetworkSceneAsyncOp LoadScene (
    SceneRef sceneRef,
    NetworkLoadSceneParameters parameters )
```

Loads a given scene with the specified parameters.

Returns

Returns a [NetworkSceneAsyncOp](#) that can be waited

6.93.2.6 MakeDontDestroyOnLoad()

```
void MakeDontDestroyOnLoad (
    GameObject obj )
```

Mark an object as DontDestroyOnLoad.

6.93.2.7 MoveGameObjectToScene()

```
bool MoveGameObjectToScene (
    GameObject gameObject,
    SceneRef sceneRef )
```

Move a GameObject to a desired scene.

Returns

Return true if the operation was successfully

6.93.2.8 OnSceneInfoChanged()

```
bool OnSceneInfoChanged (
    NetworkSceneInfo sceneInfo,
    NetworkSceneInfoChangeSource changeSource )
```

Implement this method and return true if you want to handle scene info changes manually. Return false if the default scene info change handling should be done by the [NetworkRunner](#) instead.

Returns

Return true if a custom handling is provided, false otherwise to use the default one

6.93.2.9 Shutdown()

```
void Shutdown ( )
```

Callback for shutdown and clean up

6.93.2.10 TryGetPhysicsScene2D()

```
bool TryGetPhysicsScene2D (
    out PhysicsScene2D scene2D )
```

Tries to get the physics scene 2D.

Returns

Returns true if the operation was successfully

6.93.2.11 TryGetPhysicsScene3D()

```
bool TryGetPhysicsScene3D (
    out PhysicsScene scene3D )
```

Tries to get the physics scene 3D.

Returns

Returns true if the operation was successfully

6.93.2.12 UnloadScene()

```
NetworkSceneAsyncOp UnloadScene (
    SceneRef sceneRef )
```

Unloads a given scene.

Returns

Returns a [NetworkSceneAsyncOp](#) that can be waited

6.93.3 Property Documentation

6.93.3.1 IsBusy

```
bool IsBusy [get]
```

Signals if the [INetworkSceneManager](#) instance is busy with any scene loading operations

6.93.3.2 MainRunnerScene

```
Scene MainRunnerScene [get]
```

The main scene of the [NetworkRunner](#). Mostly used for Multipeer logic

6.94 INetworkStruct Interface Reference

Base interface for all [Fusion](#) Network Structs

Inherited by [Angle](#), BitSet128, BitSet192, BitSet256, BitSet512, BitSet64, [FloatCompressed](#), [NetworkBehaviourId](#), [NetworkBool](#), [NetworkButtons](#), [NetworkId](#), [NetworkObjectGuid](#), [NetworkObjectHeader](#), [NetworkObjectNestingKey](#), [NetworkObjectTypeId](#), [NetworkPhysicsInfo](#), [NetworkPrefabId](#), [NetworkPrefabRef](#), [NetworkRNG](#), [NetworkSceneInfo](#), [NetworkString< TSize >](#), [NetworkTRSPData](#), [PlayerRef](#), [Ptr](#), [QuaternionCompressed](#), [SceneRef](#), [TickTimer](#), [Vector2Compressed](#), [Vector3Compressed](#), [Vector4Compressed](#), [_128](#), [_16](#), [_2](#), [_256](#), [_32](#), [_4](#), [_512](#), [_64](#), and [_8](#).

6.94.1 Detailed Description

Base interface for all [Fusion](#) Network Structs

6.95 INetworkTRSPTeleport Interface Reference

Implement this interface on a [NetworkTRSP](#) implementation to indicate it can be teleported.

Inherited by [NetworkTransform](#).

Public Member Functions

- void [Teleport](#) (Vector3? position=null, Quaternion? rotation=null)
Teleports to the indicated values, and network the Teleport event.

6.95.1 Detailed Description

Implement this interface on a [NetworkTRSP](#) implementation to indicate it can be teleported.

6.95.2 Member Function Documentation

6.95.2.1 Teleport()

```
void Teleport (
    Vector3? position = null,
    Quaternion? rotation = null )
```

Teleports to the indicated values, and network the Teleport event.

Implemented in [NetworkTransform](#).

6.96 InlineHelpAttribute Class Reference

If applied to a field, checks if there is a help text for the field or the field's type and shows it in the inspector.

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [InlineHelpAttribute \(\)](#)
Initializes a new instance of the [InlineHelpAttribute](#) class.

Properties

- bool [ShowTypeHelp](#) = true [get, set]
Gets or sets a value indicating whether to show help for the type.

Additional Inherited Members

6.96.1 Detailed Description

If applied to a field, checks if there is a help text for the field or the field's type and shows it in the inspector.

6.96.2 Constructor & Destructor Documentation

6.96.2.1 [InlineHelpAttribute\(\)](#)

[InlineHelpAttribute](#) ()

Initializes a new instance of the [InlineHelpAttribute](#) class.

6.96.3 Property Documentation

6.96.3.1 ShowTypeHelp

```
bool ShowTypeHelp = true [get], [set]
```

Gets or sets a value indicating whether to show help for the type.

6.97 IUnitySurrogate Interface Reference

Represents an interface for Unity surrogates. This interface provides methods for reading and writing data.

Inherited by [IUnityValueSurrogate< T >](#), and [UnitySurrogateBase](#).

Public Member Functions

- void [Read](#) (int *data, int capacity)
Reads data from a specified memory location.
- void [Write](#) (int *data, int capacity)
Writes data to a specified memory location.

6.97.1 Detailed Description

Represents an interface for Unity surrogates. This interface provides methods for reading and writing data.

6.97.2 Member Function Documentation

6.97.2.1 Read()

```
void Read (
    int * data,
    int capacity )
```

Reads data from a specified memory location.

Parameters

<i>data</i>	The memory location to read from.
<i>capacity</i>	The number of elements to read.

Implemented in [UnityValueSurrogate< T, TReaderWriter >](#), [UnitySurrogateBase](#), [UnityLinkedListSurrogate< T, ReaderWriter >](#), [UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >](#), and [UnityArraySurrogate< T, ReaderWriter >](#).

6.97.2.2 Write()

```
void Write (
    int * data,
    int capacity )
```

Writes data to a specified memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>capacity</i>	The number of elements to write.

Implemented in [UnityValueSurrogate< T, TReaderWriter >](#), [UnitySurrogateBase](#), [UnityLinkedListSurrogate< T, ReaderWriter >](#), [UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >](#), and [UnityArraySurrogate< T, ReaderWriter >](#).

6.98 IUnityValueSurrogate< T > Interface Template Reference

Represents an interface for Unity value surrogates. This interface provides a property for accessing and modifying the data.

Inherits [IUnitySurrogate](#).

Inherited by [UnityValueSurrogate< T, TReaderWriter >](#).

Properties

- [T DataProperty \[get, set\]](#)
Gets or sets the data for the Unity value surrogate.

Additional Inherited Members

6.98.1 Detailed Description

Represents an interface for Unity value surrogates. This interface provides a property for accessing and modifying the data.

Template Parameters

<i>T</i>	The type of the data.
----------	-----------------------

6.98.2 Property Documentation

6.98.2.1 DataProperty

`T DataProperty [get], [set]`

Gets or sets the data for the Unity value surrogate.

6.99 UnityArraySurrogate< T, ReaderWriter > Class Template Reference

A base class for Unity array surrogates.

Inherits [UnitySurrogateBase](#).

Public Member Functions

- `override void Init (int capacity)`
Initializes the [UnityArraySurrogate](#) with a specified capacity.
- `override void Read (int *data, int capacity)`
Reads data into the [UnityArraySurrogate](#) from a specified memory location.
- `override void Write (int *data, int capacity)`
Writes data from the [UnityArraySurrogate](#) to a specified memory location.

Properties

- `abstract T[] DataProperty [get, set]`
Gets or sets the data array for the [UnityArraySurrogate](#).

6.99.1 Detailed Description

A base class for Unity array surrogates.

Template Parameters

<code>T</code>	Unmanaged type of the array elements.
<code>ReaderWriter</code>	Unmanaged type of the reader/writer for the array elements.

Type Constraints

`T : unmanaged`
`ReaderWriter : unmanaged`
`ReaderWriter : IElementReaderWriter<T>`

6.99.2 Member Function Documentation

6.99.2.1 Init()

```
override void Init (
    int capacity ) [virtual]
```

Initializes the [UnityArraySurrogate](#) with a specified capacity.

Parameters

<i>capacity</i>	The capacity to initialize the UnityArraySurrogate with.
-----------------	--

Implements [UnitySurrogateBase](#).

6.99.2.2 Read()

```
override void Read (
    int * data,
    int capacity ) [virtual]
```

Reads data into the [UnityArraySurrogate](#) from a specified memory location.

Parameters

<i>data</i>	The memory location to read from.
<i>capacity</i>	The number of elements to read.

Implements [UnitySurrogateBase](#).

6.99.2.3 Write()

```
override void Write (
    int * data,
    int capacity ) [virtual]
```

Writes data from the [UnityArraySurrogate](#) to a specified memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>capacity</i>	The number of elements to write.

Implements [UnitySurrogateBase](#).

6.99.3 Property Documentation

6.99.3.1 DataProperty

abstract T [] DataProperty [get], [set]

Gets or sets the data array for the [UnityArraySurrogate](#).

6.100 UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter > Class Template Reference

A surrogate for serializing a dictionary.

Inherits [UnitySurrogateBase](#).

Public Member Functions

- override void [Init](#) (int capacity)
Initializes the [UnityDictionarySurrogate](#) with a specified capacity.
- override void [Read](#) (int *data, int capacity)
Reads data into the [UnityDictionarySurrogate](#) from a specified memory location.
- override void [Write](#) (int *data, int capacity)
Writes data from the [UnityDictionarySurrogate](#) to a specified memory location.

Properties

- abstract [SerializableDictionary](#)< TKeyType, TValueType > [DataProperty](#) [get, set]
Gets or sets the data property.

6.100.1 Detailed Description

A surrogate for serializing a dictionary.

Template Parameters

<i>TKeyType</i>	The type of the key.
<i>TKeyReaderWriter</i>	The type of the key reader writer.
<i>TValueType</i>	The type of the value.
<i>TValueReaderWriter</i>	The type of the value reader writer.

See also

[Fusion.Internal.UnitySurrogateBase](#)

Type Constraints

TKeyType : *unmanaged*
TKeyReaderWriter : *unmanaged*
TKeyReaderWriter : *IElementReaderWriter*<*TKeyType*>
TValueType : *unmanaged*
TValueReaderWriter : *unmanaged*
TValueReaderWriter : *IElementReaderWriter*<*TValueType*>

6.100.2 Member Function Documentation

6.100.2.1 Init()

```
override void Init (
    int capacity ) [virtual]
```

Initializes the [UnityDictionarySurrogate](#) with a specified capacity.

Parameters

<i>capacity</i>	The capacity to initialize the UnityDictionarySurrogate with.
-----------------	---

Implements [UnitySurrogateBase](#).

6.100.2.2 Read()

```
override void Read (
    int * data,
    int capacity ) [virtual]
```

Reads data into the [UnityDictionarySurrogate](#) from a specified memory location.

Parameters

<i>data</i>	The memory location to read from.
<i>capacity</i>	The number of elements to read.

Implements [UnitySurrogateBase](#).

6.100.2.3 Write()

```
override void Write (
    int * data,
    int capacity ) [virtual]
```

Writes data from the [UnityDictionarySurrogate](#) to a specified memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>capacity</i>	The number of elements to write.

Implements [UnitySurrogateBase](#).

6.100.3 Property Documentation

6.100.3.1 DataProperty

```
abstract SerializableDictionary<TKeyType, TValueType> DataProperty [get], [set]
```

Gets or sets the data property.

6.101 UnityLinkedListSurrogate< T, ReaderWriter > Class Template Reference

A surrogate for serializing a linked list.

Inherits [UnitySurrogateBase](#).

Public Member Functions

- override void [Init](#) (int capacity)
Initializes the [UnityLinkedListSurrogate](#) with a specified capacity.
- override void [Read](#) (int *data, int capacity)
Reads data into the [UnityLinkedListSurrogate](#) from a specified memory location.
- override void [Write](#) (int *data, int capacity)
Writes data from the [UnityLinkedListSurrogate](#) to a specified memory location.

Properties

- abstract T[] [DataProperty](#) [get, set]
Gets or sets the data property.

6.101.1 Detailed Description

A surrogate for serializing a linked list.

Template Parameters

<i>T</i>	The type of the elements in the linked list.
<i>ReaderWriter</i>	The type of the reader writer for the elements in the linked list.

Type Constraints

T : *unmanaged**ReaderWriter* : *unmanaged**ReaderWriter* : *IElementReaderWriter*<*T*>

6.101.2 Member Function Documentation

6.101.2.1 Init()

```
override void Init (
    int capacity ) [virtual]
```

Initializes the [UnityLinkedListSurrogate](#) with a specified capacity.

Parameters

<i>capacity</i>	The capacity to initialize the UnityLinkedListSurrogate with.
-----------------	---

Implements [UnitySurrogateBase](#).

6.101.2.2 Read()

```
override void Read (
    int * data,
    int capacity ) [virtual]
```

Reads data into the [UnityLinkedListSurrogate](#) from a specified memory location.

Parameters

<i>data</i>	The memory location to read from.
<i>capacity</i>	The number of elements to read.

Implements [UnitySurrogateBase](#).

6.101.2.3 Write()

```
override void Write (
    int * data,
    int capacity ) [virtual]
```

Writes data from the [UnityLinkedListSurrogate](#) to a specified memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>capacity</i>	The number of elements to write.

Implements [UnitySurrogateBase](#).

6.101.3 Property Documentation

6.101.3.1 DataProperty

```
abstract T [] DataProperty [get], [set]
```

Gets or sets the data property.

6.102 UnitySurrogateBase Class Reference

Represents a base class for Unity surrogates. This class is serializable and provides abstract methods for reading, writing, and initializing data.

Inherits [IUnitySurrogate](#).

Inherited by [UnityArraySurrogate< T, ReaderWriter >](#), [UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >](#), [UnityLinkedListSurrogate< T, ReaderWriter >](#), and [UnityValueSurrogate< T, TReaderWriter >](#).

Public Member Functions

- abstract void [Init](#) (int capacity)
Initializes the [UnitySurrogateBase](#) with a specified capacity.
- abstract void [Read](#) (int *data, int capacity)
Reads data from a specified memory location into the [UnitySurrogateBase](#).
- abstract void [Write](#) (int *data, int capacity)
Writes data from the [UnitySurrogateBase](#) to a specified memory location.

6.102.1 Detailed Description

Represents a base class for Unity surrogates. This class is serializable and provides abstract methods for reading, writing, and initializing data.

6.102.2 Member Function Documentation

6.102.2.1 Init()

```
abstract void Init (
    int capacity ) [pure virtual]
```

Initializes the [UnitySurrogateBase](#) with a specified capacity.

Parameters

<i>capacity</i>	The capacity to initialize the UnitySurrogateBase with.
-----------------	---

Implemented in [UnityValueSurrogate< T, TReaderWriter >](#), [UnityLinkedListSurrogate< T, ReaderWriter >](#), [UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >](#), and [UnityArraySurrogate< T, ReaderWriter >](#).

6.102.2.2 Read()

```
abstract void Read (
    int * data,
    int capacity ) [pure virtual]
```

Reads data from a specified memory location into the [UnitySurrogateBase](#).

Parameters

<i>data</i>	The memory location to read from.
<i>capacity</i>	The number of elements to read.

Implements [IUnitySurrogate](#).

Implemented in [UnityValueSurrogate< T, TReaderWriter >](#), [UnityLinkedListSurrogate< T, ReaderWriter >](#), [UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >](#), and [UnityArraySurrogate< T, ReaderWriter >](#).

6.102.2.3 Write()

```
abstract void Write (
    int * data,
    int capacity ) [pure virtual]
```

Writes data from the [UnitySurrogateBase](#) to a specified memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>capacity</i>	The number of elements to write.

Implements [IUnitySurrogate](#).

Implemented in [UnityValueSurrogate< T, TReaderWriter >](#), [UnityLinkedListSurrogate< T, ReaderWriter >](#), [UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >](#), and [UnityArraySurrogate< T, ReaderWriter >](#).

6.103 UnityValueSurrogate< T, TReaderWriter > Class Template Reference

Represents a base class for Unity value surrogates. This class is serializable and provides methods for reading, writing, and initializing data.

Inherits [UnitySurrogateBase](#), and [IUnityValueSurrogate< T >](#).

Public Member Functions

- override void [Init](#) (int capacity)
Initializes the [UnityValueSurrogate](#) with a specified capacity.
- override void [Read](#) (int *data, int capacity)
Reads data into the [UnityValueSurrogate](#) from a specified memory location.
- override void [Write](#) (int *data, int capacity)
Writes data from the [UnityValueSurrogate](#) to a specified memory location.

Properties

- abstract T [DataProperty](#) [get, set]
Gets or sets the data for the [UnityValueSurrogate](#).

6.103.1 Detailed Description

Represents a base class for Unity value surrogates. This class is serializable and provides methods for reading, writing, and initializing data.

Template Parameters

<i>T</i>	The type of the data.
<i>TReaderWriter</i>	The type of the reader/writer for the data. Must be unmanaged and implement IElementReaderWriter{T} .

Type Constraints

TReaderWriter : *unmanaged*
TReaderWriter : *IElementReaderWriter*<*T*>

6.103.2 Member Function Documentation

6.103.2.1 Init()

```
override void Init (
    int capacity ) [virtual]
```

Initializes the [UnityValueSurrogate](#) with a specified capacity.

Parameters

<i>capacity</i>	The capacity to initialize the UnityValueSurrogate with.
-----------------	--

Implements [UnitySurrogateBase](#).

6.103.2.2 Read()

```
override void Read (
    int * data,
    int capacity ) [virtual]
```

Reads data into the [UnityValueSurrogate](#) from a specified memory location.

Parameters

<i>data</i>	The memory location to read from.
<i>capacity</i>	The number of elements to read.

Implements [UnitySurrogateBase](#).

6.103.2.3 Write()

```
override void Write (
    int * data,
    int capacity ) [virtual]
```

Writes data from the [UnityValueSurrogate](#) to a specified memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>capacity</i>	The number of elements to write.

Implements [UnitySurrogateBase](#).

6.103.3 Property Documentation

6.103.3.1 DataProperty

abstract T DataProperty [get], [set]

Gets or sets the data for the [UnityValueSurrogate](#).

6.104 InterpolatedErrorCorrectionSettings Class Reference

A set of parameters that tune the interpolated correction of prediction error on transform data.

Public Attributes

- Single [MaxRate](#) = 10f
- Single [MinRate](#) = 3.3f
- Single [PosBlendEnd](#) = 1f
- Single [PosBlendStart](#) = 0.25f
- Single [PosMinCorrection](#) = 0.025f
- Single [PosTeleportDistance](#) = 2f
- Single [RotBlendEnd](#) = 0.5f
- Single [RotBlendStart](#) = 0.1f
- Single [RotTeleportRadians](#) = 1.5f

6.104.1 Detailed Description

A set of parameters that tune the interpolated correction of prediction error on transform data.

6.104.2 Member Data Documentation

6.104.2.1 MaxRate

```
Single MaxRate = 10f
```

A factor with dimension of 1/s (Hz) that works as a upper limit for how much of the accumulated prediction error is corrected every frame. This factor affects both the position and the rotation correction. Suggested values are greater than [MinRate](#) and smaller than half of a target rendering rate.

E.g.: MaxRate = 15, rendering delta time = (1/60)s: at maximum 25% ($15 * 1/60$) of the accumulated error will be corrected on this rendered frame.

This threshold might not be respected if the resultant correction magnitude is below the [PosMinCorrection](#) or above the [PosTeleportDistance](#), for the position error, or above the [RotTeleportRadians](#), for the rotation error.

6.104.2.2 MinRate

```
Single MinRate = 3.3f
```

A factor with dimension of 1/s (Hz) that works as a lower limit for how much of the accumulated prediction error is corrected every frame. This factor affects both the position and the rotation correction. Suggested values are greater than zero and smaller than [MaxRate](#).

E.g.: MinRate = 3, rendering delta time = (1/60)s: at least 5% ($3 * 1/60$) of the accumulated error will be corrected on this rendered frame.

This threshold might not be respected if the resultant correction magnitude is below the [PosMinCorrection](#) or above the [PosTeleportDistance](#), for the position error, or above the [RotTeleportRadians](#), for the rotation error.

6.104.2.3 PosBlendEnd

```
Single PosBlendEnd = 1f
```

The reference for the magnitude of the accumulated position error, in meters, at which the position error will be corrected at the [MaxRate](#). Suggested values are greater than [PosBlendStart](#) and smaller than [PosTeleportDistance](#).

In other words, if the magnitude of the accumulated error is equal to or greater than this threshold, it will be corrected at the [MaxRate](#). If, instead, the magnitude is between [PosBlendStart](#) and this threshold, the error is corrected at a rate between [MinRate](#) and [MaxRate](#), proportionally. If it is equal to or smaller than [PosBlendStart](#), it will be corrected at the [MinRate](#).

Note: as the factor is expressed in distance units (meters), it might need to be scaled proportionally to the overall scale of objects in the scene and speeds at which they move, which are factors that affect the expected magnitude of prediction errors.

6.104.2.4 PosBlendStart

```
Single PosBlendStart = 0.25f
```

The reference for the magnitude of the accumulated position error, in meters, at which the position error will be corrected at the [MinRate](#). Suggested values are greater than [PosMinCorrection](#) and smaller than [PosBlendEnd](#).

In other words, if the magnitude of the accumulated error is equal to or smaller than this threshold, it will be corrected at the [MinRate](#). If, instead, the magnitude is between this threshold and [PosBlendEnd](#), the error is corrected at a rate between [MinRate](#) and [MaxRate](#), proportionally. If it is equal to or greater than [PosBlendEnd](#), it will be corrected at the [MaxRate](#).

Note: as the factor is expressed in distance units (meters), it might need to be scaled proportionally to the overall scale of objects in the scene and speeds at which they move, which are factors that affect the expected magnitude of prediction errors.

6.104.2.5 PosMinCorrection

```
Single PosMinCorrection = 0.025f
```

The value, in meters, that represents the minimum magnitude of the accumulated position error that will be corrected in a single frame, until it is fully corrected.

This setting has priority over the resultant correction rate, i.e. the restriction will be respected even if it makes the effective correction rate be different than the one computed according to the min/max rates and start/end blend values. Suggested values are greater than zero and smaller than [PosBlendStart](#).

Note: as the factor is expressed in distance units (meters), it might need to be scaled proportionally to the overall scale of objects in the scene and speeds at which they move, which are factors that affect the expected magnitude of prediction errors.

6.104.2.6 PosTeleportDistance

```
Single PosTeleportDistance = 2f
```

The value, in meters, that represents the magnitude of the accumulated position error above which the error will be instantaneously corrected, effectively teleporting the rendered object to its correct position. Suggested values are greater than [PosBlendEnd](#).

This setting has priority over the resultant correction rate, i.e. the restriction will be respected even if it makes the effective correction rate be different than the one computed according to the min/max rates and start/end blend values.

Note: as the factor is expressed in distance units (meters), it might need to be scaled proportionally to the overall scale of objects in the scene and speeds at which they move, which are factors that affect the expected magnitude of prediction errors.

6.104.2.7 RotBlendEnd

```
Single RotBlendEnd = 0.5f
```

The reference for the magnitude of the accumulated rotation error, in radians, at which the rotation error will be corrected at the [MaxRate](#). Suggested values are greater than [RotBlendStart](#) and smaller than [RotTeleportRadians](#).

In other words, if the magnitude of the accumulated error is equal to or greater than this threshold, it will be corrected at the [MaxRate](#). If, instead, the magnitude is between [RotBlendStart](#) and this threshold, the error is corrected at a rate between [MinRate](#) and [MaxRate](#), proportionally. If it is equal to or smaller than [RotBlendStart](#), it will be corrected at the [MinRate](#).

6.104.2.8 RotBlendStart

```
Single RotBlendStart = 0.1f
```

The reference for the magnitude of the accumulated rotation error, in radians, at which the rotation error will be corrected at the [MinRate](#). Suggested values are smaller than [RotBlendEnd](#).

In other words, if the magnitude of the accumulated error is equal to or smaller than this threshold, it will be corrected at the [MinRate](#). If, instead, the magnitude is between this threshold and [RotBlendEnd](#), the error is corrected at a rate between [MinRate](#) and [MaxRate](#), proportionally. If it is equal to or greater than [RotBlendEnd](#), it will be corrected at the [MaxRate](#).

6.104.2.9 RotTeleportRadians

```
Single RotTeleportRadians = 1.5f
```

The value, in radians, that represents the magnitude of the accumulated rotation error above which the error will be instantaneously corrected, effectively teleporting the rendered object to its correct orientation. Suggested values are greater than [RotBlendEnd](#).

This setting has priority over the resultant correction rate, i.e. the restriction will be respected even if it makes the effective correction rate be different than the one computed according to the min/max rates and start/end blend values.

6.105 IPlayerJoined Interface Reference

Interface for handling the event when a player joins the game.

Public Member Functions

- void [PlayerJoined](#) ([PlayerRef](#) player)
Method to be called when a player joins the game.

6.105.1 Detailed Description

Interface for handling the event when a player joins the game.

6.105.2 Member Function Documentation

6.105.2.1 PlayerJoined()

```
void PlayerJoined (
    PlayerRef player )
```

Method to be called when a player joins the game.

Parameters

<i>player</i>	The player who joined the game.
---------------	---------------------------------

6.106 IPlayerLeft Interface Reference

Interface for handling the event when a player leaves the game.

Public Member Functions

- void [PlayerLeft \(PlayerRef player\)](#)
Method to be called when a player leaves the game.

6.106.1 Detailed Description

Interface for handling the event when a player leaves the game.

6.106.2 Member Function Documentation

6.106.2.1 PlayerLeft()

```
void PlayerLeft (
    PlayerRef player )
```

Method to be called when a player leaves the game.

Parameters

<i>player</i>	The player who left the game.
---------------	-------------------------------

6.107 IRemotePrefabCreated Interface Reference

Interface for handling the event when a remote prefab is created.

Public Member Functions

- void [RemotePrefabCreated \(\)](#)
Method to be called when a remote prefab is created.

6.107.1 Detailed Description

Interface for handling the event when a remote prefab is created.

6.107.2 Member Function Documentation

6.107.2.1 RemotePrefabCreated()

```
void RemotePrefabCreated ( )
```

Method to be called when a remote prefab is created.

6.108 ISceneLoadDone Interface Reference

Interface for handling the event when a scene load operation is completed.

Public Member Functions

- void [SceneLoadDone](#) (in [SceneLoadDoneArgs](#) sceneInfo)
Method to be called when a scene load operation is completed.

6.108.1 Detailed Description

Interface for handling the event when a scene load operation is completed.

6.108.2 Member Function Documentation

6.108.2.1 SceneLoadDone()

```
void SceneLoadDone (
    in SceneLoadDoneArgs sceneInfo )
```

Method to be called when a scene load operation is completed.

Parameters

<code>sceneInfo</code>	The information about the loaded scene.
------------------------	---

6.109 ISceneLoadStart Interface Reference

Interface for handling the event when a scene load operation is started.

Public Member Functions

- void [SceneLoadStart](#) ([SceneRef](#) sceneRef)
Method to be called when a scene load operation is started.

6.109.1 Detailed Description

Interface for handling the event when a scene load operation is started.

6.109.2 Member Function Documentation

6.109.2.1 SceneLoadStart()

```
void SceneLoadStart (
    SceneRef sceneRef )
```

Method to be called when a scene load operation is started.

Parameters

sceneRef	Reference to the scene that is being loaded.
----------	--

6.110 ISimulationEnter Interface Reference

Interface for [SimulationEnter](#) callback. Called when the [NetworkObject](#) joins AreaOfInterest. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Public Member Functions

- void [SimulationEnter](#) ()

Called when the [NetworkObject](#) joins AreaOfInterest. Object is now receiving snapshot updates. Object will execute [NetworkBehaviour FixedUpdateNetwork\(\)](#) and [Render\(\)](#) methods until the object leaves simulation.

6.110.1 Detailed Description

Interface for [SimulationEnter](#) callback. Called when the [NetworkObject](#) joins AreaOfInterest. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.110.2 Member Function Documentation

6.110.2.1 SimulationEnter()

```
void SimulationEnter ( )
```

Called when the [NetworkObject](#) joins AreaOfInterest. Object is now receiving snapshot updates. Object will execute [NetworkBehaviour FixedUpdateNetwork\(\)](#) and [Render\(\)](#) methods until the object leaves simulation.

6.111 ISimulationExit Interface Reference

Interface for the [SimulationExit](#) callback. Called when the [NetworkObject](#) leaves AreaOfInterest. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

Public Member Functions

- void [SimulationExit](#) ()

Called when the [NetworkObject](#) leaves AreaOfInterest. Object is no longer receiving snapshot updates. Object will stop executing [NetworkBehaviour](#) FixedUpdateNetwork() and Render() methods until the object rejoins simulation.

6.111.1 Detailed Description

Interface for the [SimulationExit](#) callback. Called when the [NetworkObject](#) leaves AreaOfInterest. Implement this interface on [SimulationBehaviour](#) and [NetworkBehaviour](#) classes.

6.111.2 Member Function Documentation

6.111.2.1 SimulationExit()

```
void SimulationExit ( )
```

Called when the [NetworkObject](#) leaves AreaOfInterest. Object is no longer receiving snapshot updates. Object will stop executing [NetworkBehaviour](#) FixedUpdateNetwork() and Render() methods until the object rejoins simulation.

6.112 ISpawned Interface Reference

Interface for handling the event when an object is spawned.

Inherited by [HitboxManager](#), and [NetworkBehaviour](#).

Public Member Functions

- void [Spawned](#) ()

Method to be called when an object is spawned.

6.112.1 Detailed Description

Interface for handling the event when an object is spawned.

6.112.2 Member Function Documentation

6.112.2.1 Spawned()

```
void Spawned ( )
```

Method to be called when an object is spawned.

Implemented in [NetworkTransform](#), [NetworkMecanimAnimator](#), and [NetworkBehaviour](#).

6.113 IStateAuthorityChanged Interface Reference

Interface for handling the event when the state authority changes.

Public Member Functions

- void [StateAuthorityChanged \(\)](#)
Method to be called when the state authority changes.

6.113.1 Detailed Description

Interface for handling the event when the state authority changes.

6.113.2 Member Function Documentation

6.113.2.1 StateAuthorityChanged()

```
void StateAuthorityChanged ( )
```

Method to be called when the state authority changes.

6.114 LagCompensatedHit Struct Reference

Defines a lag compensated query hit result.

Static Public Member Functions

- static [operator LagCompensatedHit](#) (RaycastHit raycastHit)
Creates a [LagCompensatedHit](#) structure from the information on a Unity RaycastHit.
- static [operator LagCompensatedHit](#) (RaycastHit2D raycastHit2D)
Creates a [LagCompensatedHit](#) structure from the information on a Unity RaycastHit2D.

Public Attributes

- Collider [Collider](#)
PhysX collider hit. Null in case hit is a [Fusion Hitbox](#) or a Box2D hit.
- Collider2D [Collider2D](#)
Box2D collider hit. Null in case hit is a [Fusion Hitbox](#) or a PhysX hit.
- float [Distance](#)
Distance (if requested) to hit, at the lag compensated time.
- GameObject [GameObject](#)
The Unity Game Object that was hit. Its data is not lag compensated. This is either the [Hitbox](#)'s or the [Collider](#)'s [gameObject](#), depending on the object hit being a lag-compensated [Hitbox](#) or a regular Unity collider, respectively.
- Hitbox [Hitbox](#)
Fusion's [Hitbox](#). Null in case the hit was on PhysX or Box2D.
- Vector3 [HitboxColliderPosition](#)
The hitbox collider position on the snapshot.
- Quaternion [HitboxColliderRotation](#)
The hitbox collider rotation on the snapshot.
- Vector3 [Normal](#)
Surface normal (if requested) of the hit, at the lag compensated time.
- Vector3 [Point](#)
Point of impact of the hit, at the lag compensated time.
- HitType [Type](#)
Hit object source (PhysX or [Fusion](#) Hitboxes).

6.114.1 Detailed Description

Defines a lag compensated query hit result.

6.114.2 Member Function Documentation

6.114.2.1 operator LagCompensatedHit() [1/2]

```
static operator LagCompensatedHit (
    RaycastHit raycastHit ) [explicit], [static]
```

Creates a [LagCompensatedHit](#) structure from the information on a Unity RaycastHit.

Parameters

<i>raycastHit</i>	The RaycastHit used as source.
-------------------	--------------------------------

Returns

The built [LagCompensatedHit](#) structure.

6.114.2.2 operator LagCompensatedHit() [2/2]

```
static operator LagCompensatedHit (
    RaycastHit2D raycastHit2D ) [explicit], [static]
```

Creates a [LagCompensatedHit](#) structure from the information on a Unity RaycastHit2D.

Parameters

<i>raycastHit2D</i>	The RaycastHit2D used as source.
---------------------	----------------------------------

Returns

The built [LagCompensatedHit](#) structure.

6.114.3 Member Data Documentation**6.114.3.1 Collider**

`Collider Collider`

PhysX collider hit. Null in case hit is a [Fusion Hitbox](#) or a Box2D hit.

6.114.3.2 Collider2D

`Collider2D Collider2D`

Box2D collider hit. Null in case hit is a [Fusion Hitbox](#) or a PhysX hit.

6.114.3.3 Distance

```
float Distance
```

Distance (if requested) to hit, at the lag compensated time.

6.114.3.4 GameObject

```
GameObject GameObject
```

The Unity Game Object that was hit. Its data is not lag compensated. This is either the [Hitbox](#)'s or the [Collider](#)'s `gameObject`, depending on the object hit being a lag-compensated [Hitbox](#) or a regular Unity collider, respectively.

6.114.3.5 Hitbox

```
Hitbox Hitbox
```

Fusion's [Hitbox](#). Null in case the hit was on PhysX or Box2D.

6.114.3.6 HitboxColliderPosition

```
Vector3 HitboxColliderPosition
```

The hitbox collider position on the snapshot.

6.114.3.7 HitboxColliderRotation

```
Quaternion HitboxColliderRotation
```

The hitbox collider rotation on the snapshot.

6.114.3.8 Normal

```
Vector3 Normal
```

Surface normal (if requested) of the hit, at the lag compensated time.

6.114.3.9 Point

`Vector3 Point`

Point of impact of the hit, at the lag compensated time.

6.114.3.10 Type

`HitType Type`

Hit object source (PhysX or Fusion Hitboxes).

6.115 AABB Struct Reference

Represents an Axis-Aligned Bounding Box ([AABB](#)).

Public Member Functions

- [`AABB \(Bounds bounds\)`](#)
Constructs an [AABB](#) from a Unity `Bounds` object.
- [`AABB \(Vector3 center, Vector3 extents\)`](#)
Constructs an [AABB](#) from a center point and extents.

Public Attributes

- `readonly Vector3 Center`
Represents the center of the [AABB](#).
- `readonly Vector3 Extents`
Represents the extents (half-widths) of the [AABB](#).
- `readonly Vector3 Max`
Represents the maximum point (upper corner) of the [AABB](#).
- `readonly Vector3 Min`
Represents the minimum point (lower corner) of the [AABB](#).

6.115.1 Detailed Description

Represents an Axis-Aligned Bounding Box ([AABB](#)).

6.115.2 Constructor & Destructor Documentation

6.115.2.1 `AABB()` [1/2]

```
AABB (
    Bounds bounds )
```

Constructs an [AABB](#) from a Unity `Bounds` object.

Parameters

<i>bounds</i>	The Unity Bounds object to construct the AABB from.
---------------	---

6.115.2.2 [AABB\(\)](#) [2/2]

```
AABB (
    Vector3 center,
    Vector3 extents )
```

Constructs an [AABB](#) from a center point and extents.

Parameters

<i>center</i>	The center point of the AABB .
<i>extents</i>	The extents (half-widths) of the AABB .

6.115.3 Member Data Documentation**6.115.3.1 Center**

```
readonly Vector3 Center
```

Represents the center of the [AABB](#).

6.115.3.2 Extents

```
readonly Vector3 Extents
```

Represents the extents (half-widths) of the [AABB](#).

6.115.3.3 Max

```
readonly Vector3 Max
```

Represents the maximum point (upper corner) of the [AABB](#).

6.115.3.4 Min

readonly Vector3 Min

Represents the minimum point (lower corner) of the [AABB](#).

6.116 BoxOverlapQuery Class Reference

Class that represents a box overlap query. Used to query against the [NetworkRunner.LagCompensation API](#).

Inherits [Query](#).

Public Member Functions

- [BoxOverlapQuery](#) (ref [BoxOverlapQueryParams](#) boxOverlapParams)
Create a new [BoxOverlapQuery](#) with the given boxOverlapParams .
- [BoxOverlapQuery](#) (ref [BoxOverlapQueryParams](#) boxOverlapParams, Collider[] physXOverlapHitsCache, Collider2D[] box2DOverlapHitsCache)
Create a new [BoxOverlapQuery](#) with the given boxOverlapParams . The result colliders arrays can be provided to avoid allocation.

Public Attributes

- [Vector3 Center](#)
The box query center.
- [Vector3 Extents](#)
The box query extents.
- [Quaternion Rotation](#)
The box query rotation.

Protected Member Functions

- override bool [Check](#) (ref [AABB](#) bounds)
Check if the given bounds overlaps with this query.

6.116.1 Detailed Description

Class that represents a box overlap query. Used to query against the [NetworkRunner.LagCompensation API](#).

6.116.2 Constructor & Destructor Documentation

6.116.2.1 BoxOverlapQuery() [1/2]

```
BoxOverlapQuery (
    ref BoxOverlapQueryParams boxOverlapParams )
```

Create a new [BoxOverlapQuery](#) with the given boxOverlapParams .

Parameters

<i>boxOverlapParams</i>	The parameters to be used when creating the query.
-------------------------	--

6.116.2.2 BoxOverlapQuery() [2/2]

```
BoxOverlapQuery (
    ref BoxOverlapQueryParams boxOverlapParams,
    Collider[ ] physXOverlapHitsCache,
    Collider2D[ ] box2DOverlapHitsCache )
```

Create a new [BoxOverlapQuery](#) with the given *boxOverlapParams*. The result colliders arrays can be provided to avoid allocation.

Parameters

<i>boxOverlapParams</i>	The parameters to be used when creating the query.
<i>physXOverlapHitsCache</i>	Array to write the results of the PhysX query if used.
<i>box2DOverlapHitsCache</i>	Array to write the results of the Box2D query if used.

6.116.3 Member Function Documentation**6.116.3.1 Check()**

```
override bool Check (
    ref AABB bounds ) [protected], [virtual]
```

Check if the given *bounds* overlaps with this query.

Parameters

<i>bounds</i>	The bounds to check.
---------------	----------------------

Returns

True if the bounds overlaps with this query, false otherwise.

Implements [Query](#).

6.116.4 Member Data Documentation

6.116.4.1 Center

Vector3 Center

The box query center.

6.116.4.2 Extents

Vector3 Extents

The box query extents.

6.116.4.3 Rotation

Quaternion Rotation

The box query rotation.

6.117 BoxOverlapQueryParams Struct Reference

Base parameters needed to execute a box overlap query

Public Member Functions

- [BoxOverlapQueryParams](#) (QueryParams queryParams, Vector3 center, Vector3 extents, Quaternion rotation, int staticHitsCapacity)
Create a new BoxOverlapQueryParams

Public Attributes

- Vector3 [Center](#)
Represents the center of the box for the overlap query.
- Vector3 [Extents](#)
Represents the extents of the box for the overlap query.
- QueryParams [queryParams](#)
Represents the base parameters for the query.
- Quaternion [Rotation](#)
Represents the rotation of the box for the overlap query.
- int [StaticHitsCapacity](#)
Represents the capacity for the cached PhysX and Box2D static hits.

6.117.1 Detailed Description

Base parameters needed to execute a box overlap query

6.117.2 Constructor & Destructor Documentation

6.117.2.1 BoxOverlapQueryParams()

```
BoxOverlapQueryParams (
    QueryParams queryParams,
    Vector3 center,
    Vector3 extents,
    Quaternion rotation,
    int staticHitsCapacity )
```

Create a new BoxOverlapQueryParams

Parameters

<i>queryParams</i>	Parameters to be used
<i>center</i>	The query center
<i>extents</i>	The query extents
<i>rotation</i>	The query rotation
<i>staticHitsCapacity</i>	Capacity for the cached PhysX and Box2D static hits.

6.117.3 Member Data Documentation

6.117.3.1 Center

Vector3 Center

Represents the center of the box for the overlap query.

6.117.3.2 Extents

Vector3 Extents

Represents the extents of the box for the overlap query.

6.117.3.3 QueryParams

`QueryParams` `QueryParams`

Represents the base parameters for the query.

6.117.3.4 Rotation

`Quaternion` `Rotation`

Represents the rotation of the box for the overlap query.

6.117.3.5 StaticHitsCapacity

`int` `StaticHitsCapacity`

Represents the capacity for the cached PhysX and Box2D static hits.

6.118 BVHDraw Class Reference

Provide a way to iterate over BVH and return a [BVHNodeDrawInfo](#) for each node.

Inherits `IEnumerable< BVHNodeDrawInfo >`.

Public Member Functions

- `IEnumerator< BVHNodeDrawInfo > GetEnumerator ()`
Returns an enumerator that iterates through the [BVHNodeDrawInfo](#).

6.118.1 Detailed Description

Provide a way to iterate over BVH and return a [BVHNodeDrawInfo](#) for each node.

6.118.2 Member Function Documentation

6.118.2.1 GetEnumerator()

`IEnumerator<BVHNodeDrawInfo>` `GetEnumerator ()`

Returns an enumerator that iterates through the [BVHNodeDrawInfo](#).

6.119 BVHNodeDrawInfo Class Reference

Container class to provide the necessary info to draw nodes from the BVH

Properties

- **Bounds** `Bounds [get]`
Get the node Bounds
- **int Depth** `[get]`
Get the node depth on the BVH
- **int MaxDepth** `[get]`
Get the BVH max depth

6.119.1 Detailed Description

Container class to provide the necessary info to draw nodes from the BVH

6.119.2 Property Documentation

6.119.2.1 Bounds

`Bounds Bounds [get]`

Get the node Bounds

6.119.2.2 Depth

`int Depth [get]`

Get the node depth on the BVH

6.119.2.3 MaxDepth

`int MaxDepth [get]`

Get the BVH max depth

6.120 ColliderDrawInfo Class Reference

Container class to provide the necessary information to draw a hitbox collider

Properties

- `Vector3 BoxExtents [get]`
The box extends of the collider Used on [HitboxTypes](#) of types: Box
- `Vector3 CapsuleBottomCenter [get]`
Represents the bottom center position of the capsule collider.
- `float CapsuleExtents [get]`
The height for capsule colliders.
See also
[HitboxTypes](#)
- `Vector3 CapsuleTopCenter [get]`
Represents the top center position of the capsule collider.
- `Matrix4x4 LocalToWorldMatrix [get]`
The local to world matrix of the collider.
- `Vector3 Offset [get]`
The offset of the collider.
- `float Radius [get]`
The radius of the collider. Used on [HitboxTypes](#) of types: Sphere and Capsule.
- `HitboxTypes Type [get]`
The [HitboxTypes](#) of the collider.

6.120.1 Detailed Description

Container class to provide the necessary information to draw a hitbox collider

6.120.2 Property Documentation

6.120.2.1 BoxExtents

`Vector3 BoxExtents [get]`

The box extends of the collider Used on [HitboxTypes](#) of types: Box

6.120.2.2 CapsuleBottomCenter

`Vector3 CapsuleBottomCenter [get]`

Represents the bottom center position of the capsule collider.

6.120.2.3 CapsuleExtents

```
float CapsuleExtents [get]
```

The height for capsule colliders.

See also

[HitboxTypes](#)

6.120.2.4 CapsuleTopCenter

```
Vector3 CapsuleTopCenter [get]
```

Represents the top center position of the capsule collider.

6.120.2.5 LocalToWorldMatrix

```
Matrix4x4 LocalToWorldMatrix [get]
```

The local to world matrix of the collider.

6.120.2.6 Offset

```
Vector3 Offset [get]
```

The offset of the collider.

6.120.2.7 Radius

```
float Radius [get]
```

The radius of the collider. Used on [HitboxTypes](#) of types: Sphere and Capsule.

6.120.2.8 Type

```
HitboxTypes Type [get]
```

The [HitboxTypes](#) of the collider.

6.121 HitboxColliderContainerDraw Class Reference

Provide a way to iterate over the HitboxBuffer.HitboxSnapshot and return the [ColliderDrawInfo](#) for each collider on the snapshot.

Inherits [IEnumerable< ColliderDrawInfo >](#).

Public Member Functions

- [IEnumerator< ColliderDrawInfo > GetEnumerator \(\)](#)
Returns an enumerator that iterates through the [ColliderDrawInfo](#) of this container.

6.121.1 Detailed Description

Provide a way to iterate over the HitboxBuffer.HitboxSnapshot and return the [ColliderDrawInfo](#) for each collider on the snapshot.

6.121.2 Member Function Documentation

6.121.2.1 GetEnumerator()

```
IEnumerator<ColliderDrawInfo> GetEnumerator ( )
```

Returns an enumerator that iterates through the [ColliderDrawInfo](#) of this container.

6.122 LagCompensatedExt Class Reference

LagCompensated Extension methods

Static Public Member Functions

- static void [SortDistance](#) (this List< [LagCompensatedHit](#) > hits)
Sorts all hits in ascending order of [LagCompensatedHit.Distance](#).
- static void [SortReference](#) (this List< [LagCompensatedHit](#) > hits, Vector3 reference)
Sorts all hits in ascending order of distance from [LagCompensatedHit.Point](#) to the reference point.

6.122.1 Detailed Description

LagCompensated Extension methods

6.122.2 Member Function Documentation

6.122.2.1 SortDistance()

```
static void SortDistance (
    this List< LagCompensatedHit > hits ) [static]
```

Sorts all *hits* in ascending order of [LagCompensatedHit.Distance](#).

Parameters

<i>hits</i>	List containing hits to be sorted.
-------------	------------------------------------

Exceptions

<i>NullReferenceException</i>	If <i>hits</i> are null.
-------------------------------	--------------------------

6.122.2.2 SortReference()

```
static void SortReference (
    this List< LagCompensatedHit > hits,
    Vector3 reference ) [static]
```

Sorts all *hits* in ascending order of distance from [LagCompensatedHit.Point](#) to the *reference* point.

Parameters

<i>hits</i>	List containing hits to be sorted.
<i>reference</i>	Used as reference point to compute distance from hit points.

Exceptions

<i>NullReferenceException</i>	If <i>hits</i> are null.
-------------------------------	--------------------------

6.123 LagCompensationDraw Class Reference

Provide access to iterate over the lag compensation system components and give the necessary information to draw them.

Static Public Member Functions

- static void [GizmosDrawWireCapsule](#) (Vector3 topCenter, Vector3 bottomCenter, float capsuleRadius)
Method to draw capsules out of simple shapes.

Public Attributes

- [BVHDraw](#) [BVHDraw](#)
Iterate over to get the BVH node draw data.
- [SnapshotHistoryDraw](#) [SnapshotHistoryDraw](#)
Iterate over to get the hitbox snapshots draw data. Iterate the received hitbox snapshot draw data to get all the colliders draw info for that snapshot.

6.123.1 Detailed Description

Provide access to iterate over the lag compensation system components and give the necessary information to draw them.

6.123.2 Member Function Documentation

6.123.2.1 `GizmosDrawWireCapsule()`

```
static void GizmosDrawWireCapsule (
    Vector3 topCenter,
    Vector3 bottomCenter,
    float capsuleRadius ) [static]
```

Method to draw capsules out of simple shapes.

Parameters

<i>topCenter</i>	The top capsule end position
<i>bottomCenter</i>	The bottom capsule end position
<i>capsuleRadius</i>	The capsule radius

6.123.3 Member Data Documentation

6.123.3.1 `BVHDraw`

`BVHDraw BVHDraw`

Iterate over to get the BVH node draw data.

6.123.3.2 `SnapshotHistoryDraw`

`SnapshotHistoryDraw SnapshotHistoryDraw`

Iterate over to get the hitbox snapshots draw data. Iterate the received hitbox snapshot draw data to get all the colliders draw info for that snapshot.

6.124 LagCompensationUtils.ContactData Struct Reference

Details regarding a shape intersection. It does not carry information about the intersection happening or not.

Public Attributes

- `Vector3 Normal`
Vector that described the plane of smallest penetration between the shapes.
- `float Penetration`
Penetration along the normal plane.
- `Vector3 Point`
Contact point.

6.124.1 Detailed Description

Details regarding a shape intersection. It does not carry information about the intersection happening or not.

6.124.2 Member Data Documentation

6.124.2.1 Normal

`Vector3 Normal`

Vector that described the plane of smallest penetration between the shapes.

6.124.2.2 Penetration

`float Penetration`

Penetration along the normal plane.

6.124.2.3 Point

`Vector3 Point`

Contact point.

6.125 PositionRotationQueryParams Struct Reference

[Query](#) parameters for position rotation query

Public Member Functions

- [PositionRotationQueryParams \(QueryParams queryParams, Hitbox hitbox\)](#)
Create a new PositionRotationQueryParams.

Public Attributes

- [Hitbox Hitbox](#)
Represents the hitbox to be queried.
- [queryParams queryParams](#)
Represents the base parameters for the query.

6.125.1 Detailed Description

[Query](#) parameters for position rotation query

6.125.2 Constructor & Destructor Documentation

6.125.2.1 PositionRotationQueryParams()

```
PositionRotationQueryParams (
    queryParams queryParams,
    Hitbox hitbox )
```

Create a new PositionRotationQueryParams.

Parameters

<code>queryParams</code>	Parameters to be used
<code>hitbox</code>	The hitbox to be queried

6.125.3 Member Data Documentation

6.125.3.1 Hitbox

[Hitbox Hitbox](#)

Represents the hitbox to be queried.

6.125.3.2 QueryParams

`QueryParams` `QueryParams`

Represents the base parameters for the query.

6.126 Query Class Reference

Base class for all Lag Compensation queries

Inherits `IBoundsTraversalTest`.

Inherited by `BoxOverlapQuery`, `RaycastQuery`, and `SphereOverlapQuery`.

Public Attributes

- float? `Alpha`
Represents the interpolation factor between the current and next simulation tick.
- `LayerMask` `LayerMask`
Represents the layer mask to selectively ignore colliders when performing the query.
- `HitOptions` `Options`
Represents the options for the hit detection of the query.
- `PlayerRef` `Player`
Represents the player who initiated the query.
- `PreProcessingDelegate` `PreProcessingDelegate`
Represents the delegate to be called for pre-processing before the query is performed.
- int? `Tick`
Represents the simulation tick at which the query was initiated.
- int? `TickCount`
Represents the simulation tick to which the query is performed.
- `QueryTriggerInteraction` `TriggerInteraction`
Represents the interaction type of the query with triggers.
- void * `UserArgs`
Represents the user arguments for the query.

Protected Member Functions

- abstract bool `Check` (ref `AABB` bounds)
Checks if the provided bounds should be included in the query.
- `Query` (ref `QueryParams` `queryParams`)
Initializes a new instance of the `Query` class using the provided `QueryParams`.

6.126.1 Detailed Description

Base class for all Lag Compensation queries

6.126.2 Constructor & Destructor Documentation

6.126.2.1 Query()

```
Query (
    ref QueryParams qParams ) [protected]
```

Initializes a new instance of the [Query](#) class using the provided [QueryParams](#).

Parameters

<i>queryParams</i>	The QueryParams to use for initializing the Query .
--------------------	---

6.126.3 Member Function Documentation

6.126.3.1 Check()

```
abstract bool Check (
    ref AABB bounds ) [protected], [pure virtual]
```

Checks if the provided bounds should be included in the query.

Parameters

<i>bounds</i>	The bounds to check.
---------------	----------------------

Returns

True if the bounds should be included in the query, false otherwise.

Implemented in [SphereOverlapQuery](#), [RaycastQuery](#), and [BoxOverlapQuery](#).

6.126.4 Member Data Documentation

6.126.4.1 Alpha

```
float? Alpha
```

Represents the interpolation factor between the current and next simulation tick.

6.126.4.2 LayerMask

```
LayerMask LayerMask
```

Represents the layer mask to selectively ignore colliders when performing the query.

6.126.4.3 Options

`HitOptions` Options

Represents the options for the hit detection of the query.

6.126.4.4 Player

`PlayerRef` Player

Represents the player who initiated the query.

6.126.4.5 PreProcessingDelegate

`PreProcessingDelegate` PreProcessingDelegate

Represents the delegate to be called for pre-processing before the query is performed.

6.126.4.6 Tick

`int?` `Tick`

Represents the simulation tick at which the query was initiated.

6.126.4.7 TickTo

`int?` `TickTo`

Represents the simulation tick to which the query is performed.

6.126.4.8 TriggerInteraction

`QueryTriggerInteraction` TriggerInteraction

Represents the interaction type of the query with triggers.

6.126.4.9 UserArgs

```
void* UserArgs
```

Represents the user arguments for the query.

6.127 QueryParams Struct Reference

Base parameters needed to execute a query.

Public Attributes

- float? [Alpha](#)
Represents the interpolation factor between the current and next simulation tick.
- LayerMask [LayerMask](#)
Represents the layer mask to selectively ignore colliders when performing the query.
- [HitOptions Options](#)
Represents the options for the hit detection of the query.
- [PlayerRef Player](#)
Represents the player who initiated the query.
- PreProcessingDelegate [PreProcessingDelegate](#)
Represents the delegate to be called for pre-processing before the query is performed.
- int [Tick](#)
Represents the simulation tick at which the query was initiated.
- int? [TickTo](#)
Represents the simulation tick to which the query is performed.
- [QueryTriggerInteraction TriggerInteraction](#)
Represents the interaction type of the query with triggers.
- void * [UserArgs](#)
Represents the user arguments for the query.

6.127.1 Detailed Description

Base parameters needed to execute a query.

6.127.2 Member Data Documentation

6.127.2.1 Alpha

```
float? Alpha
```

Represents the interpolation factor between the current and next simulation tick.

6.127.2.2 LayerMask

`LayerMask` `LayerMask`

Represents the layer mask to selectively ignore colliders when performing the query.

6.127.2.3 Options

`HitOptions` `Options`

Represents the options for the hit detection of the query.

6.127.2.4 Player

`PlayerRef` `Player`

Represents the player who initiated the query.

6.127.2.5 PreProcessingDelegate

`PreProcessingDelegate` `PreProcessingDelegate`

Represents the delegate to be called for pre-processing before the query is performed.

6.127.2.6 Tick

`int` `Tick`

Represents the simulation tick at which the query was initiated.

6.127.2.7 TickTo

`int?` `TickTo`

Represents the simulation tick to which the query is performed.

6.127.2.8 TriggerInteraction

```
QueryTriggerInteraction TriggerInteraction
```

Represents the interaction type of the query with triggers.

6.127.2.9 UserArgs

```
void* UserArgs
```

Represents the user arguments for the query.

6.128 RaycastAllQuery Class Reference

Class that represents a raycast all query. Used to query against the [NetworkRunner.LagCompensation](#) API.

Inherits [RaycastQuery](#).

Public Member Functions

- [RaycastAllQuery](#) (ref [RaycastQueryParams](#) raycastQueryParams)
Create a new [RaycastAllQuery](#) with the given [RaycastQueryParams](#).
- [RaycastAllQuery](#) (ref [RaycastQueryParams](#) raycastQueryParams, [RaycastHit\[\]](#) physXRaycastHitsCache, [RaycastHit2D\[\]](#) box2DRaycastHitCache)
Create a new [RaycastAllQuery](#) with the given [RaycastQueryParams](#). The result colliders arrays can be provided to avoid allocation.

Additional Inherited Members

6.128.1 Detailed Description

Class that represents a raycast all query. Used to query against the [NetworkRunner.LagCompensation](#) API.

6.128.2 Constructor & Destructor Documentation

6.128.2.1 RaycastAllQuery() [1/2]

```
RaycastAllQuery (
    ref RaycastQueryParams raycastQueryParams )
```

Create a new [RaycastAllQuery](#) with the given [RaycastQueryParams](#).

Parameters

<code>raycastQueryParams</code>	The parameters to be used when creating the query.
---------------------------------	--

6.128.2.2 RaycastAllQuery() [2/2]

```
RaycastAllQuery (
    ref RaycastQueryParams raycastQueryParams,
    RaycastHit[] physXRaycastHitsCache,
    RaycastHit2D[] box2DRaycastHitCache )
```

Create a new [RaycastAllQuery](#) with the given [RaycastQueryParams](#). The result colliders arrays can be provided to avoid allocation.

Parameters

<code>raycastQueryParams</code>	The parameters to be used when creating the query.
<code>physXRaycastHitsCache</code>	Array to write the results of the PhysX query if used.
<code>box2DRaycastHitCache</code>	Array to write the results of the Box2D query if used.

6.129 RaycastQuery Class Reference

Class that represents a raycast query. Used to query against the [NetworkRunner.LagCompensation](#) API.

Inherits [Query](#).

Inherited by [RaycastAllQuery](#).

Public Member Functions

- [RaycastQuery](#) (ref [RaycastQueryParams](#) raycastQueryParams)
Create a new [RaycastQuery](#) with the given [RaycastQueryParams](#)

Public Attributes

- [Vector3 Direction](#)
Represents the direction of the raycast for the query.
- float [Length](#)
Represents the maximum length of the raycast for the query.
- [Vector3 Origin](#)
Represents the origin point of the raycast for the query.

Protected Member Functions

- override bool [Check](#) (ref [AABB](#) bounds)

Check if the provided bounds should be included in the query.

6.129.1 Detailed Description

Class that represents a raycast query. Used to query against the [NetworkRunner.LagCompensation](#) API.

6.129.2 Constructor & Destructor Documentation

6.129.2.1 RaycastQuery()

```
RaycastQuery (
    ref RaycastQueryParams raycastQueryParams )
```

Create a new [RaycastQuery](#) with the given [RaycastQueryParams](#)

Parameters

<i>raycastQueryParams</i>	The parameters to be used when creating the query.
---------------------------	--

6.129.3 Member Function Documentation

6.129.3.1 Check()

```
override bool Check (
    ref AABB bounds ) [protected], [virtual]
```

Check if the provided bounds should be included in the query.

Parameters

<i>bounds</i>	The bounds to check.
---------------	----------------------

Returns

True if the bounds should be included in the query, false otherwise.

Implements [Query](#).

6.129.4 Member Data Documentation

6.129.4.1 Direction

Vector3 **Direction**

Represents the direction of the raycast for the query.

6.129.4.2 Length

float **Length**

Represents the maximum length of the raycast for the query.

6.129.4.3 Origin

Vector3 **Origin**

Represents the origin point of the raycast for the query.

6.130 RaycastQueryParams Struct Reference

Base parameters needed to execute a raycast query

Public Member Functions

- [RaycastQueryParams](#) ([QueryParams](#) queryParams, Vector3 origin, Vector3 direction, float length, int staticHitsCapacity=64)
Create a new RaycastQueryParams

Public Attributes

- Vector3 **Direction**
Represents the direction of the raycast for the query.
- float **Length**
Represents the maximum length of the raycast for the query.
- Vector3 **Origin**
Represents the origin point of the raycast for the query.
- [QueryParams](#) **queryParams**
Represents the base parameters for the raycast query.
- int **StaticHitsCapacity**
Represents the capacity for the cached PhysX and Box2D static hits.

6.130.1 Detailed Description

Base parameters needed to execute a raycast query

6.130.2 Constructor & Destructor Documentation

6.130.2.1 RaycastQueryParams()

```
RaycastQueryParams (
    QueryParams queryParams,
    Vector3 origin,
    Vector3 direction,
    float length,
    int staticHitsCapacity = 64 )
```

Create a new RaycastQueryParams

Parameters

<i>queryParams</i>	Parameters to be used
<i>origin</i>	The raycast origin
<i>direction</i>	The raycast direction
<i>length</i>	The raycast max length
<i>staticHitsCapacity</i>	Capacity for the cached PhysX and Box2D static hits.

6.130.3 Member Data Documentation

6.130.3.1 Direction

```
Vector3 Direction
```

Represents the direction of the raycast for the query.

6.130.3.2 Length

```
float Length
```

Represents the maximum length of the raycast for the query.

6.130.3.3 Origin

Vector3 Origin

Represents the origin point of the raycast for the query.

6.130.3.4 QueryParams

QueryParams QueryParams

Represents the base parameters for the raycast query.

6.130.3.5 StaticHitsCapacity

int StaticHitsCapacity

Represents the capacity for the cached PhysX and Box2D static hits.

6.131 SnapshotHistoryDraw Class Reference

Provide a way to iterate over the HitboxBuffer and return the [HitboxColliderContainerDraw](#) container for each snapshot on the buffer.

Inherits [IEnumerable< HitboxColliderContainerDraw >](#).

Public Member Functions

- [IEnumerator< HitboxColliderContainerDraw > GetEnumerator \(\)](#)
Returns an enumerator that iterates through the [HitboxColliderContainerDraw](#).

6.131.1 Detailed Description

Provide a way to iterate over the HitboxBuffer and return the [HitboxColliderContainerDraw](#) container for each snapshot on the buffer.

6.131.2 Member Function Documentation

6.131.2.1 GetEnumerator()

```
IEnumerator<HitboxColliderContainerDraw> GetEnumerator ( )
```

Returns an enumerator that iterates through the [HitboxColliderContainerDraw](#).

6.132 SphereOverlapQuery Class Reference

Class that represents a sphere overlap query. Used to query against the [NetworkRunner.LagCompensation](#) API.

Inherits [Query](#).

Public Member Functions

- [SphereOverlapQuery](#) (ref [SphereOverlapQueryParams](#) sphereOverlapParams)
Create a new [SphereOverlapQuery](#) with the given [SphereOverlapQueryParams](#).
- [SphereOverlapQuery](#) (ref [SphereOverlapQueryParams](#) sphereOverlapParams, Collider[] physXOverlapHitsCache, Collider2D[] box2DOverlapHitsCache)
Create a new [SphereOverlapQuery](#) with the given [SphereOverlapQueryParams](#).

Public Attributes

- Vector3 [Center](#)
Represents the center of the sphere for the overlap query.
- float [Radius](#)
Represents the radius of the sphere for the overlap query.

Protected Member Functions

- override bool [Check](#) (ref [AABB](#) bounds)
Check if the given [AABB](#) intersects with the sphere overlap query.

6.132.1 Detailed Description

Class that represents a sphere overlap query. Used to query against the [NetworkRunner.LagCompensation](#) API.

6.132.2 Constructor & Destructor Documentation

6.132.2.1 SphereOverlapQuery() [1/2]

```
SphereOverlapQuery (
    ref SphereOverlapQueryParams sphereOverlapParams )
```

Create a new [SphereOverlapQuery](#) with the given [SphereOverlapQueryParams](#).

Parameters

<i>sphereOverlapParams</i>	The parameters to be used when creating the query.
----------------------------	--

6.132.2.2 SphereOverlapQuery() [2/2]

```
SphereOverlapQuery (
    ref SphereOverlapQueryParams sphereOverlapParams,
    Collider[ ] physXOverlapHitsCache,
    Collider2D[ ] box2DOverlapHitsCache )
```

Create a new [SphereOverlapQuery](#) with the given [SphereOverlapQueryParams](#).

Parameters

<i>sphereOverlapParams</i>	The parameters to be used when creating the query.
<i>physXOverlapHitsCache</i>	Array to write the results of the PhysX query if used.
<i>box2DOverlapHitsCache</i>	Array to write the results of the Box2D query if used.

6.132.3 Member Function Documentation**6.132.3.1 Check()**

```
override bool Check (
    ref AABB bounds ) [protected], [virtual]
```

Check if the given [AABB](#) intersects with the sphere overlap query.

Parameters

<i>bounds</i>	The bounds to check against.
---------------	------------------------------

Returns

True if the bounds intersects with the sphere overlap query, false otherwise.

Implements [Query](#).

6.132.4 Member Data Documentation

6.132.4.1 Center

Vector3 Center

Represents the center of the sphere for the overlap query.

6.132.4.2 Radius

float Radius

Represents the radius of the sphere for the overlap query.

6.133 SphereOverlapQueryParams Struct Reference

Base parameters needed to execute a sphere overlap query

Public Member Functions

- [SphereOverlapQueryParams \(QueryParams queryParams, Vector3 center, float radius, int staticHitsCapacity\)](#)
Create a new [SphereOverlapQueryParams](#).

Public Attributes

- Vector3 [Center](#)
Represents the center of the sphere for the overlap query.
- [queryParams](#) [queryParams](#)
Represents the base parameters for the sphere overlap query.
- float [Radius](#)
Represents the radius of the sphere for the overlap query.
- int [StaticHitsCapacity](#)
Represents the capacity for the cached PhysX and Box2D static hits.

6.133.1 Detailed Description

Base parameters needed to execute a sphere overlap query

6.133.2 Constructor & Destructor Documentation

6.133.2.1 [SphereOverlapQueryParams\(\)](#)

```
SphereOverlapQueryParams (
    QueryParams queryParams,
    Vector3 center,
    float radius,
    int staticHitsCapacity )
```

Create a new [SphereOverlapQueryParams](#).

Parameters

<i>queryParams</i>	Parameters to be used
<i>center</i>	The query center
<i>radius</i>	The query radius
<i>staticHitsCapacity</i>	Capacity for the cached PhysX and Box2D static hits.

6.133.3 Member Data Documentation

6.133.3.1 Center

`Vector3 Center`

Represents the center of the sphere for the overlap query.

6.133.3.2 QueryParams

`QueryParams queryParams`

Represents the base parameters for the sphere overlap query.

6.133.3.3 Radius

`float Radius`

Represents the radius of the sphere for the overlap query.

6.133.3.4 StaticHitsCapacity

`int StaticHitsCapacity`

Represents the capacity for the cached PhysX and Box2D static hits.

6.134 LagCompensationSettings Class Reference

Settings for lag compensation history.

Public Attributes

- int `CachedStaticCollidersSize` = 64
The size of the cached static colliders (PhysX or Box2D) array of the default Lag Compensation Queries.
- bool `Enabled` = false
Indicates if a `HitboxManager` instance should be added when the `NetworkRunner` is initialized.
- int `HitboxBufferLengthInMs` = 200
Hitbox snapshot history length in milliseconds.
- int `HitboxDefaultCapacity` = 512
Hitbox capacity per snapshot.

Properties

- float `ExpansionFactor` [get]
Broadphase BVH node expansion factor (default 20%) for leaf nodes, so updates are not too frequent.
- bool `Optimize` [get]
Optional: tries to optimize broadphase BVH every update. May be removed in the future.

6.134.1 Detailed Description

Settings for lag compensation history.

6.134.2 Member Data Documentation

6.134.2.1 CachedStaticCollidersSize

```
int CachedStaticCollidersSize = 64
```

The size of the cached static colliders (PhysX or Box2D) array of the default Lag Compensation Queries.

6.134.2.2 Enabled

```
bool Enabled = false
```

Indicates if a `HitboxManager` instance should be added when the `NetworkRunner` is initialized.

6.134.2.3 HitboxBufferLengthInMs

```
int HitboxBufferLengthInMs = 200
```

Hitbox snapshot history length in milliseconds.

6.134.2.4 HitboxDefaultCapacity

```
int HitboxDefaultCapacity = 512
```

[Hitbox](#) capacity per snapshot.

6.134.3 Property Documentation

6.134.3.1 ExpansionFactor

```
float ExpansionFactor [get]
```

Broadphase BVH node expansion factor (default 20%) for leaf nodes, so updates are not too frequent.

6.134.3.2 Optimize

```
bool Optimize [get]
```

Optional: tries to optimize broadphase BVH every update. May be removed in the future.

6.135 LayerAttribute Class Reference

Specifies that an int field should be drawn as a layer field in the inspector.

Inherits [DrawerPropertyAttribute](#).

6.135.1 Detailed Description

Specifies that an int field should be drawn as a layer field in the inspector.

6.136 LayerMatrixAttribute Class Reference

Specifies that the integer array field should be drawn as a layer matrix in the inspector.

Inherits [DrawerPropertyAttribute](#).

6.136.1 Detailed Description

Specifies that the integer array field should be drawn as a layer matrix in the inspector.

6.137 LobbyInfo Class Reference

Holds information about a Lobby

Properties

- bool `IsValid` [get]
Flag to signal if the `LobbyInfo` is ready for use. This is only true if the peer is currently connected to a Lobby.
- string `Name` [get]
Lobby Name
- string `Region` [get]
Stores the current connected Region

6.137.1 Detailed Description

Holds information about a Lobby

6.137.2 Property Documentation

6.137.2.1 IsValid

```
bool IsValid [get]
```

Flag to signal if the `LobbyInfo` is ready for use. This is only true if the peer is currently connected to a Lobby.

6.137.2.2 Name

```
string Name [get]
```

Lobby Name

6.137.2.3 Region

```
string Region [get]
```

Stores the current connected Region

6.138 LogSimpleUnity Class Reference

Log Simple Unity

Inherits ILogger.

Public Member Functions

- void [Log](#) (LogType logType, object message, in LogContext logContext)
- void [LogException](#) (Exception ex, in LogContext logContext)

6.138.1 Detailed Description

Log Simple Unity

6.139 Mask256 Struct Reference

[Mask256](#) is a 256-bit mask that can be used to store 256 boolean values.

Inherits IEquatable< Mask256 >.

Public Member Functions

- void [Clear](#) ()
Sets all bits to 0.
- bool [Equals](#) (Mask256 other)
Returns true if the masks are equal.
- override bool [Equals](#) (object obj)
Returns true if the masks are equal.
- bool [GetBit](#) (int bitIndex)
Returns a specific bit from the mask.
- override int [GetHashCode](#) ()
Calculates the hash code of the mask.
- bool [IsNothing](#) ()
Returns true if the mask is empty.
- Mask256 (long a, long b=0, long c=0, long d=0)
Creates a new mask with specified initial values.
- void [SetBit](#) (int bitIndex, bool set)
Sets a specific bit in the mask.
- override string [ToString](#) ()
Converts the mask to a string in the form of "a:b:c:d".

Static Public Member Functions

- static implicit [operator long \(Mask256 mask\)](#)
Equivalent to
- static implicit [operator Mask256 \(long value\)](#)
Converts a long value to a mask.
- static [Mask256 operator& \(Mask256 a, Mask256 b\)](#)
Performs a logical-AND operation.
- static [Mask256 operator| \(Mask256 a, Mask256 b\)](#)
Performs a logical-OR operation.
- static [Mask256 operator~ \(Mask256 a\)](#)
Performs a logical-NOT operation.

Properties

- long [this\[int i\] \[get, set\]](#)
Returns selected 64-bit value from the mask.

6.139.1 Detailed Description

[Mask256](#) is a 256-bit mask that can be used to store 256 boolean values.

6.139.2 Constructor & Destructor Documentation

6.139.2.1 Mask256()

```
Mask256 (
    long a,
    long b = 0,
    long c = 0,
    long d = 0 )
```

Creates a new mask with specified initial values.

6.139.3 Member Function Documentation

6.139.3.1 Clear()

```
void Clear ( )
```

Sets all bits to 0.

6.139.3.2 Equals() [1/2]

```
bool Equals (  
    Mask256 other )
```

Returns `true` if the masks are equal.

6.139.3.3 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Returns `true` if the masks are equal.

6.139.3.4 GetBit()

```
bool GetBit (   
    int bitIndex )
```

Returns a specific bit from the mask.

6.139.3.5 GetHashCode()

```
override int GetHashCode ( )
```

Calculates the hash code of the mask.

6.139.3.6 IsNothing()

```
bool IsNothing ( )
```

Returns `true` if the mask is empty.

Returns

6.139.3.7 operator long()

```
static implicit operator long (
    Mask256 mask ) [static]
```

Equivalent to

```
mask[0]
```

6.139.3.8 operator Mask256()

```
static implicit operator Mask256 (
    long value ) [static]
```

Converts a long value to a mask.

6.139.3.9 operator&()

```
static Mask256 operator& (
    Mask256 a,
    Mask256 b ) [static]
```

Performs a logical-AND operation.

6.139.3.10 operator" | ()

```
static Mask256 operator| (
    Mask256 a,
    Mask256 b ) [static]
```

Performs a logical-OR operation.

6.139.3.11 operator~()

```
static Mask256 operator~ (
    Mask256 a ) [static]
```

Performs a logical-NOT operation.

6.139.3.12 SetBit()

```
void SetBit (
    int bitIndex,
    bool set )
```

Sets a specific bit in the mask.

6.139.3.13 ToString()

```
override string ToString ( )
```

Converts the mask to a string in the form of "a:b:c:d".

6.139.4 Property Documentation

6.139.4.1 this[int i]

```
long this[int i] [get], [set]
```

Returns selected 64-bit value from the mask.

Parameters

<i>i</i>	
----------	--

6.140 MaxStringByteCountAttribute Class Reference

Specifies that the string field should be drawn as a text field with a maximum byte count for given encoding.

Inherits [DrawerPropertyAttribute](#).

Public Member Functions

- [MaxStringByteCountAttribute](#) (int count, string encoding)

Initializes a new instance of the [MaxStringByteCountAttribute](#) class with the specified byte count and encoding.

Properties

- int [ByteCount](#) [get]
Maximum byte count for the string.
- string [Encoding](#) [get]
The encoding of the string.

6.140.1 Detailed Description

Specifies that the string field should be drawn as a text field with a maximum byte count for given encoding.

6.140.2 Constructor & Destructor Documentation

6.140.2.1 MaxStringByteCountAttribute()

```
MaxStringByteCountAttribute (
    int count,
    string encoding )
```

Initializes a new instance of the [MaxStringByteCountAttribute](#) class with the specified byte count and encoding.

Parameters

<i>count</i>	
<i>encoding</i>	

6.140.3 Property Documentation

6.140.3.1 ByteCount

```
int ByteCount [get]
```

Maximum byte count for the string.

6.140.3.2 Encoding

```
string Encoding [get]
```

The encoding of the string.

6.141 NestedComponentUtilities Class Reference

Tools to replace GetComponent variants that respects nested objects. These are used to find components of a NetworkedObjects without also finding components that belong to parent or child NetworkedObjects.

Static Public Member Functions

- static T `EnsureRootComponentExists< T, TStopOn >` (this Transform transform)

Ensures that a component of type T exists on the root object of the provided transform. If a component of type T does not exist, it is added. The search for the root object stops if a component of type TStopOn is found.
- static T[] `FindObjectsOfTypeInOrder< T >` (this UnityEngine.SceneManagement.Scene scene, bool includeInactive=false)

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slower operation, and does produce garbage collection.
- static void `FindObjectsOfTypeInOrder< T >` (this UnityEngine.SceneManagement.Scene scene, List< T > list, bool includeInactive=false)

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slower operation which should not be run every update.
- static TCast[] `FindObjectsOfTypeInOrder< T, TCast >` (this UnityEngine.SceneManagement.Scene scene, bool includeInactive=false)

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slow operation, and does produce garbage collection.
- static void `FindObjectsOfTypeInOrder< T, TCast >` (this UnityEngine.SceneManagement.Scene scene, List< TCast > list, bool includeInactive=false)

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slower operation and should not be run every update.
- static T `GetNestedComponentInChildren< T, TStopOn >` (this Transform t, bool includeInactive)

Finds the first component of type T in the children of the provided transform, stopping the search if a component of type TStopOn is found.
- static T `GetNestedComponentInParent< T, TStopOn >` (this Transform t)

Same as GetComponentInParent, but will always include inactive objects in search. Will also stop recursing up the hierarchy when the StopOnT is found.
- static T `GetNestedComponentInParents< T, TStopOn >` (this Transform t)

Finds the first component of type T in the parents of the provided transform, stopping the search if a component of type TStopOn is found.
- static List< T > `GetNestedComponentsInChildren< T >` (this Transform t, List< T > list, bool includeInactive=true, params System.Type[] stopOn)

Same as GetComponentsInChildren, but will not recurse into children with any component of the types in the stopOn array.
- static void `GetNestedComponentsInChildren< T, TSearch, TStop >` (this Transform t, bool includeInactive, List< T > list)

Same as GetComponentsInChildren, but will not recurse into children with component of the StopT type.
- static List< T > `GetNestedComponentsInChildren< T, TStopOn >` (this Transform t, List< T > list, bool includeInactive=true)

Same as GetComponentsInChildren, but will not recurse into children with component of the StopT type.
- static void `GetNestedComponentsInParents< T >` (this Transform t, List< T > list)

Returns all T found between the child transform and its root. Order in List from child to parent, with the root/parent most being last.
- static void `GetNestedComponentsInParents< T, TStop >` (this Transform t, List< T > list)

Finds components of type T on supplied transform, and every parent above that node, inclusively stopping on node StopT component.
- static T `GetParentComponent< T >` (this Transform t)

Find T on supplied transform or any parent. Unlike GetComponentInParent, GameObjects do not need to be active to be found.

6.141.1 Detailed Description

Tools to replace GetComponent variants that respects nested objects. These are used to find components of a NetworkedObjects without also finding components that belong to parent or child NetworkedObjects.

6.141.2 Member Function Documentation

6.141.2.1 EnsureRootComponentExists< T, TStopOn >()

```
static T EnsureRootComponentExists< T, TStopOn > (
    this Transform transform ) [static]
```

Ensures that a component of type T exists on the root object of the provided transform. If a component of type T does not exist, it is added. The search for the root object stops if a component of type TStopOn is found.

Template Parameters

<i>T</i>	The type of component to ensure exists on the root object.
<i>TStopOn</i>	The type of component that stops the search for the root object.

Parameters

<i>transform</i>	The transform to start the search from.
------------------	---

Returns

The component of type T on the root object. Returns null if no root object is found.

Type Constraints

T : **Component**

TStopOn : **Component**

6.141.2.2 FindObjectsOfTypeInOrder< T >() [1/2]

```
static T [] FindObjectsOfTypeInOrder< T > (
    this UnityEngine.SceneManagement.Scene scene,
    bool includeInactive = false ) [static]
```

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slower operation, and does produce garbage collection.

Type Constraints

T : **class**

6.141.2.3 FindObjectsOfTypeInOrder< T >() [2/2]

```
static void FindObjectsOfTypeInOrder< T > (
    this UnityEngine.SceneManagement.Scene scene,
    List< T > list,
    bool includeInactive = false ) [static]
```

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slower operation which should not be run every update.

Template Parameters

<i>T</i>	The type being searched for.
----------	------------------------------

Parameters

<i>scene</i>	Scene to search.
<i>list</i>	Supplied list that will be populated by this find.
<i>includeInactive</i>	Whether results should include inactive components.

Type Constraints

*T : class*6.141.2.4 **FindObjectsOfTypeInOrder< T, TCast >()** [1/2]

```
static TCast [] FindObjectsOfTypeInOrder< T, TCast > (
    this UnityEngine.SceneManagement.Scene scene,
    bool includeInactive = false ) [static]
```

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slow operation, and does produce garbage collection.

Template Parameters

<i>T</i>	The type being searched for.
<i>TCast</i>	Casts all found objects to this type, and returns collection of this type. Objects that fail cast are excluded.

Parameters

<i>scene</i>	Scene to search.
<i>includeInactive</i>	Whether results should include inactive components.

Type Constraints

*T : class**TCast : class*6.141.2.5 **FindObjectsOfTypeInOrder< T, TCast >()** [2/2]

```
static void FindObjectsOfTypeInOrder< T, TCast > (
    this UnityEngine.SceneManagement.Scene scene,
```

```
List< TCast > list,
bool includeInactive = false ) [static]
```

Find All instances of Component type in a scene. Attempts to respect the hierarchy of the scene objects to produce a more deterministic order. This is a slower operation and should not be run every update.

Template Parameters

<i>T</i>	Type being searched for.
<i>TCast</i>	Type to cast found objects to.

Parameters

<i>scene</i>	Scene to search.
<i>list</i>	Supplied list that will be filled with found objects.
<i>includeInactive</i>	Whether results should include inactive components.

Type Constraints

T : class
TCast : class

6.141.2.6 GetNestedComponentInChildren< T, TStopOn >()

```
static T GetNestedComponentInChildren< T, TStopOn > (
    this Transform t,
    bool includeInactive ) [static]
```

Finds the first component of type T in the children of the provided transform, stopping the search if a component of type TStopOn is found.

Template Parameters

<i>T</i>	The type of component to find.
<i>TStopOn</i>	The type of component that stops the search.

Parameters

<i>t</i>	The transform to start the search from.
<i>includeInactive</i>	Whether to include inactive game objects in the search.

Returns

The first component of type T found. Returns null if no component is found.

Type Constraints

T : class

TStopOn : class

6.141.2.7 GetNestedComponentInParent< T, TStopOn >()

```
static T GetNestedComponentInParent< T, TStopOn > (
    this Transform t ) [static]
```

Same as GetComponentInParent, but will always include inactive objects in search. Will also stop recursing up the hierarchy when the StopOnT is found.

Type Constraints

T : class

TStopOn : class

6.141.2.8 GetNestedComponentInParents< T, TStopOn >()

```
static T GetNestedComponentInParents< T, TStopOn > (
    this Transform t ) [static]
```

Finds the first component of type T in the parents of the provided transform, stopping the search if a component of type TStopOn is found.

Template Parameters

T	The type of component to find.
TStopOn	The type of component that stops the search.

Parameters

t	The transform to start the search from.
----------	---

Returns

The first component of type T found in the parents. Returns null if no component is found or a component of type TStopOn is found.

Type Constraints

T : class

TStopOn : class

6.141.2.9 GetNestedComponentsInChildren< T >()

```
static List<T> GetNestedComponentsInChildren< T > (
    this Transform t,
    List< T > list,
    bool includeInactive = true,
    params System.Type[] stopOn ) [static]
```

Same as GetComponentsInChildren, but will not recurse into children with any component of the types in the stopOn array.

Type Constraints

T : class

6.141.2.10 GetNestedComponentsInChildren< T, TSearch, TStop >()

```
static void GetNestedComponentsInChildren< T, TSearch, TStop > (
    this Transform t,
    bool includeInactive,
    List< T > list ) [static]
```

Same as GetComponentsInChildren, but will not recurse into children with component of the StopT type.

Template Parameters

T	Cast found components to this type. Typically Component, but any other class/interface will work as long as they are assignable from SearchT.
TSearch	Find components of this class or interface type.
TStop	When this component is found, no further recursing will be performed on that node.

Type Constraints

T : class

TSearch : class

6.141.2.11 GetNestedComponentsInChildren< T, TStopOn >()

```
static List<T> GetNestedComponentsInChildren< T, TStopOn > (
    this Transform t,
    List< T > list,
    bool includeInactive = true ) [static]
```

Same as GetComponentsInChildren, but will not recurse into children with component of the StopT type.

Type Constraints

T : class

TStopOn : class

6.141.2.12 GetNestedComponentsInParents< T >()

```
static void GetNestedComponentsInParents< T > (
    this Transform t,
    List< T > list ) [static]
```

Returns all T found between the child transform and its root. Order in List from child to parent, with the root/parent most being last.

Type Constraints

T : Component

6.141.2.13 GetNestedComponentsInParents< T, TStop >()

```
static void GetNestedComponentsInParents< T, TStop > (
    this Transform t,
    List< T > list ) [static]
```

Finds components of type T on supplied transform, and every parent above that node, inclusively stopping on node StopT component.

Type Constraints

T : class

TStop : class

6.141.2.14 GetParentComponent< T >()

```
static T GetParentComponent< T > (
    this Transform t ) [static]
```

Find T on supplied transform or any parent. Unlike GetComponentInParent, GameObjects do not need to be active to be found.

Type Constraints

T : Component

6.142 NetworkArray< T > Struct Template Reference

Fusion type for networking arrays. Maximum capacity is fixed, and is set with the CapacityAttribute.

Inherits IEnumerable< T >, and INetworkArray.

Classes

- struct [Enumerator](#)
Enumerator for NetworkArray.

Public Member Functions

- void [Clear \(\)](#)
Clears the array, setting all values to default.
- void [CopyFrom \(List< T > source, int sourceOffset, int sourceCount\)](#)
Copies a range of values from a supplied source list into the NetworkArray.
- void [CopyFrom \(T\[\] source, int sourceOffset, int sourceCount\)](#)
Copies a range of elements from a source array into the NetworkArray.
- void [CopyTo \(List< T > list\)](#)
Adds each value to the supplied List. This does not clear the list, so values will be appended to the existing list.
- void [CopyTo \(NetworkArray< T > array\)](#)
Copies values to the supplied array.
- void [CopyTo \(T\[\] array, bool throwIfOverflow=true\)](#)
Copies values to the supplied array.
- T [Get \(int index\)](#)
Returns the array value at supplied index.
- [IEnumerator GetEnumerator \(\)](#)
Returns an enumerator that iterates through the collection.
- [IEnumerator< T > IEnumerable< T >. GetEnumerator \(\)](#)
- [IEnumerator IEnumerable. GetEnumerator \(\)](#)
- [NetworkArray \(byte *array, int length, IElementReaderWriter< T > readerWriter\)](#)
NetworkArray constructor.
- T [Set \(int index, T value\)](#)
Sets the array value at the supplied index.
- T[] [ToArray \(\)](#)
Allocates a new array and copies values from this array. For a non-alloc alternative use CopyTo(List< T >).
- string [ToString \(\)](#)
Returns the elements of this array as a string, with value separated by characters. Specifically for use in the Unity inspector. This is private and only is found by NetworkBehaviourEditor using reflection, so do not rename this method.
- [NetworkArrayReadOnly< T > ToReadOnly \(\)](#)
Returns a NetworkArrayReadOnly view of this array.
- override string [ToString \(\)](#)
Returns a string that represents the current object.

Static Public Member Functions

- static implicit [operator NetworkArrayReadOnly< T > \(NetworkArray< T > value\)](#)
Returns a NetworkArrayReadOnly view of this array.

Public Attributes

- byte * [_array](#)
- int [_length](#)
- [IElementReaderWriter< T > _readerWriter](#)

Static Public Attributes

- static StringBuilder **_stringBuilderCached**

Properties

- int **Length** [get]
The fixed size of the array.
- T **this[int index]** [get, set]
Indexer of array elements.
- object INetworkArray. **this[int index]** [get, set]

6.142.1 Detailed Description

Fusion type for networking arrays. Maximum capacity is fixed, and is set with the [CapacityAttribute](#).

Typical Usage: [Networked, Capacity(4)]
`NetworkArray<float> syncedArray => default;`

Optional usage (for NetworkBehaviours ONLY - this is not legal in INetworkStructs): [Networked, Capacity(4)]
`NetworkArray<int> syncedArray { get; } = MakeInitializer(new int[] { 1, 2, 3, 4 });`

Usage for modifying data: `array.Set(123); array[0] = 456;`

Template Parameters

T	T can be a primitive, or an INetworkStruct .
----------	--

6.142.2 Constructor & Destructor Documentation

6.142.2.1 NetworkArray()

```
NetworkArray (
    byte * array,
    int length,
    IElementReaderWriter< T > readerWriter )
```

`NetworkArray` constructor.

6.142.3 Member Function Documentation

6.142.3.1 Clear()

```
void Clear( )
```

Clears the array, setting all values to default.

6.142.3.2 CopyFrom() [1/2]

```
void CopyFrom(
    List< T > source,
    int sourceOffset,
    int sourceCount )
```

Copies a range of values from a supplied source list into the [NetworkArray](#).

Parameters

<i>source</i>	The source list from which to copy elements.
<i>sourceOffset</i>	The zero-based index in the source list at which copying begins.
<i>sourceCount</i>	The number of elements to copy from the source list.

Exceptions

<i>ArgumentNullException</i>	Thrown when the provided source list is null.
<i>ArgumentException</i>	Thrown when the number of elements to copy is greater than the length of the NetworkArray .
<i>ArgumentOutOfRangeException</i>	Thrown when the sum of sourceOffset and sourceCount is greater than the length of the source list.

6.142.3.3 CopyFrom() [2/2]

```
void CopyFrom(
    T[ ] source,
    int sourceOffset,
    int sourceCount )
```

Copies a range of elements from a source array into the [NetworkArray](#).

Parameters

<i>source</i>	The source array from which to copy elements. Must not be null.
<i>sourceOffset</i>	The zero-based index in the source array at which copying begins.
<i>sourceCount</i>	The number of elements to copy from the source array.

Exceptions

<i>ArgumentNullException</i>	Thrown when the provided source array is null.
<i>ArgumentException</i>	Thrown when the number of elements to copy is greater than the length of the NetworkArray .
<i>ArgumentOutOfRangeException</i>	Thrown when the sum of sourceOffset and sourceCount is greater than the length of the source array.

6.142.3.4 CopyTo() [1/3]

```
void CopyTo (
    List< T > list )
```

Adds each value to the supplied List. This does not clear the list, so values will be appended to the existing list.

6.142.3.5 CopyTo() [2/3]

```
void CopyTo (
    NetworkArray< T > array )
```

Copies values to the supplied array.

Parameters

<i>array</i>	NetworkArray to copy to.
--------------	--

Exceptions

<i>ArgumentException</i>	Thrown if the supplied array is smaller than this NetworkArray<T> .
--------------------------	---

6.142.3.6 CopyTo() [3/3]

```
void CopyTo (
    T[ ] array,
    bool throwIfOverflow = true )
```

Copies values to the supplied array.

Parameters

<i>array</i>	Array to copy to.
<i>throwIfOverflow</i>	If true, this method will throw an error if the supplied array is smaller than this NetworkArray<T> . If false, will only copy as many elements as the target array can hold.

6.142.3.7 Get()

```
T Get (
    int index )
```

Returns the array value at supplied index.

6.142.3.8 GetEnumerator()

```
Enumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the collection.

6.142.3.9 operator NetworkArrayReadOnly< T >()

```
static implicit operator NetworkArrayReadOnly< T > (
    NetworkArray< T > value ) [static]
```

Returns a [NetworkArrayReadOnly](#) view of this array.

Parameters

value	NetworkArray to convert.
-------	--

Returns

[NetworkArrayReadOnly](#) view of this array.

6.142.3.10 Set()

```
T Set (
    int index,
    T value )
```

Sets the array value at the supplied index.

6.142.3.11 ToArray()

```
T [ ] ToArray ( )
```

Allocates a new array and copies values from this array. For a non-alloc alternative use [CopyTo\(List<T>\)](#).

6.142.3.12 ToStringString()

```
string ToStringString ( )
```

Returns the elements of this array as a string, with value separated by characters. Specifically for use in the Unity inspector. This is private and only is found by NetworkBehaviourEditor using reflection, so do not rename this method.

6.142.3.13 ToReadOnly()

```
NetworkArrayReadOnly<T> ToReadOnly ( )
```

Returns a [NetworkArrayReadOnly](#) view of this array.

6.142.3.14 ToString()

```
override string ToString ( )
```

Returns a string that represents the current object.

6.142.4 Property Documentation

6.142.4.1 Length

```
int Length [get]
```

The fixed size of the array.

6.142.4.2 this[int index]

```
T this[int index] [get], [set]
```

Indexer of array elements.

6.143 NetworkArray< T >.Enumerator Struct Reference

Enumerator for [NetworkArray](#).

Inherits [IEnumerator< T >](#).

Public Member Functions

- void [Dispose \(\)](#)
Releases all resources used by the [Enumerator](#).
- [Enumerator \(NetworkArray< T > array\)](#)
Initializes a new instance of the [Enumerator](#) with the specified [NetworkArray](#).
- bool [MoveNext \(\)](#)
Advances the enumerator to the next element of the collection.
- void [Reset \(\)](#)
Sets the enumerator to its initial position, which is before the first element in the collection.

Public Attributes

- [NetworkArray< T > _array](#)
- int [_index](#)

Properties

- T [Current \[get\]](#)
Gets the current element in the collection.
- object [IEnumerator.Current \[get\]](#)
Gets the current element in the collection.

6.143.1 Detailed Description

Enumerator for [NetworkArray](#).

6.143.2 Constructor & Destructor Documentation

6.143.2.1 Enumerator()

```
Enumerator (
    NetworkArray< T > array )
```

Initializes a new instance of the [Enumerator](#) with the specified [NetworkArray](#).

Parameters

<code>array</code>	The NetworkArray to enumerate.
--------------------	--

6.143.3 Member Function Documentation

6.143.3.1 Dispose()

```
void Dispose ( )
```

Releases all resources used by the [Enumerator](#).

6.143.3.2 MoveNext()

```
bool MoveNext ( )
```

Advances the enumerator to the next element of the collection.

Returns

true if the enumerator was successfully advanced to the next element; false if the enumerator has passed the end of the collection.

6.143.3.3 Reset()

```
void Reset ( )
```

Sets the enumerator to its initial position, which is before the first element in the collection.

6.143.4 Property Documentation

6.143.4.1 Current [1/2]

```
T Current [get]
```

Gets the current element in the collection.

6.143.4.2 Current [2/2]

```
object IEnumarator. Current [get]
```

Gets the current element in the collection.

6.144 NetworkArrayExtensions Class Reference

Provides extension methods for the [NetworkArray](#) class.

Static Public Member Functions

- static ref T [GetRef< T >](#) (this [NetworkArray< T >](#) array, int index)
Returns a reference to the element at a specified position in the [NetworkArray](#).
- static int [IndexOf< T >](#) (this [NetworkArray< T >](#) array, T elem)
Finds the index of the first occurrence of a specified element in the [NetworkArray](#).

6.144.1 Detailed Description

Provides extension methods for the [NetworkArray](#) class.

6.144.2 Member Function Documentation

6.144.2.1 GetRef< T >()

```
static ref T GetRef< T > (
    this NetworkArray< T > array,
    int index ) [static]
```

Returns a reference to the element at a specified position in the [NetworkArray](#).

Parameters

<i>array</i>	The NetworkArray to search.
<i>index</i>	The zero-based index of the element to get a reference for.

Returns

A reference to the element at the specified position in the [NetworkArray](#).

Type Constraints

T : unmanaged

6.144.2.2 IndexOf< T >()

```
static int IndexOf< T > (
    this NetworkArray< T > array,
    T elem ) [static]
```

Finds the index of the first occurrence of a specified element in the [NetworkArray](#).

Parameters

<i>array</i>	The NetworkArray to search.
<i>elem</i>	The element to locate in the NetworkArray .

Returns

The zero-based index of the first occurrence of *elem* within the entire [NetworkArray](#), if found; otherwise, -1.

Type Constraints

- T : unmanaged*
- T : IEquatable< T >*

6.145 NetworkArrayReadOnly< T > Struct Template Reference

Provides a read-only view of a network array.

Public Attributes

- readonly byte * *_array*
- readonly int *_length*
- readonly [IElementReaderWriter](#)< T > *_readerWriter*

Properties

- int *Length* [get]
The fixed size of the array.
- T *this[int index]* [get]
Indexer of array elements.

6.145.1 Detailed Description

Provides a read-only view of a network array.

Template Parameters

<i>T</i>	The type of the elements.
----------	---------------------------

6.145.2 Property Documentation

6.145.2.1 Length

int Length [get]

The fixed size of the array.

6.145.2.2 this[int index]

T this[int index] [get]

Indexer of array elements.

6.146 NetworkAssemblyIgnoreAttribute Class Reference

Network Assembly Ignore Attribute

Inherits Attribute.

6.146.1 Detailed Description

Network Assembly Ignore Attribute

6.147 NetworkAssemblyWeavedAttribute Class Reference

Network Assembly Weaved Attribute

Inherits Attribute.

6.147.1 Detailed Description

Network Assembly Weaved Attribute

6.148 NetworkBehaviour Class Reference

Base class for [Fusion](#) network components, which are associated with a [NetworkObject](#).

Inherits [SimulationBehaviour](#), [ISpawned](#), and [IDespawned](#).

Inherited by [HitboxRoot](#), [NetworkMecanimAnimator](#), and [NetworkTRSP](#).

Classes

- struct [ArrayReader](#)
Provides a reader for network arrays of type T.
- struct [BehaviourReader](#)
Provides a reader for network behaviours of type T.
- class [ChangeDetector](#)
Change detector for a NetworkBehaviour
- struct [DictionaryReader](#)
Provides a reader for network dictionaries with keys of type K and values of type V.
- struct [LinkListReader](#)
Provides a reader for network linked lists of type T.
- struct [PropertyReader](#)
Provides a reader for properties of type T in a network behaviour.

Public Member Functions

- virtual void [CopyBackingFieldsToState](#) (bool firstTime)
Copies the backing fields to the state. This method is meant to be overridden in derived classes.
- void [CopyStateFrom](#) ([NetworkBehaviour](#) source)
Copies entire state of passed in source NetworkBehaviour
- virtual void [CopyStateToBackingFields](#) ()
Copies the state to the backing fields. This method is meant to be overridden in derived classes.
- virtual void [Despawned](#) ([NetworkRunner](#) runner, bool hasState)
Called before the network object is despawned
- override void [FixedUpdateNetwork](#) ()
Fixed update callback for networked behaviours.
- [ArrayReader](#)< T > [GetArrayReader](#)< T > (string property)
Gets an ArrayReader for a network array of type T.
- [BehaviourReader](#)< T > [GetBehaviourReader](#)< T > (string property)
Gets a BehaviourReader for a network behaviour of type T.
- [ChangeDetector](#) [GetChangeDetector](#) ([ChangeDetector.Source](#) source, bool copyInitial=true)
Creates a ChangeDetector for this network behaviour.
- [DictionaryReader](#)< K, V > [GetDictionaryReader](#)< K, V > (string property)
Gets a DictionaryReader for a network dictionary with keys of type K and values of type V.
- T? [GetInput](#)< T > ()
Returns true if it a valid INetworkInput can be found for the current simulation tick (Typically this is used in FixedUpdateNetwork).
- [LinkListReader](#)< T > [GetLinkListReader](#)< T > (string property)
Gets a LinkListReader for a network linked list of type T.
- int [GetLocalAuthorityMask](#) ()
Gets a bitmask of AuthorityMasks flags, representing the current local authority over this NetworkObject.
- [PropertyReader](#)< T > [GetPropertyReader](#)< T > (string property)
Gets a PropertyReader for a property of type T in this network behaviour.
- ref T [ReinterpretState](#)< T > (int offset=0)
Allows read and write access to the internal state buffer
- void [ReplicateTo](#) ([PlayerRef](#) player, bool replicate)
Controls if this network behaviours state is replicated to a player or not
- void [ReplicateToAll](#) (bool replicate)

Sets the default replicated state for this behaviour, this by default is true so calling ReplicateToAll(true) does nothing if ReplicateToAll(false) hasn't been called before

- void [ResetState \(\)](#)
Resets the state of the object to the original state
- virtual void [Spawned \(\)](#)
Post spawn callback.
- bool [TryGetSnapshotsBuffers](#) (out [NetworkBehaviourBuffer](#) from, out [NetworkBehaviourBuffer](#) to, out float alpha)
Tries to get the snapshot buffers for this network behaviour.

Static Public Member Functions

- static [ArrayReader< T > GetArrayReader< T >](#) (Type behaviourType, string property, [IElementReaderWriter< T >](#) readerWriter=null)
Gets an [ArrayReader](#) for a network array of type T in a network behaviour of a specific type.
- static [BehaviourReader< T > GetBehaviourReader< T >](#) ([NetworkRunner](#) runner, Type behaviourType, string property)
Gets a [BehaviourReader](#) for a network behaviour of type T.
- static [BehaviourReader< TProperty > GetBehaviourReader< TBehaviour, TProperty >](#) ([NetworkRunner](#) runner, string property)
Gets a [BehaviourReader](#) for a network behaviour with a specific property of type TProperty.
- static [DictionaryReader< K, V > GetDictionaryReader< K, V >](#) (Type behaviourType, string property, [IElementReaderWriter< K >](#) keyReaderWriter=null, [IElementReaderWriter< V >](#) valueReaderWriter=null)
Gets a [DictionaryReader](#) for a network dictionary with keys of type K and values of type V in a network behaviour of a specific type.
- static [LinkListReader< T > GetLinkListReader< T >](#) (Type behaviourType, string property, [IElementReaderWriter< T >](#) readerWriter=null)
Gets a [LinkListReader](#) for a network linked list of type T in a network behaviour of a specific type.
- static [PropertyReader< T > GetPropertyReader< T >](#) (Type behaviourType, string property)
Gets a [PropertyReader](#) for a property of type T in a network behaviour of a specific type.
- static [PropertyReader< TProperty > GetPropertyReader< TBehaviour, TProperty >](#) (string property)
Gets a [PropertyReader](#) for a property of type TProperty in a network behaviour of type TBehaviour.
- static [NetworkBehaviourUtils.DictionaryInitializer< K, V > MakeInitializer< K, V >](#) (Dictionary< K, V > dictionary)
This is a special method that is meant to be used only for [Networked] properties inline initialization.
- static [NetworkBehaviourUtils.ArrayInitializer< T > MakeInitializer< T >](#) (T[] array)
This is a special method that is meant to be used only for [Networked] properties inline initialization.
- static T * [MakePtr< T >](#) ()
Creates a pointer to a value of type T. This method is meant to be used only for [Networked] properties inline initialization.
- static T * [MakePtr< T >](#) (T defaultValue)
Creates a pointer to a value of type T, initializing it with the provided default value. This method is meant to be used only for [Networked] properties inline initialization.
- static ref T [MakeRef< T >](#) ()
Creates a reference to a value of type T. This method is meant to be used only for [Networked] properties inline initialization.
- static ref T [MakeRef< T >](#) (T defaultValue)
Creates a reference to a value of type T, initializing it with the provided default value. This method is meant to be used only for [Networked] properties inline initialization.
- static int [NetworkDeserialize](#) ([NetworkRunner](#) runner, byte *data, ref [NetworkBehaviour](#) result)
Deserializes a [NetworkBehaviour](#) from a byte array.

- static int [NetworkSerialize](#) ([NetworkBehaviour](#) obj, byte *data)
Serializes a [NetworkBehaviour](#) into a byte array.
- static int [NetworkSerialize](#) ([NetworkRunner](#) runner, [NetworkBehaviour](#) obj, byte *data)
Serializes a [NetworkBehaviour](#) into a byte array.
- static [NetworkBehaviour](#) [NetworkUnwrap](#) ([NetworkRunner](#) runner, [NetworkBehaviourId](#) wrapper)
Converts a [NetworkBehaviourId](#) to a [NetworkBehaviour](#).
- static [NetworkBehaviourId](#) [NetworkWrap](#) ([NetworkRunner](#) runner, [NetworkBehaviour](#) obj)
Converts a [NetworkBehaviour](#) to a [NetworkBehaviourId](#).
- static implicit operator [NetworkBehaviourId](#) ([NetworkBehaviour](#) behaviour)
Converts [NetworkBehaviour](#) to [NetworkBehaviourId](#)

Public Attributes

- int [offset](#)
Gives access to the offset (in 32 bit words) and count (in 32 bit words) of this behaviour backing data

Protected Member Functions

- virtual bool [ReplicateTo](#) ([PlayerRef](#) player)
Determines whether to replicate the network behaviour to the specified player. This method can be overridden in derived classes to implement custom replication logic.

Properties

- [Tick](#) [ChangedTick](#) [get]
The tick the data on this networked behaviour changed
- virtual ? int [DynamicWordCount](#) [get]
Override this value for custom memory allocations. This is for advanced use cases only, and cannot be used if [NetworkedAttribute](#) is used in the derived class.
- bool? [HasInputAuthority](#) [get]
Returns true if the [Simulation.LocalPlayer](#) of the associated [NetworkRunner](#) is the designated as Input Source for this network entity.
- bool? [HasStateAuthority](#) [get]
Returns true if the associated [NetworkRunner](#) is the State Source for this network entity.
- [NetworkBehaviourId?](#) [Id](#) [get]
The unique identifier for this network behaviour.
- bool? [IsProxy](#) [get]
Returns true if the associated [NetworkRunner](#) is neither the Input nor State Authority for this network entity. It is recommended to use [!HasStateAuthority](#) or [!HasInputAuthority](#) when possible instead, as this check requires evaluating both authorities - and is therefore less performant than the individual checks.
- [NetworkBehaviourBuffer](#) [StateBuffer](#) [get]
Gets the state buffer associated with the network behaviour.
- bool [StateBufferIsValid](#) [get]
Gets a value indicating whether the state buffer is valid.

6.148.1 Detailed Description

Base class for [Fusion](#) network components, which are associated with a [NetworkObject](#).

Derived from [SimulationBehaviour](#), components derived from this class are associated with a [NetworkRunner](#) and [Simulation](#). Components derived from this class are associated with a parent [NetworkObject](#). and can use the [NetworkedAttribute](#) on properties to automate state synchronization, and can use the [RpcAttribute](#) on methods, to automate messaging.

6.148.2 Member Function Documentation

6.148.2.1 CopyBackingFieldsToState()

```
virtual void CopyBackingFieldsToState (
    bool firstTime ) [virtual]
```

Copies the backing fields to the state. This method is meant to be overridden in derived classes.

Parameters

<i>firstTime</i>	Indicates whether this is the first time the method is called.
------------------	--

6.148.2.2 CopyStateFrom()

```
void CopyStateFrom (
    NetworkBehaviour source )
```

Copies entire state of passed in source [NetworkBehaviour](#)

Parameters

<i>source</i>	Source NetworkBehaviour to copy data from
---------------	---

6.148.2.3 CopyStateToBackingFields()

```
virtual void CopyStateToBackingFields ( ) [virtual]
```

Copies the state to the backing fields. This method is meant to be overridden in derived classes.

6.148.2.4 Despawned()

```
virtual void Despawned (
    NetworkRunner runner,
    bool hasState ) [virtual]
```

Called before the network object is despawned

Parameters

<i>runner</i>	The runner that owns the object
<i>hasState</i>	If the state of the behaviour is still accessible

Implements [IDespawned](#).

Reimplemented in [HitboxRoot](#).

6.148.2.5 FixedUpdateNetwork()

```
override void FixedUpdateNetwork ( ) [virtual]
```

Fixed update callback for networked behaviours.

Reimplemented from [SimulationBehaviour](#).

6.148.2.6 GetArrayReader< T >() [1/2]

```
ArrayReader<T> GetArrayReader< T > (
    string property )
```

Gets an [ArrayReader](#) for a network array of type T.

Template Parameters

<i>T</i>	The type of elements in the network array.
----------	--

Parameters

<i>property</i>	The name of the property to be read.
-----------------	--------------------------------------

Returns

An [ArrayReader](#) for the network array of type T.

6.148.2.7 GetArrayReader< T >() [2/2]

```
static ArrayReader<T> GetArrayReader< T > (
    Type behaviourType,
    string property,
    IElementReaderWriter< T > readerWriter = null ) [static]
```

Gets an [ArrayReader](#) for a network array of type T in a network behaviour of a specific type.

Template Parameters

<i>T</i>	The type of elements in the network array.
----------	--

Parameters

<i>behaviourType</i>	The type of the network behaviour.
<i>property</i>	The name of the property to be read.
<i>readerWriter</i>	

Returns

An [ArrayReader](#) for the network array of type T in the network behaviour of the specified type.

6.148.2.8 GetBehaviourReader< T >() [1/2]

```
static BehaviourReader<T> GetBehaviourReader< T > (
    NetworkRunner runner,
    Type behaviourType,
    string property ) [static]
```

Gets a [BehaviourReader](#) for a network behaviour of type T.

Template Parameters

<i>T</i>	The type of the network behaviour.
----------	------------------------------------

Parameters

<i>runner</i>	The NetworkRunner associated with the network behaviour.
<i>behaviourType</i>	The type of the behaviour to be read.
<i>property</i>	The name of the property to be read.

Returns

A [BehaviourReader](#) for the network behaviour of type T.

Type Constraints

T : *NetworkBehaviour*

6.148.2.9 GetBehaviourReader< T >() [2/2]

```
BehaviourReader<T> GetBehaviourReader< T > (
    string property )
```

Gets a [BehaviourReader](#) for a network behaviour of type T.

Template Parameters

<i>T</i>	The type of the network behaviour.
----------	------------------------------------

Parameters

<i>property</i>	The name of the property to be read.
-----------------	--------------------------------------

Returns

A [BehaviourReader](#) for the network behaviour of type T.

Type Constraints

T : *NetworkBehaviour*

6.148.2.10 GetBehaviourReader< TBehaviour, TProperty >()

```
static BehaviourReader<TProperty> GetBehaviourReader< TBehaviour, TProperty > (
    NetworkRunner runner,
    string property ) [static]
```

Gets a [BehaviourReader](#) for a network behaviour with a specific property of type TProperty.

Template Parameters

<i>TBehaviour</i>	The type of the network behaviour.
<i>TProperty</i>	The type of the property in the network behaviour.

Parameters

<i>runner</i>	The NetworkRunner associated with the network behaviour.
<i>property</i>	The name of the property to be read.

Returns

A [BehaviourReader](#) for the network behaviour with the specific property of type *TProperty*.

Type Constraints

TBehaviour : *NetworkBehaviour*

TProperty : *NetworkBehaviour*

6.148.2.11 GetChangeDetector()

```
ChangeDetector GetChangeDetector (
    ChangeDetector.Source source,
    bool copyInitial = true )
```

Creates a [ChangeDetector](#) for this network behaviour.

Parameters

<i>source</i>	The source of the change detector.
<i>copyInitial</i>	Indicates whether to copy the initial state of the network behaviour.

Returns

A [ChangeDetector](#) for this network behaviour.

6.148.2.12 GetDictionaryReader< K, V >() [1/2]

```
DictionaryReader<K, V> GetDictionaryReader< K, V > (
    string property )
```

Gets a [DictionaryReader](#) for a network dictionary with keys of type *K* and values of type *V*.

Template Parameters

<i>K</i>	The type of keys in the network dictionary.
<i>V</i>	The type of values in the network dictionary.

Parameters

<i>property</i>	The name of the property to be read.
-----------------	--------------------------------------

Returns

A [DictionaryReader](#) for the network dictionary with keys of type K and values of type V.

6.148.2.13 GetDictionaryReader< K, V >() [2/2]

```
static DictionaryReader<K, V> GetDictionaryReader< K, V > (
    Type behaviourType,
    string property,
    IElementReaderWriter< K > keyReaderWriter = null,
    IElementReaderWriter< V > valueReaderWriter = null ) [static]
```

Gets a [DictionaryReader](#) for a network dictionary with keys of type K and values of type V in a network behaviour of a specific type.

Template Parameters

<i>K</i>	The type of keys in the network dictionary.
<i>V</i>	The type of values in the network dictionary.

Parameters

<i>behaviourType</i>	The type of the network behaviour.
<i>property</i>	The name of the property to be read.
<i>keyReaderWriter</i>	
<i>valueReaderWriter</i>	

Returns

A [DictionaryReader](#) for the network dictionary with keys of type K and values of type V in the network behaviour of the specified type.

6.148.2.14 GetInput< T >() [1/2]

```
T? GetInput< T > ( )
```

Template Parameters

<i>T</i>	
----------	--

Type Constraints

T : *unmanaged*

T : *INetworkInput*

6.148.2.15 GetInput< T >() [2/2]

```
bool GetInput< T > (
    out T input )
```

Returns true if it a valid [INetworkInput](#) can be found for the current simulation tick (Typically this is used in [FixedUpdateNetwork](#)).

The returned input struct originates from the [NetworkObject.InputAuthority](#), and if valid contains the inputs supplied by that [PlayerRef](#) for the current simulation tick.

Type Constraints

T : unmanaged

T : INetworkInput

6.148.2.16 GetLinkListReader< T >() [1/2]

```
LinkListReader<T> GetLinkListReader< T > (
    string property )
```

Gets a [LinkListReader](#) for a network linked list of type T.

Template Parameters

<i>T</i>	The type of elements in the network linked list.
----------	--

Parameters

<i>property</i>	The name of the property to be read.
-----------------	--------------------------------------

Returns

A [LinkListReader](#) for the network linked list of type T.

6.148.2.17 GetLinkListReader< T >() [2/2]

```
static LinkListReader<T> GetLinkListReader< T > (
    Type behaviourType,
    string property,
    IElementReaderWriter< T > readerWriter = null ) [static]
```

Gets a [LinkListReader](#) for a network linked list of type T in a network behaviour of a specific type.

Template Parameters

<i>T</i>	The type of elements in the network linked list.
----------	--

Parameters

<i>behaviourType</i>	The type of the network behaviour.
<i>property</i>	The name of the property to be read.
<i>readerWriter</i>	

Returns

A [LinkListReader](#) for the network linked list of type T in the network behaviour of the specified type.

6.148.2.18 GetLocalAuthorityMask()

```
int GetLocalAuthorityMask ( )
```

Gets a bitmask of [AuthorityMasks](#) flags, representing the current local authority over this [NetworkObject](#).

6.148.2.19 GetPropertyReader< T >() [1/2]

```
PropertyReader<T> GetPropertyReader< T > (
    string property )
```

Gets a [PropertyReader](#) for a property of type T in this network behaviour.

Template Parameters

<i>T</i>	The type of the property in the network behaviour. Must be unmanaged.
----------	---

Parameters

<i>property</i>	The name of the property to be read.
-----------------	--------------------------------------

Returns

A [PropertyReader](#) for the property of type T in this network behaviour.

Type Constraints

T : **unmanaged**

6.148.2.20 GetPropertyReader< T >() [2/2]

```
static PropertyReader<T> GetPropertyReader< T > (
    Type behaviourType,
    string property) [static]
```

Gets a [PropertyReader](#) for a property of type T in a network behaviour of a specific type.

Template Parameters

<i>T</i>	The type of the property in the network behaviour. Must be unmanaged.
----------	---

Parameters

<i>behaviourType</i>	The type of the network behaviour.
<i>property</i>	The name of the property to be read.

Returns

A [PropertyReader](#) for the property of type T in the network behaviour of the specified type.

Type Constraints

T : unmanaged

6.148.2.21 GetPropertyReader< TBehaviour, TProperty >()

```
static PropertyReader<TProperty> GetPropertyReader< TBehaviour, TProperty > (
    string property) [static]
```

Gets a [PropertyReader](#) for a property of type TProperty in a network behaviour of type TBehaviour.

Template Parameters

<i>TBehaviour</i>	The type of the network behaviour.
<i>TProperty</i>	The type of the property in the network behaviour. Must be unmanaged.

Parameters

<i>property</i>	The name of the property to be read.
-----------------	--------------------------------------

Returns

A [PropertyReader](#) for the property of type TProperty in the network behaviour of type TBehaviour.

Type Constraints

TBehaviour : NetworkBehaviour

*T*Property : *unmanaged*

6.148.2.22 MakeInitializer< K, V >()

```
static NetworkBehaviourUtils.DictionaryInitializer<K, V> MakeInitializer< K, V > (
    Dictionary< K, V > dictionary ) [static]
```

This is a special method that is meant to be used only for [Networked] properties inline initialization.

6.148.2.23 MakeInitializer< T >()

```
static NetworkBehaviourUtils.ArrayInitializer<T> MakeInitializer< T > (
    T[ ] array ) [static]
```

This is a special method that is meant to be used only for [Networked] properties inline initialization.

6.148.2.24 MakePtr< T >() [1/2]

```
static T* MakePtr< T > ( ) [static]
```

Creates a pointer to a value of type T. This method is meant to be used only for [Networked] properties inline initialization.

Template Parameters

<i>T</i>	The type of the value. Must be unmanaged.
----------	---

Returns

A pointer to a value of type T.

Exceptions

<i>NotImplementedException</i>	Thrown because this method is meant to be used only for [Networked] properties inline initialization.
--------------------------------	---

Type Constraints

T: *unmanaged*

6.148.2.25 MakePtr< T >() [2/2]

```
static T* MakePtr< T > (
    T defaultValue )  [static]
```

Creates a pointer to a value of type T, initializing it with the provided default value. This method is meant to be used only for [Networked] properties inline initialization.

Template Parameters

<i>T</i>	The type of the value. Must be unmanaged.
----------	---

Parameters

<i>defaultValue</i>	The default value to initialize the value with.
---------------------	---

Returns

A pointer to a value of type T.

Exceptions

<i>NotImplementedException</i>	Thrown because this method is meant to be used only for [Networked] properties inline initialization.
--------------------------------	---

Type Constraints

T: **unmanaged**

6.148.2.26 MakeRef< T >() [1/2]

```
static ref T MakeRef< T > ( )  [static]
```

Creates a reference to a value of type T. This method is meant to be used only for [Networked] properties inline initialization.

Template Parameters

<i>T</i>	The type of the value. Must be unmanaged.
----------	---

Returns

A reference to a value of type T.

Exceptions

<i>NotImplementedException</i>	Thrown because this method is meant to be used only for [Networked] properties inline initialization.
--------------------------------	---

Type Constraints

T : **unmanaged**

6.148.2.27 MakeRef< T >() [2/2]

```
static ref T MakeRef< T > (
    T defaultValue ) [static]
```

Creates a reference to a value of type T, initializing it with the provided default value. This method is meant to be used only for [Networked] properties inline initialization.

Template Parameters

<i>T</i>	The type of the value. Must be unmanaged.
----------	---

Parameters

<i>defaultValue</i>	The default value to initialize the value with.
---------------------	---

Returns

A reference to a value of type T.

Exceptions

<i>NotImplementedException</i>	Thrown because this method is meant to be used only for [Networked] properties inline initialization.
--------------------------------	---

Type Constraints

T : **unmanaged**

6.148.2.28 NetworkDeserialize()

```
static int NetworkDeserialize (
    NetworkRunner runner,
    byte * data,
    ref NetworkBehaviour result ) [static]
```

Deserializes a [NetworkBehaviour](#) from a byte array.

Parameters

<i>runner</i>	The NetworkRunner associated with the NetworkBehaviour .
<i>data</i>	The byte pointer to read the serialized data from.
<i>result</i>	The NetworkBehaviour to store the deserialized data in.

Returns

The size of the deserialized data in bytes.

6.148.2.29 NetworkSerialize() [1/2]

```
static int NetworkSerialize (
    NetworkBehaviour obj,
    byte * data ) [static]
```

Serializes a [NetworkBehaviour](#) into a byte array.

Parameters

<i>obj</i>	The NetworkBehaviour to be serialized.
<i>data</i>	The byte pointer to write the serialized data to.

Returns

The size of the serialized data in bytes.

6.148.2.30 NetworkSerialize() [2/2]

```
static int NetworkSerialize (
    NetworkRunner runner,
    NetworkBehaviour obj,
    byte * data ) [static]
```

Serializes a [NetworkBehaviour](#) into a byte array.

Parameters

<i>runner</i>	
<i>obj</i>	The NetworkBehaviour to be serialized.
<i>data</i>	The byte pointer to write the serialized data to.

Returns

The size of the serialized data in bytes.

6.148.2.31 NetworkUnwrap()

```
static NetworkBehaviour NetworkUnwrap (
    NetworkRunner runner,
    NetworkBehaviourId wrapper ) [static]
```

Converts a [NetworkBehaviourId](#) to a [NetworkBehaviour](#).

Parameters

<i>runner</i>	The NetworkRunner associated with the NetworkBehaviour .
<i>wrapper</i>	The NetworkBehaviourId to be converted.

Returns

The [NetworkBehaviour](#) represented by the [NetworkBehaviourId](#). If the [NetworkBehaviourId](#) is not valid or a [NetworkBehaviour](#) with the [NetworkBehaviourId](#) does not exist, returns null.

6.148.2.32 NetworkWrap()

```
static NetworkBehaviourId NetworkWrap (
    NetworkRunner runner,
    NetworkBehaviour obj ) [static]
```

Converts a [NetworkBehaviour](#) to a [NetworkBehaviourId](#).

Parameters

<i>runner</i>	The NetworkRunner associated with the NetworkBehaviour .
<i>obj</i>	The NetworkBehaviour to be converted.

Returns

A [NetworkBehaviourId](#) representing the [NetworkBehaviour](#). If the [NetworkBehaviour](#) is null or its [NetworkObject](#)'s Id is default, returns a default [NetworkBehaviourId](#).

6.148.2.33 operator NetworkBehaviourId()

```
static implicit operator NetworkBehaviourId (
    NetworkBehaviour behaviour ) [static]
```

Converts [NetworkBehaviour](#) to [NetworkBehaviourId](#)

Parameters

<i>behaviour</i>	NetworkBehaviour to convert
------------------	-----------------------------

Returns

NetworkBehaviourId representing the NetworkBehaviour

6.148.2.34 ReinterpretState< T >()

```
ref T ReinterpretState< T > (
    int offset = 0 )
```

Allows read and write access to the internal state buffer

Parameters

<i>offset</i>	The offset to generate a ref for, in integer words
---------------	--

Template Parameters

<i>T</i>	The type of the ref to generate
----------	---------------------------------

Returns

Reference to the location in memory defined by offset

Type Constraints

T: *unmanaged*

6.148.2.35 ReplicateTo() [1/2]

```
virtual bool ReplicateTo (
    PlayerRef player ) [protected], [virtual]
```

Determines whether to replicate the network behaviour to the specified player. This method can be overridden in derived classes to implement custom replication logic.

Parameters

<i>player</i>	The player to potentially replicate the network behaviour to.
---------------	---

Returns

True if the network behaviour should be replicated to the player, false otherwise. The default implementation always returns true.

6.148.2.36 ReplicateTo() [2/2]

```
void ReplicateTo (
    PlayerRef player,
    bool replicate )
```

Controls if this network behaviours state is replicated to a player or not

Parameters

<i>player</i>	The player to change replication status for
<i>replicate</i>	true = replicate, false = don't replicate

6.148.2.37 ReplicateToAll()

```
void ReplicateToAll (
    bool replicate )
```

Sets the default replicated state for this behaviour, this by default is true so calling ReplicateToAll(true) does nothing if ReplicateToAll(false) hasn't been called before

Parameters

<i>replicate</i>	The default state of this behaviour
------------------	-------------------------------------

6.148.2.38 ResetState()

```
void ResetState ( )
```

Resets the state of the object to the original state

6.148.2.39 Spawned()

```
virtual void Spawned ( ) [virtual]
```

Post spawn callback.

Implements [ISpawned](#).

Reimplemented in [NetworkTransform](#), and [NetworkMecanimAnimator](#).

6.148.2.40 TryGetSnapshotsBuffers()

```
bool TryGetSnapshotsBuffers (
    out NetworkBehaviourBuffer from,
    out NetworkBehaviourBuffer to,
    out float alpha )
```

Tries to get the snapshot buffers for this network behaviour.

Parameters

<i>from</i>	The buffer representing the state of the network behaviour at the start of the render frame.
<i>to</i>	The buffer representing the state of the network behaviour at the end of the render frame.
<i>alpha</i>	The interpolation factor between the start and end of the render frame.

Returns

True if the snapshot buffers are valid, false otherwise.

6.148.3 Member Data Documentation

6.148.3.1 offset

```
int offset
```

Gives access to the offset (in 32 bit words) and count (in 32 bit words) of this behaviour backing data

6.148.4 Property Documentation

6.148.4.1 ChangedTick

```
Tick ChangedTick [get]
```

The tick the data on this networked behaviour changed

6.148.4.2 DynamicWordCount

```
virtual ? int DynamicWordCount [get]
```

Override this value for custom memory allocations. This is for advanced use cases only, and cannot be used if [NetworkedAttribute](#) is used in the derived class.

6.148.4.3 HasInputAuthority

```
bool? HasInputAuthority [get]
```

Returns true if the [Simulation.LocalPlayer](#) of the associated [NetworkRunner](#) is the designated as Input Source for this network entity.

6.148.4.4 HasStateAuthority

```
bool? HasStateAuthority [get]
```

Returns true if the associated [NetworkRunner](#) is the State Source for this network entity.

6.148.4.5 Id

```
NetworkBehaviourId? Id [get]
```

The unique identifier for this network behaviour.

6.148.4.6 IsProxy

```
bool? IsProxy [get]
```

Returns true if the associated [NetworkRunner](#) is neither the Input nor State Authority for this network entity. It is recommended to use [!HasStateAuthority](#) or [!HasInputAuthority](#) when possible instead, as this check requires evaluating both authorities - and is therefore less performant than the individual checks.

6.148.4.7 StateBuffer

```
NetworkBehaviourBuffer StateBuffer [get]
```

Gets the state buffer associated with the network behaviour.

6.148.4.8 StateBufferIsValid

```
bool StateBufferIsValid [get]
```

Gets a value indicating whether the state buffer is valid.

6.149 NetworkBehaviour.ArrayReader< T > Struct Template Reference

Provides a reader for network arrays of type T.

Public Member Functions

- [NetworkArrayReadOnly< T > Read \(NetworkBehaviourBuffer first\)](#)
Reads a network array from the provided network behaviour buffer.

6.149.1 Detailed Description

Provides a reader for network arrays of type T.

Template Parameters

T	The type of elements in the network array.
---	--

6.149.2 Member Function Documentation

6.149.2.1 Read()

```
NetworkArrayReadOnly<T> Read (
    NetworkBehaviourBuffer first )
```

Reads a network array from the provided network behaviour buffer.

Parameters

<code>first</code>	The network behaviour buffer to read the network array from.
--------------------	--

Returns

A read-only view of the network array.

6.150 NetworkBehaviour.BehaviourReader< T > Struct Template Reference

Provides a reader for network behaviours of type T.

Public Member Functions

- **T Read (NetworkBehaviourBuffer first)**
Reads a network behaviour from the provided network behaviour buffer.
- **T Read (NetworkBehaviourBuffer first, NetworkBehaviourBuffer second)**

Public Attributes

- **T**
Reads two network behaviours from the provided network behaviour buffers.

6.150.1 Detailed Description

Provides a reader for network behaviours of type T.

Template Parameters

<code>T</code>	The type of the network behaviour.
----------------	------------------------------------

Type Constraints

`T : NetworkBehaviour`

6.150.2 Member Function Documentation

6.150.2.1 Read()

```
T Read (
    NetworkBehaviourBuffer first )
```

Reads a network behaviour from the provided network behaviour buffer.

Parameters

<i>first</i>	The network behaviour buffer to read the network behaviour from.
--------------	--

Returns

The network behaviour of type T read from the buffer. Returns null if the behaviour is not found.

6.150.3 Member Data Documentation

6.150.3.1 T

T

Reads two network behaviours from the provided network behaviour buffers.

Parameters

<i>first</i>	The first network behaviour buffer to read the network behaviour from.
<i>second</i>	The second network behaviour buffer to read the network behaviour from.

Returns

A tuple containing the two network behaviours of type T read from the buffers.

6.151 NetworkBehaviour.ChangeDetector Class Reference

Change detector for a [NetworkBehaviour](#)

Classes

- struct [Enumerable](#)
Struct representing a collection of changes detected in a [NetworkBehaviour](#).
- struct [Enumerator](#)
Enumerator for the collection of changes detected in a [NetworkBehaviour](#).

Public Types

- enum class [Source](#)
Enum representing the source of a [NetworkBehaviour](#)'s state.

Public Member Functions

- **Enumerable DetectChanges (NetworkBehaviour b, bool copyChanges=true)**
Detects changes in a NetworkBehaviour and returns an Enumerable of the changes.
- **Enumerable DetectChanges (NetworkBehaviour b, out NetworkBehaviourBuffer previous, out NetworkBehaviourBuffer current, bool copyChanges=true)**
Detects changes in a NetworkBehaviour and returns an Enumerable of the changes.
- **void Init (NetworkBehaviour networkBehaviour, Source source, bool copyInitial=true)**
Initializes the ChangeDetector for a given NetworkBehaviour.

6.151.1 Detailed Description

Change detector for a NetworkBehaviour

This class is used to detect changes in a NetworkBehaviour. It can be used to detect changes in a NetworkBehaviour between two snapshots, or between the current state and a snapshot.

6.151.2 Member Enumeration Documentation

6.151.2.1 Source

```
enum Source [strong]
```

Enum representing the source of a NetworkBehaviour's state.

Enumerator

SimulationState	The state is the current simulation state of the NetworkBehaviour.
SnapshotFrom	The state is from a previous snapshot of the NetworkBehaviour.
SnapshotTo	The state is from a future snapshot of the NetworkBehaviour.

6.151.3 Member Function Documentation

6.151.3.1 DetectChanges() [1/2]

```
Enumerable DetectChanges (
    NetworkBehaviour b,
    bool copyChanges = true )
```

Detects changes in a NetworkBehaviour and returns an Enumerable of the changes.

Parameters

<i>b</i>	The NetworkBehaviour instance to detect changes in.
<i>copyChanges</i>	Whether to copy the changes detected. Defaults to true.

Returns

An [Enumerable](#) of the changes detected in the [NetworkBehaviour](#).

6.151.3.2 DetectChanges() [2/2]

```
Enumerable DetectChanges (
    NetworkBehaviour b,
    out NetworkBehaviourBuffer previous,
    out NetworkBehaviourBuffer current,
    bool copyChanges = true )
```

Detects changes in a [NetworkBehaviour](#) and returns an [Enumerable](#) of the changes.

Parameters

<i>b</i>	The NetworkBehaviour instance to detect changes in.
<i>previous</i>	The previous state of the NetworkBehaviour .
<i>current</i>	The current state of the NetworkBehaviour .
<i>copyChanges</i>	Whether to copy the changes detected. Defaults to true.

Returns

An [Enumerable](#) of the changes detected in the [NetworkBehaviour](#).

6.151.3.3 Init()

```
void Init (
    NetworkBehaviour networkBehaviour,
    Source source,
    bool copyInitial = true )
```

Initializes the [ChangeDetector](#) for a given [NetworkBehaviour](#).

Parameters

<i>networkBehaviour</i>	The NetworkBehaviour instance to initialize the ChangeDetector for.
<i>source</i>	The source of the NetworkBehaviour 's state.
<i>copyInitial</i>	Whether to copy the initial state of the NetworkBehaviour . Defaults to true.

6.152 NetworkBehaviour.ChangeDetector.Enumerable Struct Reference

Struct representing a collection of changes detected in a [NetworkBehaviour](#).

Public Member Functions

- bool [Changed](#) (string name)
Checks if a property has changed.
- [Enumerator GetEnumerator \(\)](#)
Gets an enumerator for the collection of changes.

6.152.1 Detailed Description

Struct representing a collection of changes detected in a [NetworkBehaviour](#).

6.152.2 Member Function Documentation

6.152.2.1 Changed()

```
bool Changed (
    string name )
```

Checks if a property has changed.

Parameters

<code>name</code>	The name of the property to check.
-------------------	------------------------------------

Returns

True if the property has changed, false otherwise.

6.152.2.2 GetEnumerator()

```
Enumerator GetEnumerator ( )
```

Gets an enumerator for the collection of changes.

Returns

An [Enumerator](#) for the collection of changes.

6.153 NetworkBehaviour.ChangeDetector.Enumerator Struct Reference

Enumerator for the collection of changes detected in a NetworkBehaviour.

Public Member Functions

- bool [MoveNext \(\)](#)
Advances the enumerator to the next property in the array of changed properties.
- void [Reset \(\)](#)
Resets the enumerator to its initial state.

Properties

- string [Current \[get\]](#)
Gets the current property name in the array of changed properties.

6.153.1 Detailed Description

Enumerator for the collection of changes detected in a NetworkBehaviour.

6.153.2 Member Function Documentation

6.153.2.1 MoveNext()

```
bool MoveNext ( )
```

Advances the enumerator to the next property in the array of changed properties.

Returns

True if the enumerator was successfully advanced to the next property, false if the enumerator has passed the end of the array.

6.153.2.2 Reset()

```
void Reset ( )
```

Resets the enumerator to its initial state.

6.153.3 Property Documentation

6.153.3.1 Current

```
string Current [get]
```

Gets the current property name in the array of changed properties.

6.154 NetworkBehaviour.DictionaryReader< K, V > Struct Template Reference

Provides a reader for network dictionaries with keys of type K and values of type V.

Public Member Functions

- [NetworkDictionaryReadOnly< K, V > Read \(NetworkBehaviourBuffer first\)](#)
Reads a network dictionary from the provided network behaviour buffer.

6.154.1 Detailed Description

Provides a reader for network dictionaries with keys of type K and values of type V.

Template Parameters

<i>K</i>	The type of keys in the network dictionary.
<i>V</i>	The type of values in the network dictionary.

6.154.2 Member Function Documentation

6.154.2.1 Read()

```
NetworkDictionaryReadOnly<K, V> Read (
    NetworkBehaviourBuffer first )
```

Reads a network dictionary from the provided network behaviour buffer.

Parameters

<i>first</i>	The network behaviour buffer to read the network dictionary from.
--------------	---

Returns

A read-only view of the network dictionary.

6.155 NetworkBehaviour.LinkedListReader< T > Struct Template Reference

Provides a reader for network linked lists of type T.

Public Member Functions

- [NetworkLinkedListReadOnly< T > Read \(NetworkBehaviourBuffer first\)](#)

Reads a network linked list from the provided network behaviour buffer.

6.155.1 Detailed Description

Provides a reader for network linked lists of type T.

Template Parameters

<i>T</i>	The type of elements in the network linked list.
----------	--

6.155.2 Member Function Documentation

6.155.2.1 Read()

```
NetworkLinkedListReadOnly<T> Read (
    NetworkBehaviourBuffer first )
```

Reads a network linked list from the provided network behaviour buffer.

Parameters

<i>first</i>	The network behaviour buffer to read the network linked list from.
--------------	--

Returns

A read-only view of the network linked list.

6.156 NetworkBehaviour.PropertyReader< T > Struct Template Reference

Provides a reader for properties of type T in a network behaviour.

Public Member Functions

- [PropertyReader \(int offset\)](#)
Constructs a new `PropertyReader` with the provided offset.
- [T Read \(NetworkBehaviourBuffer first\)](#)
Reads a property of type T from the provided network behaviour buffer.
- [T Read \(NetworkBehaviourBuffer first, NetworkBehaviourBuffer second\)](#)

Public Attributes

- [T](#)
Reads a property of type T from the provided network behaviour buffers.

6.156.1 Detailed Description

Provides a reader for properties of type T in a network behaviour.

Template Parameters

<code>T</code>	The type of the property in the network behaviour. Must be unmanaged.
----------------	---

Type Constraints

`T : unmanaged`

6.156.2 Constructor & Destructor Documentation

6.156.2.1 PropertyReader()

```
PropertyReader (
    int offset )
```

Constructs a new `PropertyReader` with the provided offset.

Parameters

<i>offset</i>	The offset of the property in the network behaviour buffer.
---------------	---

6.156.3 Member Function Documentation

6.156.3.1 Read()

```
T Read (
    NetworkBehaviourBuffer first )
```

Reads a property of type T from the provided network behaviour buffer.

Parameters

<i>first</i>	The network behaviour buffer to read the property from.
--------------	---

Returns

The property of type T read from the buffer.

6.156.4 Member Data Documentation

6.156.4.1 T

T

Reads a property of type T from the provided network behaviour buffers.

Parameters

<i>first</i>	The first network behaviour buffer to read the property from.
<i>second</i>	The second network behaviour buffer to read the property from.

Returns

A tuple containing the property of type T read from the first and second buffers.

6.157 NetworkBehaviourBuffer Struct Reference

Provides low level accesss to data buffers that can be read using a NetworkBehaviour.Reader

Public Member Functions

- float [Read \(NetworkBehaviour.PropertyReader< float > reader\)](#)
Reads a float property from the buffer using the provided PropertyReader.
- Quaternion [Read \(NetworkBehaviour.PropertyReader< Quaternion > reader\)](#)
Reads a Quaternion property from the buffer using the provided PropertyReader.
- Vector2 [Read \(NetworkBehaviour.PropertyReader< Vector2 > reader\)](#)
Reads a Vector2 property from the buffer using the provided PropertyReader.
- Vector3 [Read \(NetworkBehaviour.PropertyReader< Vector3 > reader\)](#)
Reads a Vector3 property from the buffer using the provided PropertyReader.
- Vector4 [Read \(NetworkBehaviour.PropertyReader< Vector4 > reader\)](#)
Reads a Vector4 property from the buffer using the provided PropertyReader.
- T [Read< T > \(NetworkBehaviour.BehaviourReader< T > reader\)](#)
Reads a NetworkBehaviour from the buffer using the provided BehaviourReader.
- T [Read< T > \(NetworkBehaviour.PropertyReader< T > reader\)](#)
Reads a property from the buffer using the provided PropertyReader.
- unsafe T [ReinterpretState< T > \(int offset=0\)](#)
Reinterprets the state of the buffer at a given offset as a specific type.

Static Public Member Functions

- static implicit [operator bool \(NetworkBehaviourBuffer buffer\)](#)
Implicit conversion operator to bool. This allows a NetworkBehaviourBuffer instance to be used in conditions directly.

Properties

- int [Length \[get\]](#)
Gets the length of the buffer.
- int [this\[int index\] \[get\]](#)
Indexer to get the value at a specific index in the buffer.
- Tick [Tick \[get\]](#)
Gets the tick at which the buffer was created.
- bool [Valid \[get\]](#)
Gets a value indicating whether the buffer is valid. A buffer is considered valid if the pointer to the start of the buffer is not null and the length of the buffer is greater than 0.

6.157.1 Detailed Description

Provides low level accesss to data buffers that can be read using a NetworkBehaviour.Reader

6.157.2 Member Function Documentation

6.157.2.1 operator bool()

```
static implicit operator bool (
    NetworkBehaviourBuffer buffer ) [static]
```

Implicit conversion operator to bool. This allows a NetworkBehaviourBuffer instance to be used in conditions directly.

Parameters

<i>buffer</i>	The NetworkBehaviourBuffer instance to convert.
---------------	---

Returns

True if the buffer is valid, false otherwise.

6.157.2.2 Read() [1/5]

```
float Read (
    NetworkBehaviour.PropertyReader< float > reader )
```

Reads a float property from the buffer using the provided PropertyReader.

Parameters

<i>reader</i>	The PropertyReader to use for reading the float property.
---------------	---

Returns

The read float property.

6.157.2.3 Read() [2/5]

```
Quaternion Read (
    NetworkBehaviour.PropertyReader< Quaternion > reader )
```

Reads a Quaternion property from the buffer using the provided PropertyReader.

Parameters

<i>reader</i>	The PropertyReader to use for reading the Quaternion property.
---------------	--

Returns

The read Quaternion property.

6.157.2.4 Read() [3/5]

```
Vector2 Read (
    NetworkBehaviour.PropertyReader< Vector2 > reader )
```

Reads a Vector2 property from the buffer using the provided PropertyReader.

Parameters

<i>reader</i>	The PropertyReader to use for reading the Vector2 property.
---------------	---

Returns

The read Vector2 property.

6.157.2.5 Read() [4/5]

```
Vector3 Read (
    NetworkBehaviour.PropertyReader< Vector3 > reader )
```

Reads a Vector3 property from the buffer using the provided PropertyReader.

Parameters

<i>reader</i>	The PropertyReader to use for reading the Vector3 property.
---------------	---

Returns

The read Vector3 property.

6.157.2.6 Read() [5/5]

```
Vector4 Read (
    NetworkBehaviour.PropertyReader< Vector4 > reader )
```

Reads a Vector4 property from the buffer using the provided PropertyReader.

Parameters

<i>reader</i>	The PropertyReader to use for reading the Vector4 property.
---------------	---

Returns

The read Vector4 property.

6.157.2.7 Read< T >() [1/2]

```
T Read< T > (
    NetworkBehaviour.BehaviourReader< T > reader )
```

Reads a [NetworkBehaviour](#) from the buffer using the provided BehaviourReader.

Template Parameters

<i>T</i>	The type of NetworkBehaviour to read. Must be a subclass of NetworkBehaviour .
----------	--

Parameters

<i>reader</i>	The BehaviourReader to use for reading the NetworkBehaviour .
---------------	---

Returns

The read [NetworkBehaviour](#).

Type Constraints

T : [*NetworkBehaviour*](#)

6.157.2.8 Read< T >() [2/2]

```
T Read< T > (  
    NetworkBehaviour.PropertyReader< T > reader )
```

Reads a property from the buffer using the provided PropertyReader.

Template Parameters

<i>T</i>	The type of the property to read. Must be unmanaged.
----------	--

Parameters

<i>reader</i>	The PropertyReader to use for reading the property.
---------------	---

Returns

The read property.

Type Constraints

T : [*unmanaged*](#)

6.157.2.9 ReinterpretState< T >()

```
unsafe T ReinterpretState< T > (  
    int offset = 0 )
```

Reinterprets the state of the buffer at a given offset as a specific type.

Template Parameters

<i>T</i>	The type to reinterpret the state as. Must be unmanaged.
----------	--

Parameters

<i>offset</i>	The offset at which to start reinterpreting. Defaults to 0.
---------------	---

Returns

The state of the buffer at the given offset, reinterpreted as the specified type.

Type Constraints

T: *unmanaged*

6.157.3 Property Documentation

6.157.3.1 Length

```
int Length [get]
```

Gets the length of the buffer.

6.157.3.2 this[int index]

```
int this[int index] [get]
```

Indexer to get the value at a specific index in the buffer.

Parameters

<i>index</i>	The index to get the value from.
--------------	----------------------------------

Returns

The value at the specified index in the buffer.

6.157.3.3 Tick

```
Tick Tick [get]
```

Gets the tick at which the buffer was created.

6.157.3.4 Valid

```
bool Valid [get]
```

Gets a value indicating whether the buffer is valid. A buffer is considered valid if the pointer to the start of the buffer is not null and the length of the buffer is greater than 0.

6.158 NetworkBehaviourBufferInterpolator Struct Reference

The [NetworkBehaviourBufferInterpolator](#) struct is used to interpolate between two [NetworkBehaviourBuffer](#) instances. This is a read-only, ref struct, meaning it cannot be boxed and it can only be used on the stack.

Public Member Functions

- float [Angle](#) ([NetworkBehaviour.PropertyReader< Angle >](#) property)
Gets the interpolated angle of a property.
- float [Angle](#) (string property)
Gets the interpolated angle of a property.
- bool [Bool](#) ([NetworkBehaviour.PropertyReader< bool >](#) property)
Gets the interpolated boolean value of a property.
- bool [Bool](#) (string property)
Gets the interpolated boolean value of a property.
- float [Float](#) ([NetworkBehaviour.PropertyReader< float >](#) property)
Gets the interpolated float value of a property.
- float [Float](#) (string property)
Gets the interpolated float value of a property.
- int [Int](#) ([NetworkBehaviour.PropertyReader< int >](#) property)
Gets the interpolated integer value of a property.
- int [Int](#) (string property)
Gets the interpolated integer value of a property.
- [NetworkBehaviourBufferInterpolator](#) ([NetworkBehaviour](#) nb)
Constructor for the [NetworkBehaviourBufferInterpolator](#) struct.
- Quaternion [Quaternion](#) ([NetworkBehaviour.PropertyReader< Quaternion >](#) property)
Gets the interpolated Quaternion value of a property.
- Quaternion [Quaternion](#) (string property)
Gets the interpolated Quaternion value of a property.
- T [Select< T >](#) ([NetworkBehaviour.PropertyReader< T >](#) property)
Selects the interpolated value of a property.
- T [Select< T >](#) (string property)
Selects the interpolated value of a property by its name.
- Vector2 [Vector2](#) ([NetworkBehaviour.PropertyReader< Vector2 >](#) property)
Gets the interpolated Vector2 value of a property.
- Vector2 [Vector2](#) (string property)
Gets the interpolated Vector2 value of a property.

- Vector3 `Vector3 (NetworkBehaviour.PropertyReader< Vector3 > property)`
Gets the interpolated Vector3 value of a property.
- Vector3 `Vector3 (string property)`
Gets the interpolated Vector3 value of a property.
- Vector4 `Vector4 (NetworkBehaviour.PropertyReader< Vector4 > property)`
Gets the interpolated Vector4 value of a property.
- Vector4 `Vector4 (string property)`
Gets the interpolated Vector4 value of a property.

Static Public Member Functions

- static implicit `operator bool (NetworkBehaviourBufferInterpolator i)`
Implicit conversion operator to bool. This allows a NetworkBehaviourBufferInterpolator instance to be used in conditions directly.

Public Attributes

- readonly float `Alpha`
The interpolation factor, ranging from 0 to 1. This value is used to interpolate between the "from" and "to" states.
- readonly `NetworkBehaviour Behaviour`
The NetworkBehaviour instance that this interpolator is associated with.
- readonly `NetworkBehaviourBuffer From`
The NetworkBehaviourBuffer instance representing the "from" state for interpolation.
- readonly `NetworkBehaviourBuffer To`
The NetworkBehaviourBuffer instance representing the "to" state for interpolation.
- readonly bool `Valid`
A value indicating whether this interpolator is valid. An interpolator is considered valid if it has successfully retrieved the "from" and "to" buffers.

6.158.1 Detailed Description

The `NetworkBehaviourBufferInterpolator` struct is used to interpolate between two `NetworkBehaviourBuffer` instances. This is a read-only, ref struct, meaning it cannot be boxed and it can only be used on the stack.

6.158.2 Constructor & Destructor Documentation

6.158.2.1 `NetworkBehaviourBufferInterpolator()`

```
NetworkBehaviourBufferInterpolator (
    NetworkBehaviour nb )
```

Constructor for the `NetworkBehaviourBufferInterpolator` struct.

Parameters

<i>nb</i>	The NetworkBehaviour instance that this interpolator is associated with.
-----------	--

6.158.3 Member Function Documentation

6.158.3.1 Angle() [1/2]

```
float Angle (
    NetworkBehaviour.PropertyReader< Angle > property )
```

Gets the interpolated angle of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the angle of.
-----------------	--

Returns

The interpolated angle of the property.

6.158.3.2 Angle() [2/2]

```
float Angle (
    string property )
```

Gets the interpolated angle of a property.

Parameters

<i>property</i>	The name of the property to get the angle of.
-----------------	---

Returns

The interpolated angle of the property.

6.158.3.3 Bool() [1/2]

```
bool Bool (
    NetworkBehaviour.PropertyReader< bool > property )
```

Gets the interpolated boolean value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the boolean value of.
-----------------	--

Returns

The interpolated boolean value of the property.

6.158.3.4 Bool() [2/2]

```
bool Bool (
    string property )
```

Gets the interpolated boolean value of a property.

Parameters

<i>property</i>	The name of the property to get the boolean value of.
-----------------	---

Returns

The interpolated boolean value of the property.

6.158.3.5 Float() [1/2]

```
float Float (
    NetworkBehaviour.PropertyReader< float > property )
```

Gets the interpolated float value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the float value of.
-----------------	--

Returns

The interpolated float value of the property.

6.158.3.6 Float() [2/2]

```
float Float (
    string property )
```

Gets the interpolated float value of a property.

Parameters

<i>property</i>	The name of the property to get the float value of.
-----------------	---

Returns

The interpolated float value of the property.

6.158.3.7 Int() [1/2]

```
int Int (
    NetworkBehaviour.PropertyReader< int > property )
```

Gets the interpolated integer value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the integer value of.
-----------------	--

Returns

The interpolated integer value of the property.

6.158.3.8 Int() [2/2]

```
int Int (
    string property )
```

Gets the interpolated integer value of a property.

Parameters

<i>property</i>	The name of the property to get the integer value of.
-----------------	---

Returns

The interpolated integer value of the property.

6.158.3.9 operator bool()

```
static implicit operator bool (
    NetworkBehaviourBufferInterpolator i ) [static]
```

Implicit conversion operator to bool. This allows a [NetworkBehaviourBufferInterpolator](#) instance to be used in conditions directly.

Parameters

<i>i</i>	The NetworkBehaviourBufferInterpolator instance to convert.
----------	---

Returns

True if the interpolator is valid, false otherwise.

6.158.3.10 Quaternion() [1/2]

```
Quaternion Quaternion (
    NetworkBehaviour.PropertyReader< Quaternion > property )
```

Gets the interpolated Quaternion value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the Quaternion value of.
-----------------	---

Returns

The interpolated Quaternion value of the property.

6.158.3.11 Quaternion() [2/2]

```
Quaternion Quaternion (
    string property )
```

Gets the interpolated Quaternion value of a property.

Parameters

<i>property</i>	The name of the property to get the Quaternion value of.
-----------------	--

Returns

The interpolated Quaternion value of the property.

6.158.3.12 Select< T >() [1/2]

```
T Select< T > (  
    NetworkBehaviour.PropertyReader< T > property )
```

Selects the interpolated value of a property.

Template Parameters

<i>T</i>	The type of the property to select. Must be unmanaged.
----------	--

Parameters

<i>property</i>	The PropertyReader for the property to select.
-----------------	--

Returns

The interpolated value of the property.

Type Constraints

T : unmanaged

6.158.3.13 Select< T >() [2/2]

```
T Select< T > (  
    string property )
```

Selects the interpolated value of a property by its name.

Template Parameters

<i>T</i>	The type of the property to select. Must be unmanaged.
----------	--

Parameters

<i>property</i>	The name of the property to select.
-----------------	-------------------------------------

Returns

The interpolated value of the property.

Type Constraints

T : unmanaged

6.158.3.14 Vector2() [1/2]

```
Vector2 Vector2 (
    NetworkBehaviour.PropertyReader< Vector2 > property )
```

Gets the interpolated Vector2 value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the Vector2 value of.
-----------------	--

Returns

The interpolated Vector2 value of the property.

6.158.3.15 Vector2() [2/2]

```
Vector2 Vector2 (
    string property )
```

Gets the interpolated Vector2 value of a property.

Parameters

<i>property</i>	The name of the property to get the Vector2 value of.
-----------------	---

Returns

The interpolated Vector2 value of the property.

6.158.3.16 Vector3() [1/2]

```
Vector3 Vector3 (
    NetworkBehaviour.PropertyReader< Vector3 > property )
```

Gets the interpolated Vector3 value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the Vector3 value of.
-----------------	--

Returns

The interpolated Vector3 value of the property.

6.158.3.17 Vector3() [2/2]

```
Vector3 Vector3 (
    string property )
```

Gets the interpolated Vector3 value of a property.

Parameters

<i>property</i>	The name of the property to get the Vector3 value of.
-----------------	---

Returns

The interpolated Vector3 value of the property.

6.158.3.18 Vector4() [1/2]

```
Vector4 Vector4 (
    NetworkBehaviour.PropertyReader< Vector4 > property )
```

Gets the interpolated Vector4 value of a property.

Parameters

<i>property</i>	The PropertyReader for the property to get the Vector4 value of.
-----------------	--

Returns

The interpolated Vector4 value of the property.

6.158.3.19 Vector4() [2/2]

```
Vector4 Vector4 (
    string property )
```

Gets the interpolated Vector4 value of a property.

Parameters

<i>property</i>	The name of the property to get the Vector4 value of.
-----------------	---

Returns

The interpolated Vector4 value of the property.

6.158.4 Member Data Documentation

6.158.4.1 Alpha

```
readonly float Alpha
```

The interpolation factor, ranging from 0 to 1. This value is used to interpolate between the "from" and "to" states.

6.158.4.2 Behaviour

```
readonly NetworkBehaviour Behaviour
```

The [NetworkBehaviour](#) instance that this interpolator is associated with.

6.158.4.3 From

```
readonly NetworkBehaviourBuffer From
```

The [NetworkBehaviourBuffer](#) instance representing the "from" state for interpolation.

6.158.4.4 To

```
readonly NetworkBehaviourBuffer To
```

The [NetworkBehaviourBuffer](#) instance representing the "to" state for interpolation.

6.158.4.5 Valid

```
readonly bool Valid
```

A value indicating whether this interpolator is valid. An interpolator is considered valid if it has successfully retrieved the "from" and "to" buffers.

6.159 NetworkBehaviourId Struct Reference

Represents the unique identifier for a [NetworkBehaviour](#) instance.

Inherits [INetworkStruct](#), and [IEquatable< NetworkBehaviourId >](#).

Public Member Functions

- bool [Equals \(NetworkBehaviourId other\)](#)
Checks if this NetworkBehaviourId is equal to another NetworkBehaviourId. Two NetworkBehaviourIds are equal if their Objects and Behaviours are equal.
- override bool [Equals \(object obj\)](#)
Checks if this NetworkBehaviourId is equal to another object. The object is considered equal if it is a NetworkBehaviourId and its Object and Behaviour are equal to this NetworkBehaviourId's.
- override int [GetHashCode \(\)](#)
Returns a hash code for this NetworkBehaviourId. The hash code is computed based on the Object's hash code and the Behaviour.
- override string [ToString \(\)](#)
Returns a string representation of the NetworkBehaviourId. The string representation is in the format: [Object←:{Object}, Behaviour:{Behaviour}].

Static Public Member Functions

- static bool [operator!= \(NetworkBehaviourId a, NetworkBehaviourId b\)](#)
Determines whether two NetworkBehaviourId instances are not equal. Two NetworkBehaviourId instances are considered not equal if their Objects or Behaviours are not equal.
- static bool [operator== \(NetworkBehaviourId a, NetworkBehaviourId b\)](#)
Determines whether two NetworkBehaviourId instances are equal. Two NetworkBehaviourId instances are considered equal if their Objects and Behaviours are equal.

Public Attributes

- int [Behaviour](#)
The identifier for the behaviour within the object.
- [NetworkId Object](#)
The NetworkId of the object this behaviour belongs to.

Static Public Attributes

- const int [SIZE](#) = NetworkId.SIZE + sizeof(int)
Size of the NetworkBehaviourId structure in bytes.

Properties

- bool [IsValid \[get\]](#)
Checks if the NetworkBehaviourId is valid. A NetworkBehaviourId is valid if its Object is valid and its Behaviour is non-negative.
- static [NetworkBehaviourId None \[get\]](#)
Returns a new NetworkBehaviourId with default values.

6.159.1 Detailed Description

Represents the unique identifier for a [NetworkBehaviour](#) instance.

6.159.2 Member Function Documentation

6.159.2.1 Equals() [1/2]

```
bool Equals (
    NetworkBehaviourId other )
```

Checks if this [NetworkBehaviourId](#) is equal to another [NetworkBehaviourId](#). Two [NetworkBehaviourId](#)s are equal if their Objects and Behaviours are equal.

Parameters

<i>other</i>	The other NetworkBehaviourId to compare with.
--------------	---

Returns

True if the [NetworkBehaviourId](#)s are equal, false otherwise.

6.159.2.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Checks if this [NetworkBehaviourId](#) is equal to another object. The object is considered equal if it is a [NetworkBehaviourId](#) and its Object and Behaviour are equal to this [NetworkBehaviourId](#)'s.

Parameters

<i>obj</i>	The object to compare with.
------------	-----------------------------

Returns

True if the object is a [NetworkBehaviourId](#) and is equal to this, false otherwise.

6.159.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns a hash code for this [NetworkBehaviourId](#). The hash code is computed based on the Object's hash code and the [Behaviour](#).

Returns

A hash code for this [NetworkBehaviourId](#).

6.159.2.4 operator"!=()

```
static bool operator!= (
    NetworkBehaviourId a,
    NetworkBehaviourId b ) [static]
```

Determines whether two [NetworkBehaviourId](#) instances are not equal. Two [NetworkBehaviourId](#) instances are considered not equal if their Objects or Behaviours are not equal.

Parameters

<i>a</i>	The first NetworkBehaviourId to compare.
<i>b</i>	The second NetworkBehaviourId to compare.

Returns

True if the [NetworkBehaviourId](#) instances are not equal, false otherwise.

6.159.2.5 operator==()

```
static bool operator== (
    NetworkBehaviourId a,
    NetworkBehaviourId b ) [static]
```

Determines whether two [NetworkBehaviourId](#) instances are equal. Two [NetworkBehaviourId](#) instances are considered equal if their Objects and Behaviours are equal.

Parameters

<i>a</i>	The first NetworkBehaviourId to compare.
<i>b</i>	The second NetworkBehaviourId to compare.

Returns

True if the [NetworkBehaviourId](#) instances are equal, false otherwise.

6.159.2.6 ToString()

```
override string ToString ( )
```

Returns a string representation of the [NetworkBehaviourId](#). The string representation is in the format: [Object←:{Object}, Behaviour:{Behaviour}].

Returns

A string representation of the [NetworkBehaviourId](#).

6.159.3 Member Data Documentation

6.159.3.1 Behaviour

```
int Behaviour
```

The identifier for the behaviour within the object.

6.159.3.2 Object

```
NetworkId Object
```

The [NetworkId](#) of the object this behaviour belongs to.

6.159.3.3 SIZE

```
const int SIZE = NetworkId.SIZE + sizeof(int) [static]
```

Size of the [NetworkBehaviourId](#) structure in bytes.

6.159.4 Property Documentation

6.159.4.1 IsValid

```
bool IsValid [get]
```

Checks if the [NetworkBehaviourId](#) is valid. A [NetworkBehaviourId](#) is valid if its Object is valid and its Behaviour is non-negative.

6.159.4.2 None

`NetworkBehaviourId` `None` [static], [get]

Returns a new `NetworkBehaviourId` with default values.

6.160 NetworkBehaviourUtils Class Reference

This static class provides utility methods for working with `NetworkBehaviour` objects.

Classes

- struct `ArrayInitializer`
A utility structure for initializing `NetworkArray` and `NetworkLinkedList` with inline initialization.
- struct `DictionaryInitializer`
A utility structure for initializing `NetworkDictionary` with inline initialization.
- struct `MetaData`
This structure holds metadata for a `NetworkBehaviour` object.

Static Public Member Functions

- static `T[] CloneArray< T >` (`T[] array`)
Creates a new array that is a clone of the specified array.
- static void `CopyFromNetworkArray< T >` (`NetworkArray< T >` networkArray, ref `T[] dstArray`)
Copies the values from a `NetworkArray` to a destination array.
- static void `CopyFromNetworkDictionary< D, K, V >` (`NetworkDictionary< K, V >` networkDictionary, ref `D` dictionary)
Copies the values from a `NetworkDictionary` to a destination dictionary.
- static void `CopyFromNetworkList< T >` (`NetworkLinkedList< T >` networkList, ref `T[] dstArray`)
Copies the values from a `NetworkLinkedList` to a destination array.
- static `MetaData GetMetaData` (`Type type`)
Retrieves the metadata for a given type.
- static int `GetRpcStaticIndexOrThrow` (`string key`)
Retrieves the index of a static RPC (Remote Procedure Call) based on its key.
- static int `GetStaticWordCount` (`Type type`)
Retrieves the static word count for a given type. If the word count for the type has not been computed yet, it is computed and stored.
- static int `GetWordCount` (`NetworkBehaviour behaviour`)
Retrieves the word count for a given `NetworkBehaviour`. If the `NetworkBehaviour` has a dynamic word count, it is returned. Otherwise, the static word count for the type of the `NetworkBehaviour` is returned.
- static bool `HasStaticWordCount` (`Type type`)
Checks if a given type has a static word count. A type has a static word count if it is a subclass of `NetworkBehaviour` and its word count is non-negative.
- static void `InitializeNetworkArray< T >` (`NetworkArray< T >` networkArray, `T[] sourceArray`, `string name`)
Initializes a `NetworkArray` with the values from a source array.
- static void `InitializeNetworkDictionary< D, K, V >` (`NetworkDictionary< K, V >` networkDictionary, `D dictionary`, `string name`)
Initializes a `NetworkDictionary` with the values from a source dictionary.

- static void [InitializeNetworkList< T >](#) ([NetworkLinkedList< T >](#) networkList, T[] sourceArray, string name)
Initializes a [NetworkLinkedList](#) with the values from a source array.
- static void [InternalOnDestroy](#) ([SimulationBehaviour](#) obj)
This method is not meant to be called directly. Calls are injected by the Weaver.
- static void [InternalOnDisable](#) ([SimulationBehaviour](#) obj)
This method is not meant to be called directly. Calls are injected by the Weaver.
- static void [InternalOnEnable](#) ([SimulationBehaviour](#) obj)
This method is not meant to be called directly. Calls are injected by the Weaver.
- static [SerializableDictionary< K, V >](#) [MakeSerializableDictionary< K, V >](#) ([Dictionary< K, V >](#) dictionary)
Wraps a Dictionary in a [SerializableDictionary](#).
- static void [NotifyLocalSimulationNotAllowedToSendRpc](#) (string rpc, [NetworkObject](#) obj, int sources)
Logs an error message indicating that a local simulation is not allowed to send a specific RPC.
- static void [NotifyLocalTargetedRpcCulled](#) ([PlayerRef](#) player, string methodName)
Logs a warning message indicating that a local targeted RPC was culled.
- static void [NotifyNetworkUnwrapFailed< T >](#) (T wrapper, Type valueType)
Logs a warning message indicating that a network unwrap operation failed.
- static void [NotifyNetworkWrapFailed< T >](#) (T value)
Logs a warning message indicating that a network wrap operation failed.
- static void [NotifyNetworkWrapFailed< T >](#) (T value, Type wrapperType)
Logs a warning message indicating that a network wrap operation failed for a specific wrapper type.
- static void [NotifyRpcPayloadSizeExceeded](#) (string rpc, int size)
Logs an error message indicating that the target of a Remote Procedure Call (RPC) payload is too large.
- static void [NotifyRpcTargetUnreachable](#) ([PlayerRef](#) player, string rpc)
Logs an error message indicating that the target of a Remote Procedure Call (RPC) is not reachable.
- static void [RegisterMetaData](#) (Type type)
Registers the metadata for a given type. This method checks if the type has an override for the "ReplicateTo" method and stores this information in the metadata. If the metadata for the type already exists, this method does nothing.
- static void [RegisterRpcInvokeDelegates](#) (Type type)
Registers the RPC (Remote Procedure Call) invoke delegates for a given type. This method is only available when the FUSION_UNITY compilation symbol is defined.
- static bool [ShouldRegisterRpcInvokeDelegates](#) (Type type)
Determines whether the RPC (Remote Procedure Call) invoke delegates should be registered for a given type.
- static void [ThrowIfBehaviourNotInitialized](#) ([NetworkBehaviour](#) behaviour)
Checks if the [NetworkBehaviour](#) object is initialized and throws an exception if it is not.
- static bool [TryGetRpcInvokeDelegateArray](#) (Type type, out [RpclinevokeData\[\]](#) delegates)
Tries to get the RPC (Remote Procedure Call) invoke delegate array for a given type.
- static bool [TryGetRpcStaticInvokeDelegate](#) (int index, out [RpclinevokeDelegate](#) del)
Tries to get the static RPC (Remote Procedure Call) invoke delegate based on its index.

Static Public Attributes

- static bool [InvokeRpc](#) = false
A static field that determines whether to invoke RPC (Remote Procedure Call). When set to true, RPCs are invoked. When set to false, RPCs are not invoked. Default value is false.

6.160.1 Detailed Description

This static class provides utility methods for working with [NetworkBehaviour](#) objects.

6.160.2 Member Function Documentation

6.160.2.1 CloneArray< T >()

```
static T [ ] CloneArray< T > (
    T[ ] array ) [static]
```

Creates a new array that is a clone of the specified array.

Template Parameters

<i>T</i>	The type of the elements in the array.
----------	--

Parameters

<i>array</i>	The array to clone.
--------------	---------------------

Returns

A new array that is a clone of the specified array. If the specified array is null, an empty array is returned.

6.160.2.2 CopyFromNetworkArray< T >()

```
static void CopyFromNetworkArray< T > (
    NetworkArray< T > networkArray,
    ref T[ ] dstArray ) [static]
```

Copies the values from a [NetworkArray](#) to a destination array.

Template Parameters

<i>T</i>	The type of the elements in the NetworkArray and destination array.
----------	---

Parameters

<i>networkArray</i>	The NetworkArray from which to copy the values.
<i>dstArray</i>	The destination array to which to copy the values.

If the length of the destination array is not equal to the length of the [NetworkArray](#), a new array of the correct length is created and assigned to the destination array.

Type Constraints

T : *unmanaged*

6.160.2.3 CopyFromNetworkDictionary< D, K, V >()

```
static void CopyFromNetworkDictionary< D, K, V > (
    NetworkDictionary< K, V > networkDictionary,
    ref D dictionary ) [static]
```

Copies the values from a [NetworkDictionary](#) to a destination dictionary.

Template Parameters

<i>D</i>	The type of the destination dictionary. Must implement IDictionary{K, V} and have a parameterless constructor.
<i>K</i>	The type of the keys in the NetworkDictionary and destination dictionary. Must be unmanaged.
<i>V</i>	The type of the values in the NetworkDictionary and destination dictionary. Must be unmanaged.

Parameters

<i>networkDictionary</i>	The NetworkDictionary from which to copy the values.
<i>dictionary</i>	The destination dictionary to which to copy the values. If the destination dictionary is null, a new dictionary of type <i>D</i> is created.

Type Constraints

D : [IDictionary](#)

D : [K](#)

D : [V](#)

D : [new\(\)](#)

K : [unmanaged](#)

V : [unmanaged](#)

6.160.2.4 CopyFromNetworkList< T >()

```
static void CopyFromNetworkList< T > (
    NetworkLinkedList< T > networkList,
    ref T[] dstArray ) [static]
```

Copies the values from a [NetworkLinkedList](#) to a destination array.

Template Parameters

<i>T</i>	The type of the elements in the NetworkLinkedList and destination array.
----------	--

Parameters

<i>networkList</i>	The NetworkLinkedList from which to copy the values.
<i>dstArray</i>	The destination array to which to copy the values. If the length of the destination array is not equal to the count of the NetworkLinkedList , a new array of the correct length is created and assigned to the destination array.

Type Constraints

T : **unmanaged**

6.160.2.5 GetMetaData()

```
static MetaData GetMetaData (
    Type type ) [static]
```

Retrieves the metadata for a given type.

Parameters

<i>type</i>	The type for which to retrieve the metadata.
-------------	--

Returns

The metadata for the given type if it exists; otherwise, the default value.

6.160.2.6 GetRpcStaticIndexOrThrow()

```
static int GetRpcStaticIndexOrThrow (
    string key ) [static]
```

Retrieves the index of a static RPC (Remote Procedure Call) based on its key.

Parameters

<i>key</i>	The key of the static RPC.
------------	----------------------------

Returns

The index of the static RPC if it exists.

Exceptions

<i>System.ArgumentOutOfRangeException</i>	Thrown when the static RPC does not exist.
---	--

6.160.2.7 GetStaticWordCount()

```
static int GetStaticWordCount (
    Type type ) [static]
```

Retrieves the static word count for a given type. If the word count for the type has not been computed yet, it is computed and stored.

Parameters

<i>type</i>	The type for which to retrieve the static word count.
-------------	---

Returns

The static word count for the given type.

Exceptions

<i>System.ArgumentException</i>	Thrown when the provided type is not a subclass of NetworkBehaviour .
<i>System.InvalidOperationException</i>	Thrown when the provided type does not have a weaved attribute or its word count is negative.

6.160.2.8 GetWordCount()

```
static int GetWordCount (
    NetworkBehaviour behaviour ) [static]
```

Retrieves the word count for a given [NetworkBehaviour](#). If the [NetworkBehaviour](#) has a dynamic word count, it is returned. Otherwise, the static word count for the type of the [NetworkBehaviour](#) is returned.

Parameters

<i>behaviour</i>	The NetworkBehaviour for which to retrieve the word count.
------------------	--

Returns

The word count for the given [NetworkBehaviour](#).

Exceptions

<i>System.InvalidOperationException</i>	Thrown when the dynamic word count or the static word count is negative.
---	--

6.160.2.9 HasStaticWordCount()

```
static bool HasStaticWordCount (
    Type type ) [static]
```

Checks if a given type has a static word count. A type has a static word count if it is a subclass of [NetworkBehaviour](#) and its word count is non-negative.

Parameters

<i>type</i>	The type to check.
-------------	--------------------

Returns

True if the type has a static word count, false otherwise.

Exceptions

System.ArgumentException	Thrown when the provided type is not a subclass of NetworkBehaviour .
System.InvalidOperationException	Thrown when the provided type does not have a weaved attribute.

6.160.2.10 InitializeNetworkArray< T >()

```
static void InitializeNetworkArray< T > (
    NetworkArray< T > networkArray,
    T[] sourceArray,
    string name ) [static]
```

Initializes a [NetworkArray](#) with the values from a source array.

Template Parameters

<i>T</i>	The type of the elements in the NetworkArray and source array.
----------	--

Parameters

<i>networkArray</i>	The NetworkArray to initialize.
<i>sourceArray</i>	The source array from which to copy the values.
<i>name</i>	The name of the NetworkArray for logging purposes.

If the length of the source array is greater than the length of the [NetworkArray](#), a warning is logged and only the first elements up to the length of the [NetworkArray](#) are copied.

Type Constraints

T : **unmanaged**

6.160.2.11 InitializeNetworkDictionary< D, K, V >()

```
static void InitializeNetworkDictionary< D, K, V > (
    NetworkDictionary< K, V > networkDictionary,
    D dictionary,
    string name ) [static]
```

Initializes a [NetworkDictionary](#) with the values from a source dictionary.

Template Parameters

<i>D</i>	The type of the source dictionary. Must implement IDictionary{K, V} .
<i>K</i>	The type of the keys in the NetworkDictionary and source dictionary. Must be unmanaged.
<i>V</i>	The type of the values in the NetworkDictionary and source dictionary. Must be unmanaged.

Parameters

<i>networkDictionary</i>	The NetworkDictionary to initialize.
<i>dictionary</i>	The source dictionary from which to copy the values.
<i>name</i>	The name of the NetworkDictionary for logging purposes.

If the count of the source dictionary is greater than the capacity of the [NetworkDictionary](#), a warning is logged and only the first elements up to the capacity of the [NetworkDictionary](#) are copied.

Type Constraints

D : IDictionary

D : K

D : V

K : unmanaged

V : unmanaged

6.160.2.12 InitializeNetworkList< T >()

```
static void InitializeNetworkList< T > (
    NetworkLinkedList< T > networkList,
    T[ ] sourceArray,
    string name ) [static]
```

Initializes a [NetworkLinkedList](#) with the values from a source array.

Template Parameters

<i>T</i>	The type of the elements in the NetworkLinkedList and source array. Must be unmanaged.
----------	--

Parameters

<i>networkList</i>	The NetworkLinkedList to initialize.
<i>sourceArray</i>	The source array from which to copy the values.
<i>name</i>	The name of the NetworkLinkedList for logging purposes.

If the length of the source array is greater than the capacity of the [NetworkLinkedList](#), a warning is logged and only the first elements up to the capacity of the [NetworkLinkedList](#) are copied.

Type Constraints

T : *unmanaged*

6.160.2.13 InternalOnDestroy()

```
static void InternalOnDestroy (
    SimulationBehaviour obj ) [static]
```

This method is not meant to be called directly. Calls are injected by the Weaver.

Parameters

<i>obj</i>	SimulationBehaviour object.
------------	---

6.160.2.14 InternalOnDisable()

```
static void InternalOnDisable (
    SimulationBehaviour obj ) [static]
```

This method is not meant to be called directly. Calls are injected by the Weaver.

Parameters

<i>obj</i>	SimulationBehaviour object.
------------	---

6.160.2.15 InternalOnEnable()

```
static void InternalOnEnable (
    SimulationBehaviour obj ) [static]
```

This method is not meant to be called directly. Calls are injected by the Weaver.

Parameters

<i>obj</i>	SimulationBehaviour object.
------------	---

6.160.2.16 MakeSerializableDictionary< K, V >()

```
static SerializableDictionary<K, V> MakeSerializableDictionary< K, V > (
    Dictionary< K, V > dictionary ) [static]
```

Wraps a Dictionary in a [SerializableDictionary](#).

Template Parameters

<i>K</i>	The type of the keys in the dictionary. Must be unmanaged.
<i>V</i>	The type of the values in the dictionary. Must be unmanaged.

Parameters

<i>dictionary</i>	The dictionary to wrap.
-------------------	-------------------------

Returns

A [SerializableDictionary](#) that wraps the specified dictionary.

Type Constraints

K : *unmanaged*

V : *unmanaged*

6.160.2.17 NotifyLocalSimulationNotAllowedToSendRpc()

```
static void NotifyLocalSimulationNotAllowedToSendRpc (
    string rpc,
    NetworkObject obj,
    int sources ) [static]
```

Logs an error message indicating that a local simulation is not allowed to send a specific RPC.

Parameters

<i>rpc</i>	The name of the RPC that was attempted.
<i>obj</i>	The network object on which the RPC was attempted.
<i>sources</i>	The sources from which the RPC was attempted.

6.160.2.18 NotifyLocalTargetedRpcCulled()

```
static void NotifyLocalTargetedRpcCulled (
    PlayerRef player,
    string methodName ) [static]
```

Logs a warning message indicating that a local targeted RPC was culled.

Parameters

<i>player</i>	The player reference for which the RPC was culled.
<i>methodName</i>	The name of the method that was culled.

6.160.2.19 NotifyNetworkUnwrapFailed< T >()

```
static void NotifyNetworkUnwrapFailed< T > (
    T wrapper,
    Type valueType ) [static]
```

Logs a warning message indicating that a network unwrap operation failed.

Template Parameters

<i>T</i>	The type of the wrapper that failed to be unwrapped.
----------	--

Parameters

<i>wrapper</i>	The wrapper that failed to be unwrapped.
<i>valueType</i>	The type that was attempted to be obtained from the unwrap operation.

6.160.2.20 NotifyNetworkWrapFailed< T >() [1/2]

```
static void NotifyNetworkWrapFailed< T > (
    T value ) [static]
```

Logs a warning message indicating that a network wrap operation failed.

Template Parameters

<i>T</i>	The type of the value that failed to be wrapped.
----------	--

Parameters

<i>value</i>	The value that failed to be wrapped.
--------------	--------------------------------------

6.160.2.21 NotifyNetworkWrapFailed< T >() [2/2]

```
static void NotifyNetworkWrapFailed< T > (
    T value,
    Type wrapperType ) [static]
```

Logs a warning message indicating that a network wrap operation failed for a specific wrapper type.

Template Parameters

<i>T</i>	The type of the value that failed to be wrapped.
----------	--

Parameters

<i>value</i>	The value that failed to be wrapped.
<i>wrapperType</i>	The type that was attempted to be used as a wrapper.

6.160.2.22 NotifyRpcPayloadSizeExceeded()

```
static void NotifyRpcPayloadSizeExceeded (
    string rpc,
    int size ) [static]
```

Logs an error message indicating that the target of a Remote Procedure Call (RPC) payload is too large.

Parameters

<i>size</i>	Size of the payload.
<i>rpc</i>	The name of the RPC that was attempted.

6.160.2.23 NotifyRpcTargetUnreachable()

```
static void NotifyRpcTargetUnreachable (
    PlayerRef player,
    string rpc ) [static]
```

Logs an error message indicating that the target of a Remote Procedure Call (RPC) is not reachable.

Parameters

<i>player</i>	The player reference that is not reachable.
<i>rpc</i>	The name of the RPC that was attempted.

6.160.2.24 RegisterMetaData()

```
static void RegisterMetaData (
    Type type ) [static]
```

Registers the metadata for a given type. This method checks if the type has an override for the "ReplicateTo" method and stores this information in the metadata. If the metadata for the type already exists, this method does nothing.

Parameters

<i>type</i>	The type for which to register the metadata.
-------------	--

6.160.2.25 RegisterRpcInvokeDelegates()

```
static void RegisterRpcInvokeDelegates (
    Type type ) [static]
```

Registers the RPC (Remote Procedure Call) invoke delegates for a given type. This method is only available when the FUSION_UNITY compilation symbol is defined.

Parameters

<i>type</i>	The type for which to register the RPC invoke delegates.
-------------	--

Exceptions

<i>System.ArgumentException</i>	Thrown when the provided type is not a subclass of NetworkBehaviour .
<i>System.InvalidOperationException</i>	Thrown when the provided type does not have a weaved attribute or its word count is negative.

6.160.2.26 ShouldRegisterRpcInvokeDelegates()

```
static bool ShouldRegisterRpcInvokeDelegates (
    Type type ) [static]
```

Determines whether the RPC (Remote Procedure Call) invoke delegates should be registered for a given type.

Parameters

<i>type</i>	The type for which to check the registration of RPC invoke delegates.
-------------	---

Returns

True if the RPC invoke delegates should be registered for the type, false otherwise.

6.160.2.27 ThrowIfBehaviourNotInitialized()

```
static void ThrowIfBehaviourNotInitialized (
    NetworkBehaviour behaviour ) [static]
```

Checks if the [NetworkBehaviour](#) object is initialized and throws an exception if it is not.

Parameters

<i>behaviour</i>	The NetworkBehaviour object to check.
------------------	---

Exceptions

<i>System.InvalidOperationException</i>	Thrown when the NetworkBehaviour object is not initialized.
---	---

6.160.2.28 TryGetRpcInvokeDelegateArray()

```
static bool TryGetRpcInvokeDelegateArray (
    Type type,
    out RpcInvokeData[] delegates ) [static]
```

Tries to get the RPC (Remote Procedure Call) invoke delegate array for a given type.

Parameters

<i>type</i>	The type for which to get the RPC invoke delegate array.
<i>delegates</i>	When this method returns, contains the RPC invoke delegate array if the type has one; otherwise, null.

Returns

true if the type has an RPC invoke delegate array; otherwise, false.

6.160.2.29 TryGetRpcStaticInvokeDelegate()

```
static bool TryGetRpcStaticInvokeDelegate (
    int index,
    out RpcStaticInvokeDelegate del ) [static]
```

Tries to get the static RPC (Remote Procedure Call) invoke delegate based on its index.

Parameters

<i>index</i>	The index of the static RPC.
<i>del</i>	When this method returns, contains the static RPC invoke delegate if it exists; otherwise, default.

Returns

True if the static RPC invoke delegate exists; otherwise, false.

6.160.3 Member Data Documentation

6.160.3.1 InvokeRpc

```
bool InvokeRpc = false [static]
```

A static field that determines whether to invoke RPC (Remote Procedure Call). When set to true, RPCs are invoked. When set to false, RPCs are not invoked. Default value is false.

6.161 NetworkBehaviourUtils.ArrayInitializer< T > Struct Template Reference

A utility structure for initializing [NetworkArray](#) and [NetworkLinkedList](#) with inline initialization.

Static Public Member Functions

- static implicit [operator NetworkArray< T >](#) ([ArrayInitializer< T >](#) arr)
Implicitly converts an [ArrayInitializer](#) to a [NetworkArray](#).
- static implicit [operator NetworkLinkedList< T >](#) ([ArrayInitializer< T >](#) arr)
Implicitly converts an [ArrayInitializer](#) to a [NetworkLinkedList](#).

6.161.1 Detailed Description

A utility structure for initializing [NetworkArray](#) and [NetworkLinkedList](#) with inline initialization.

Template Parameters

<i>T</i>	The type of the elements in the NetworkArray and NetworkLinkedList .
----------	--

6.161.2 Member Function Documentation

6.161.2.1 operator NetworkArray< T >()

```
static implicit operator NetworkArray< T > (
    ArrayInitializer< T > arr ) [static]
```

Implicitly converts an [ArrayInitializer](#) to a [NetworkArray](#).

Parameters

<i>arr</i>	The ArrayInitializer to convert.
------------	--

Returns

A [NetworkArray](#) initialized with the values from the [ArrayInitializer](#).

Exceptions

<i>System.NotImplementedException</i>	Thrown always as this method is meant to be used only for [Networked] properties inline initialization.
---------------------------------------	---

6.161.2.2 operator NetworkLinkedList< T >()

```
static implicit operator NetworkLinkedList< T > (
    ArrayInitializer< T > arr ) [static]
```

Implicitly converts an [ArrayInitializer](#) to a [NetworkLinkedList](#).

Parameters

<i>arr</i>	The ArrayInitializer to convert.
------------	--

Returns

A [NetworkLinkedList](#) initialized with the values from the [ArrayInitializer](#).

Exceptions

<code>System.NotImplementedException</code>	Thrown always as this method is meant to be used only for [Networked] properties inline initialization.
---	---

6.162 NetworkBehaviourUtils.DictionaryInitializer< K, V > Struct Template Reference

A utility structure for initializing [NetworkDictionary](#) with inline initialization.

Static Public Member Functions

- static implicit [operator NetworkDictionary< K, V > \(DictionaryInitializer< K, V > arr\)](#)
Implicitly converts a [DictionaryInitializer](#) to a [NetworkDictionary](#).

6.162.1 Detailed Description

A utility structure for initializing [NetworkDictionary](#) with inline initialization.

Template Parameters

<code>K</code>	The type of the keys in the NetworkDictionary .
<code>V</code>	The type of the values in the NetworkDictionary .

6.162.2 Member Function Documentation

6.162.2.1 operator NetworkDictionary< K, V >()

```
static implicit operator NetworkDictionary< K, V > (
    DictionaryInitializer< K, V > arr ) [static]
```

Implicitly converts a [DictionaryInitializer](#) to a [NetworkDictionary](#).

Parameters

<code>arr</code>	The DictionaryInitializer to convert.
------------------	---

Returns

A [NetworkDictionary](#) initialized with the values from the [DictionaryInitializer](#).

Exceptions

<code>System.NotImplementedException</code>	Thrown always as this method is meant to be used only for [Networked] properties inline initialization.
---	---

6.163 NetworkBehaviourUtils.MetaData Struct Reference

This structure holds metadata for a [NetworkBehaviour](#) object.

6.163.1 Detailed Description

This structure holds metadata for a [NetworkBehaviour](#) object.

6.164 NetworkBehaviourWeavedAttribute Class Reference

Network [Behaviour](#) Weaved Attribute

Inherits Attribute.

Public Member Functions

- [NetworkBehaviourWeavedAttribute](#) (int wordCount)
NetworkBehaviourWeavedAttribute Constructor

Properties

- int [WordCount](#) [get]
NetworkBehaviour Word Count

6.164.1 Detailed Description

Network [Behaviour](#) Weaved Attribute

6.164.2 Constructor & Destructor Documentation

6.164.2.1 NetworkBehaviourWeavedAttribute()

```
NetworkBehaviourWeavedAttribute (
    int wordCount )
```

[NetworkBehaviourWeavedAttribute](#) Constructor

Parameters

<code>wordCount</code>	<code>WordCount</code>
------------------------	------------------------

6.164.3 Property Documentation

6.164.3.1 WordCount

`int WordCount [get]`

`NetworkBehaviour` Word Count

6.165 NetworkBool Struct Reference

Represents a boolean value that can be networked.

Inherits [INetworkStruct](#), and [IEquatable< NetworkBool >](#).

Public Member Functions

- `bool Equals (NetworkBool other)`
Determines whether the specified `NetworkBool` is equal to the current `NetworkBool`.
- `override bool Equals (object obj)`
Determines whether the specified object is equal to the current `NetworkBool`.
- `override int GetHashCode ()`
Serves as the default hash function.
- `NetworkBool (bool value)`
Initializes a new instance of the `NetworkBool` struct with the specified value.
- `override string ToString ()`
Returns a string that represents the current `NetworkBool`.

Static Public Member Functions

- `static implicit operator bool (NetworkBool val)`
Defines an implicit conversion of a `NetworkBool` to a `bool`.
- `static implicit operator NetworkBool (bool val)`
Defines an implicit conversion of a `bool` to a `NetworkBool`.

Public Attributes

- `int _value`

6.165.1 Detailed Description

Represents a boolean value that can be networked.

6.165.2 Constructor & Destructor Documentation

6.165.2.1 NetworkBool()

```
NetworkBool (
    bool value )
```

Initializes a new instance of the [NetworkBool](#) struct with the specified value.

Parameters

value	The boolean value.
-------	--------------------

6.165.3 Member Function Documentation

6.165.3.1 Equals() [1/2]

```
bool Equals (
    NetworkBool other )
```

Determines whether the specified [NetworkBool](#) is equal to the current [NetworkBool](#).

Parameters

other	The NetworkBool to compare with the current NetworkBool .
-------	---

Returns

true if the specified [NetworkBool](#) is equal to the current [NetworkBool](#); otherwise, false.

6.165.3.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Determines whether the specified object is equal to the current [NetworkBool](#).

Parameters

<i>obj</i>	The object to compare with the current NetworkBool .
------------	--

Returns

true if the specified object is equal to the current [NetworkBool](#); otherwise, false.

6.165.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current [NetworkBool](#).

6.165.3.4 operator bool()

```
static implicit operator bool (
    NetworkBool val ) [static]
```

Defines an implicit conversion of a [NetworkBool](#) to a bool.

Parameters

<i>val</i>	The NetworkBool to convert.
------------	---

6.165.3.5 operator NetworkBool()

```
static implicit operator NetworkBool (
    bool val ) [static]
```

Defines an implicit conversion of a bool to a [NetworkBool](#).

Parameters

<i>val</i>	The bool to convert.
------------	----------------------

6.165.3.6 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [NetworkBool](#).

Returns

A string that represents the current [NetworkBool](#).

6.166 NetworkButtons Struct Reference

Represents a set of buttons that can be networked.

Inherits [INetworkStruct](#), and [IEquatable< NetworkButtons >](#).

Public Member Functions

- `bool Equals (NetworkButtons other)`
Determines whether the specified [NetworkButtons](#) is equal to the current [NetworkButtons](#).
- `override bool Equals (object obj)`
Determines whether the specified object is equal to the current [NetworkButtons](#).
- `override int GetHashCode ()`
Serves as the default hash function.
- `NetworkButtons GetPressed (NetworkButtons previous)`
Gets the buttons that were pressed since the previous state.
- `NetworkButtons GetPressedOrReleased (NetworkButtons previous)`
- `NetworkButtons GetReleased (NetworkButtons previous)`
Gets the buttons that were released since the previous state.
- `bool IsSet (int button)`
Checks if the button at the specified index is set.
- `bool IsSet< T > (T button)`
Checks if the button of the specified enum type is set.
- `NetworkButtons (int buttons)`
Initializes a new instance of the [NetworkButtons](#) struct with the specified buttons state.
- `void Set (int button, bool state)`
Sets the button at the specified index to the specified state.
- `void Set< T > (T button, bool state)`
Sets the button of the specified enum type to the specified state.
- `void SetAllDown ()`
Sets all buttons to down.
- `void SetAllUp ()`
Sets all buttons to up.
- `void SetDown (int button)`
Sets the button at the specified index to down.
- `void SetDown< T > (T button)`
Sets the button of the specified enum type to down.
- `void SetUp (int button)`
Sets the button at the specified index to up.

- void `SetUp< T >` (`T button`)

Sets the button of the specified enum type to up.
- bool `WasPressed` (`NetworkButtons previous, int button`)

Checks if the button at the specified index was pressed since the previous state.
- bool `WasPressed< T >` (`NetworkButtons previous, T button`)

Checks if the button of the specified enum type was pressed since the previous state.
- bool `WasReleased` (`NetworkButtons previous, int button`)

Checks if the button at the specified index was released since the previous state.
- bool `WasReleased< T >` (`NetworkButtons previous, T button`)

Checks if the button of the specified enum type was released since the previous state.

Public Attributes

- int `_bits`
- `NetworkButtons`

Gets the buttons that were pressed or released since the previous state.

Properties

- int `Bits` [get]

Gets the bits representing the state of the buttons.

6.166.1 Detailed Description

Represents a set of buttons that can be networked.

6.166.2 Constructor & Destructor Documentation

6.166.2.1 `NetworkButtons()`

```
NetworkButtons (
    int buttons )
```

Initializes a new instance of the `NetworkButtons` struct with the specified buttons state.

Parameters

<code>buttons</code>	The integer representing the state of the buttons.
----------------------	--

6.166.3 Member Function Documentation

6.166.3.1 Equals() [1/2]

```
bool Equals (
    NetworkButtons other )
```

Determines whether the specified [NetworkButtons](#) is equal to the current [NetworkButtons](#).

Parameters

<i>other</i>	The NetworkButtons to compare with the current NetworkButtons .
--------------	---

Returns

true if the specified [NetworkButtons](#) is equal to the current [NetworkButtons](#); otherwise, false.

6.166.3.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Determines whether the specified object is equal to the current [NetworkButtons](#).

Parameters

<i>obj</i>	The object to compare with the current NetworkButtons .
------------	---

Returns

true if the specified object is equal to the current [NetworkButtons](#); otherwise, false.

6.166.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current [NetworkButtons](#).

6.166.3.4 GetPressed()

```
NetworkButtons GetPressed (
    NetworkButtons previous )
```

Gets the buttons that were pressed since the previous state.

Parameters

<i>previous</i>	The previous state of the buttons.
-----------------	------------------------------------

Returns

The buttons that were pressed.

6.166.3.5 GetReleased()

```
NetworkButtons GetReleased (
    NetworkButtons previous )
```

Gets the buttons that were released since the previous state.

Parameters

<i>previous</i>	The previous state of the buttons.
-----------------	------------------------------------

Returns

The buttons that were released.

6.166.3.6 IsSet()

```
bool IsSet (
    int button )
```

Checks if the button at the specified index is set.

Parameters

<i>button</i>	The index of the button to check.
---------------	-----------------------------------

Returns

true if the button is set; otherwise, false.

6.166.3.7 IsSet< T >()

```
bool IsSet< T > (
    T button )
```

Checks if the button of the specified enum type is set.

Template Parameters

<i>T</i>	The enum type of the button.
----------	------------------------------

Parameters

<i>button</i>	The button of type T to check.
---------------	--------------------------------

Returns

true if the button is set; otherwise, false.

Type Constraints

T : *unmanaged*

T : *Enum*

6.166.3.8 Set()

```
void Set (
    int button,
    bool state )
```

Sets the button at the specified index to the specified state.

Parameters

<i>button</i>	The index of the button to set.
<i>state</i>	The state to set the button to.

6.166.3.9 Set< T >()

```
void Set< T > (
    T button,
    bool state )
```

Sets the button of the specified enum type to the specified state.

Template Parameters

<i>T</i>	The enum type of the button.
----------	------------------------------

Parameters

<i>button</i>	The button of type T to set.
<i>state</i>	The state to set the button to.

Type Constraints

T : *unmanaged*

T : *Enum*

6.166.3.10 SetAllDown()

```
void SetAllDown ( )
```

Sets all buttons to down.

6.166.3.11 SetAllUp()

```
void SetAllUp ( )
```

Sets all buttons to up.

6.166.3.12 SetDown()

```
void SetDown ( int button )
```

Sets the button at the specified index to down.

Parameters

<i>button</i>	The index of the button to set to down.
---------------	---

6.166.3.13 SetDown< T >()

```
void SetDown< T > ( T button )
```

Sets the button of the specified enum type to down.

Template Parameters

<i>T</i>	The enum type of the button.
----------	------------------------------

Parameters

<i>button</i>	The button of type T to set to down.
---------------	--------------------------------------

Type Constraints

T : **unmanaged**

T : **Enum**

6.166.3.14 SetUp()

```
void SetUp (  
            int button )
```

Sets the button at the specified index to up.

Parameters

<i>button</i>	The index of the button to set to up.
---------------	---------------------------------------

6.166.3.15 SetUp< T >()

```
void SetUp< T > (  
                    T button )
```

Sets the button of the specified enum type to up.

Template Parameters

<i>T</i>	The enum type of the button.
----------	------------------------------

Parameters

<i>button</i>	The button of type T to set to up.
---------------	------------------------------------

Type Constraints

T : **unmanaged**

T : *Enum*

6.166.3.16 WasPressed()

```
bool WasPressed (
    NetworkButtons previous,
    int button )
```

Checks if the button at the specified index was pressed since the previous state.

Parameters

<i>previous</i>	The previous state of the buttons.
<i>button</i>	The index of the button to check.

Returns

true if the button was pressed; otherwise, false.

6.166.3.17 WasPressed< T >()

```
bool WasPressed< T > (
    NetworkButtons previous,
    T button )
```

Checks if the button of the specified enum type was pressed since the previous state.

Template Parameters

<i>T</i>	The enum type of the button.
----------	------------------------------

Parameters

<i>previous</i>	The previous state of the buttons.
<i>button</i>	The button of type <i>T</i> to check.

Returns

true if the button was pressed; otherwise, false.

Type Constraints

T : *unmanaged*

T : *Enum*

6.166.3.18 WasReleased()

```
bool WasReleased (
    NetworkButtons previous,
    int button )
```

Checks if the button at the specified index was released since the previous state.

Parameters

<i>previous</i>	The previous state of the buttons.
<i>button</i>	The index of the button to check.

Returns

true if the button was released; otherwise, false.

6.166.3.19 WasReleased< T >()

```
bool WasReleased< T > (
    NetworkButtons previous,
    T button )
```

Checks if the button of the specified enum type was released since the previous state.

Template Parameters

<i>T</i>	The enum type of the button.
----------	------------------------------

Parameters

<i>previous</i>	The previous state of the buttons.
<i>button</i>	The button of type <i>T</i> to check.

Returns

true if the button was released; otherwise, false.

Type Constraints

T : *unmanaged*

T : *Enum*

6.166.4 Member Data Documentation

6.166.4.1 NetworkButtons

`NetworkButtons`

Gets the buttons that were pressed or released since the previous state.

Parameters

<code>previous</code>	The previous state of the buttons.
-----------------------	------------------------------------

Returns

A tuple containing the buttons that were pressed and the buttons that were released.

6.166.5 Property Documentation

6.166.5.1 Bits

`int Bits [get]`

Gets the bits representing the state of the buttons.

6.167 NetworkConfiguration Class Reference

Main network configuration class.

Public Types

- enum class [ReliableDataTransfers](#)
Flag for allowed Reliable Data transfer modes.

Public Member Functions

- [NetworkConfiguration Init \(\)](#)
Initializes and creates a copy of this [NetworkProjectConfig](#).

Public Attributes

- `double ConnectionShutdownTime = 1`
Default delay between connection changes status to Shutdown (disconnected/invalid), and it actually being released (freeing all references to that particular connection).
- `double ConnectionTimeout = 10`
Max allowed time in seconds that the local peer can run without receiving any update from a remote peer. If a client does not receive any update from the server within this period, it will disconnect itself. If a server does not receive any update from a remote client within this period, it will disconnect that particular client.
- `ReliableDataTransfers ReliableDataTransferModes`
Current [ReliableDataTransferModes](#) mode.

Properties

- int **ConnectAttempts** [get]
Max number of connection attempts that a Client will run when trying to connect to a remote Server.
- double **ConnectInterval** [get]
Interval in seconds between each connection attempt from a Client.
- double **ConnectionDefaultRtt** [get]
Default assumed RTT in seconds for new connections (before actual RTT has been determined). The real RTT is calculated over time once the connection is established.
- double **ConnectionPingInterval** [get]
Interval in seconds between PING messages sent to a remote connection, in order to keep that connection alive.
- int **SocketRecvBufferSize** [get]
Size in Kilobytes of the underlying socket receive buffer.
- int **SocketSendBufferSize** [get]
Size in Kilobytes of the underlying socket send buffer.

6.167.1 Detailed Description

Main network configuration class.

6.167.2 Member Enumeration Documentation

6.167.2.1 ReliableDataTransfers

enum **ReliableDataTransfers** [strong]

Flag for allowed Reliable Data transfer modes.

Enumerator

ClientToServer	Allow Client to Server.
ClientToClientWithServerProxy	Allow Client to Client using Server as Proxy.

6.167.3 Member Function Documentation

6.167.3.1 Init()

NetworkConfiguration **Init** ()

Initializes and creates a copy of this **NetworkProjectConfig**.

Returns

A copy of this **NetworkProjectConfig**.

6.167.4 Member Data Documentation

6.167.4.1 ConnectionShutdownTime

```
double ConnectionShutdownTime = 1
```

Default delay between connection changes status to Shutdown (disconnected/invalid), and it actually being released (freeing all references to that particular connection).

6.167.4.2 ConnectionTimeout

```
double ConnectionTimeout = 10
```

Max allowed time in seconds that the local peer can run without receiving any update from a remote peer. If a client does not receive any update from the server within this period, it will disconnect itself. If a server does not receive any update from a remote client within this period, it will disconnect that particular client.

6.167.4.3 ReliableDataTransferModes

[ReliableDataTransfers](#) ReliableDataTransferModes

Initial value:

```
=  
    ReliableDataTransfers.ClientToServer |  
    ReliableDataTransfers.ClientToClientWithServerProxy
```

Current [ReliableDataTransferModes](#) mode.

6.167.5 Property Documentation

6.167.5.1 ConnectAttempts

```
int ConnectAttempts [get]
```

Max number of connection attempts that a Client will run when trying to connect to a remote Server.

6.167.5.2 ConnectInterval

```
double ConnectInterval [get]
```

Interval in seconds between each connection attempt from a Client.

6.167.5.3 ConnectionDefaultRtt

```
double ConnectionDefaultRtt [get]
```

Default assumed RTT in seconds for new connections (before actual RTT has been determined). The real RTT is calculated over time once the connection is established.

6.167.5.4 ConnectionPingInterval

```
double ConnectionPingInterval [get]
```

Interval in seconds between PING messages sent to a remote connection, in order to keep that connection alive.

Currently unused.

6.167.5.5 SocketRecvBufferSize

```
int SocketRecvBufferSize [get]
```

Size in Kilobytes of the underlying socket receive buffer.

6.167.5.6 SocketSendBufferSize

```
int SocketSendBufferSize [get]
```

Size in Kilobytes of the underlying socket send buffer.

6.168 NetworkDelegates Class Reference

Network Runner Callbacks Delegates

Inherits [INetworkRunnerCallbacks](#).

Public Attributes

- Action< [NetworkRunner](#) > [OnConnectedToServer](#)
- Action< [NetworkRunner](#), [NetAddress](#), [NetConnectFailedReason](#) > [OnConnectFailed](#)
- Action< [NetworkRunner](#), [NetworkRunnerCallbackArgs.ConnectRequest](#), byte[] > [OnConnectRequest](#)
- Action< [NetworkRunner](#), Dictionary< string, object > > [OnCustomAuthenticationResponse](#)
- Action< [NetworkRunner](#), [NetDisconnectReason](#) > [OnDisconnectedFromServer](#)
- Action< [NetworkRunner](#), [HostMigrationToken](#) > [OnHostMigration](#)
- Action< [NetworkRunner](#), [NetworkInput](#) > [OnInput](#)
- Action< [NetworkRunner](#), [PlayerRef](#), [NetworkInput](#) > [OnInputMissing](#)
- Action< [NetworkRunner](#), [NetworkObject](#), [PlayerRef](#) > [OnObjectEnterAOI](#)
- Action< [NetworkRunner](#), [NetworkObject](#), [PlayerRef](#) > [OnObjectExitAOI](#)
- Action< [NetworkRunner](#), [PlayerRef](#) > [OnPlayerJoined](#)
- Action< [NetworkRunner](#), [PlayerRef](#) > [OnPlayerLeft](#)
- Action< [NetworkRunner](#), [PlayerRef](#), ReliableKey, float > [OnReliableDataProgress](#)
- Action< [NetworkRunner](#), [PlayerRef](#), ReliableKey, ArraySegment< byte > > [OnReliableDataReceived](#)
- Action< [NetworkRunner](#) > [OnSceneLoadDone](#)
- Action< [NetworkRunner](#) > [OnSceneLoadStart](#)
- Action< [NetworkRunner](#), List< [SessionInfo](#) > > [OnSessionListUpdated](#)
- Action< [NetworkRunner](#), [ShutdownReason](#) > [OnShutdown](#)
- Action< [NetworkRunner](#), [SimulationMessagePtr](#) > [OnUserSimulationMessage](#)

Additional Inherited Members

6.168.1 Detailed Description

Network Runner Callbacks Delegates

6.169 NetworkDeserializeMethodAttribute Class Reference

Network Deserialize Method Attribute

Inherits Attribute.

6.169.1 Detailed Description

Network Deserialize Method Attribute

6.170 NetworkDictionary< K, V > Struct Template Reference

[Fusion](#) type for networking Dictionaries. Maximum capacity is fixed, and is set with the [CapacityAttribute](#).

Inherits [IEnumerable< KeyValuePair< K, V > >](#), and [INetworkDictionary](#).

Classes

- struct [Enumerator](#)
Enumerator for NetworkDictionary.

Public Member Functions

- bool [Add](#) (K key, V value)
Adds a new key value pair to the Dictionary. If the key already exists, will return false.
- void [INetworkDictionary](#). [Add](#) (object item)
Adds an item to the networked dictionary.
- void [Clear](#) ()
Remove all entries from the Dictionary, and clear backing memory.
- void [ClrEntry](#) (int entry)
- bool [ContainsKey](#) (K key)
Returns true if the Dictionary contains an entry for the given key.
- bool [ContainsValue](#) (V value, IEqualityComparer< V > equalityComparer=null)
Returns true if the Dictionary contains an entry value which compares as equal to given value.
- int [Find](#) (K key)
- V [Get](#) (K key)
Returns the value for the given key. Will throw an error if the key is not found.
- uint [GetBucketFromHashCode](#) (int hash)
- [Enumerator](#) [GetEnumerator](#) ()
Returns an enumerator that iterates through the NetworkDictionary.
- [IEnumerator](#)< KeyValuePair< K, V > > [IEnumerable](#)< KeyValuePair< K, V > >. [GetEnumerator](#) ()
- [IEnumerator](#) [IEnumerable](#). [GetEnumerator](#) ()
- K [GetKey](#) (int entry)
- int [GetKeyHashCode](#) (K key)
- int [GetNxt](#) (int entry)
- V [GetVal](#) (int entry)
- int [Insert](#) (K key, V val)
- [NetworkDictionary](#) (int *data, int capacity, [IElementReaderWriter](#)< K > keyReaderWriter, [IElementReaderWriter](#)< V > valReaderWriter)
Initializes a new instance of the NetworkDictionary struct with the specified data, capacity, and reader/writers.
- bool [Remove](#) (K key)
Remove entry from Dictionary.
- bool [Remove](#) (K key, out V value)
Removes entry from Dictionary. If successful (key existed), returns true and the value of removed item.
- V [Set](#) (K key, V value)
Sets the value for the given key. Will add a new key if the key does not already exist.
- void [SetKey](#) (int entry, K key)
- void [SetNxt](#) (int entry, int next)
- void [SetVal](#) (int entry, V val)
- [NetworkDictionaryReadOnly](#)< K, V > [ToReadOnly](#) ()
Converts the current NetworkDictionary to a read-only version.
- bool [TryGet](#) (K key, out V value)
Attempts to get the value for a given key. If found, returns true.

Static Public Member Functions

- static implicit [operator NetworkDictionaryReadOnly](#)< K, V > ([NetworkDictionary](#)< K, V > value)
Converts the current NetworkDictionary to a read-only version.

Public Attributes

- int **_bucketsOffset**
- int **_capacity**
- int * **_data**
- int **_entriesOffset**
- int **_entryStride**
- EqualityComparer< K > **_equalityComparer**
- int **_keyOffset**
- IElementReaderWriter< K > **_keyReaderWriter**
- int **_nxtOffset**
- int **_valOffset**
- IElementReaderWriter< V > **_valReaderWriter**

Static Public Attributes

- const int **FREE_COUNT_OFFSET** = 1
- const int **FREE_OFFSET** = 0
- const int **INVALID_ENTRY** = 0
- const int **META_WORD_COUNT** = 3
Meta word count for NetworkDictionary.
- const int **USED_COUNT_OFFSET** = 2

Properties

- int **_free** [get, set]
- int **_freeCount** [get, set]
- int **_usedCount** [get, set]
- int **Capacity** [get]
The maximum number of entries this dictionary may contain.
- int **Count** [get]
Current number of key/value entries in the Dictionary.
- V **this[K key]** [get, set]
Key indexer. Gets/Sets value for specified key.

6.170.1 Detailed Description

Fusion type for networking Dictionaries. Maximum capacity is fixed, and is set with the [CapacityAttribute](#).

Typical Usage: [Networked, Capacity(10)]
NetworkDictionary<int, float> syncedDict => default;

Usage for modifying data: var dict = syncedDict; dict.Add(5, 123); dict[5] = 456;
dict.Remove(5);

Template Parameters

K	Key can be a primitive, or an INetworkStruct .
V	Value can be a primitive, or an INetworkStruct .

6.170.2 Constructor & Destructor Documentation

6.170.2.1 NetworkDictionary()

```
NetworkDictionary (
    int * data,
    int capacity,
    IElementReaderWriter< K > keyReaderWriter,
    IElementReaderWriter< V > valReaderWriter )
```

Initializes a new instance of the [NetworkDictionary](#) struct with the specified data, capacity, and reader/writers.

Parameters

<i>data</i>	The pointer to the data of the dictionary.
<i>capacity</i>	The capacity of the dictionary.
<i>keyReaderWriter</i>	The reader/writer for the keys of the dictionary.
<i>valReaderWriter</i>	The reader/writer for the values of the dictionary.

6.170.3 Member Function Documentation

6.170.3.1 Add() [1/2]

```
bool Add (
    K key,
    V value )
```

Adds a new key value pair to the Dictionary. If the key already exists, will return false.

6.170.3.2 Add() [2/2]

```
void INetworkDictionary. Add (
    object item )
```

Adds an item to the networked dictionary.

Parameters

<i>item</i>	The item to add to the dictionary.
-------------	------------------------------------

Implements [INetworkDictionary](#).

6.170.3.3 Clear()

```
void Clear ( )
```

Remove all entries from the Dictionary, and clear backing memory.

6.170.3.4 ContainsKey()

```
bool ContainsKey (  
    K key )
```

Returns true if the Dictionary contains an entry for the given key.

6.170.3.5 ContainsValue()

```
bool ContainsValue (   
    V value,  
    IEqualityComparer< V > equalityComparer = null )
```

Returns true if the Dictionary contains an entry value which compares as equal to given value.

Parameters

<code>value</code>	The value to compare against.
<code>equalityComparer</code>	Specify custom <code>IEqualityComparer</code> to be used for compare.

6.170.3.6 Get()

```
V Get (   
    K key )
```

Returns the value for the given key. Will throw an error if the key is not found.

6.170.3.7 GetEnumerator()

```
Enumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the [NetworkDictionary](#).

6.170.3.8 operator NetworkDictionaryReadOnly< K, V >()

```
static implicit operator NetworkDictionaryReadOnly< K, V > (
    NetworkDictionary< K, V > value ) [static]
```

Converts the current [NetworkDictionary](#) to a read-only version.

Parameters

<i>value</i>	The NetworkDictionary to convert.
--------------	---

Returns

A new instance of [NetworkDictionaryReadOnly](#) with the same data, capacity, and reader/writers as the current [NetworkDictionary](#).

6.170.3.9 Remove() [1/2]

```
bool Remove (
    K key )
```

Remove entry from Dictionary.

Parameters

<i>key</i>	The key to remove.
------------	--------------------

Returns

Returns true if key was found.

6.170.3.10 Remove() [2/2]

```
bool Remove (
    K key,
    out V value )
```

Removes entry from Dictionary. If successful (key existed), returns true and the value of removed item.

Parameters

<i>key</i>	The key to remove.
<i>value</i>	Returns value of removed item. Returns default value if key did not exist.

Returns

Returns true if key was found.

6.170.3.11 Set()

```
V Set (
    K key,
    V value )
```

Sets the value for the given key. Will add a new key if the key does not already exist.

6.170.3.12 ToReadOnly()

```
NetworkDictionaryReadOnly<K, V> ToReadOnly ( )
```

Converts the current [NetworkDictionary](#) to a read-only version.

Returns

A new instance of [NetworkDictionaryReadOnly](#) with the same data, capacity, and reader/writers as the current [NetworkDictionary](#).

6.170.3.13 TryGet()

```
bool TryGet (
    K key,
    out V value )
```

Attempts to get the value for a given key. If found, returns true.

Parameters

<i>key</i>	The key to remove.
<i>value</i>	Returns value of removed item. Returns default value if key did not exist.

Returns

Returns true if key was found.

6.170.4 Member Data Documentation

6.170.4.1 META_WORD_COUNT

```
const int META_WORD_COUNT = 3 [static]
```

Meta word count for [NetworkDictionary](#).

6.170.5 Property Documentation

6.170.5.1 Capacity

```
int Capacity [get]
```

The maximum number of entries this dictionary may contain.

6.170.5.2 Count

```
int Count [get]
```

Current number of key/value entries in the Dictionary.

6.170.5.3 this[K key]

```
V this[K key] [get], [set]
```

Key indexer. Gets/Sets value for specified key.

6.171 NetworkDictionary< K, V >.Enumerator Struct Reference

Enumerator for [NetworkDictionary](#).

Inherits [IEnumerator< KeyValuePair< K, V > >](#).

Public Member Functions

- void [Dispose \(\)](#)
Dispose enumerator.
- bool [MoveNext \(\)](#)
Move to next entry in dictionary.
- void [Reset \(\)](#)
Reset enumerator.

Public Attributes

- int **_bucket**
- NetworkDictionary< K, V > **_dict**
- int **_entry**

Properties

- KeyValuePair< K, V > **Current** [get]
Current key/value pair.
- object IEnumator. **Current** [get]

6.171.1 Detailed Description

Enumerator for NetworkDictionary.

6.171.2 Member Function Documentation

6.171.2.1 Dispose()

```
void Dispose ( )
```

Dispose enumerator.

6.171.2.2 MoveNext()

```
bool MoveNext ( )
```

Move to next entry in dictionary.

Returns

Returns true if there is a next entry.

6.171.2.3 Reset()

```
void Reset ( )
```

Reset enumerator.

6.171.3 Property Documentation

6.171.3.1 Current

```
KeyValuePair<K, V> Current [get]
```

Current key/value pair.

Exceptions

<i>InvalidOperationException</i>	Thrown if enumerator is not valid.
----------------------------------	------------------------------------

6.172 NetworkDictionaryReadOnly< K, V > Struct Template Reference

A read-only version of NetworkDictionary< TKey, TValue >.

Public Member Functions

- int **Find** (K key)
- V **Get** (K key)

Returns the value for the given key. Will throw an error if the key is not found.
- uint **GetBucketFromHashCode** (int hash)
- K **GetKey** (int entry)
- int **GetNxt** (int entry)
- V **GetVal** (int entry)
- bool **TryGet** (K key, out V value)

Attempts to get the value for a given key. If found, returns true.

Public Attributes

- readonly int **_bucketsOffset**
- readonly int **_capacity**
- readonly int * **_data**
- readonly int **_entriesOffset**
- readonly int **_entryStride**
- readonly EqualityComparer< K > **_equalityComparer**
- readonly int **_keyOffset**
- readonly IElementReaderWriter< K > **_keyReaderWriter**
- readonly int **_nxtOffset**
- readonly int **_valOffset**
- readonly IElementReaderWriter< V > **_valReaderWriter**

Static Public Attributes

- const int **FREE_COUNT_OFFSET** = 1
- const int **FREE_OFFSET** = 0
- const int **INVALID_ENTRY** = 0
- const int **USED_COUNT_OFFSET** = 2

Properties

- int **_free** [get]
- int **_freeCount** [get]
- int **_usedCount** [get]
- int **Capacity** [get]

The maximum number of entries this dictionary may contain.
- int **Count** [get]

Current number of key/value entries in the Dictionary.

6.172.1 Detailed Description

A read-only version of NetworkDictionary<TKey,TValue>.

Template Parameters

<i>K</i>	The type of the key.
<i>V</i>	The type of the value.

6.172.2 Member Function Documentation

6.172.2.1 Get()

```
V Get (
    K key )
```

Returns the value for the given key. Will throw an error if the key is not found.

6.172.2.2 TryGet()

```
bool TryGet (
    K key,
    out V value )
```

Attempts to get the value for a given key. If found, returns true.

Parameters

<i>key</i>	The key to remove.
<i>value</i>	Returns value of removed item. Returns default value if key did not exist.

Returns

Returns true if key was found.

6.172.3 Property Documentation

6.172.3.1 Capacity

```
int Capacity [get]
```

The maximum number of entries this dictionary may contain.

6.172.3.2 Count

```
int Count [get]
```

Current number of key/value entries in the Dictionary.

6.173 NetworkedAttribute Class Reference

Inherits Attribute.

Public Member Functions

- [NetworkedAttribute \(\)](#)
Default constructor for NetworkedAttribute

Properties

- string [Default \[get, set\]](#)
Name of the field that holds the default value for this networked property.

6.173.1 Detailed Description

Flags a property of [NetworkBehaviour](#) for network state synchronization. The property should have empty get and set defines, which will automatically be replaced with networking code via IL Weaving.

Inside of [INetworkStruct](#), do not use AutoProperties (get; set;), as these will introduce managed types into the struct, which are not allowed. Instead use '`=> default`'. | [\[Networked\]](#)
| public string StringProp { get => default; set { } }

6.173.2 Constructor & Destructor Documentation

6.173.2.1 NetworkedAttribute()

```
NetworkedAttribute ()
```

Default constructor for NetworkedAttribute

6.173.3 Property Documentation

6.173.3.1 Default

```
string Default [get], [set]
```

Name of the field that holds the default value for this networked property.

6.174 NetworkedWeavedAttribute Class Reference

Networked Weaved Attribute

Inherits Attribute.

Public Member Functions

- [NetworkedWeavedAttribute](#) (int wordOffset, int wordCount)
NetworkedWeavedAttribute Constructor

Properties

- int [WordCount](#) [get]
Networked Property Word Count
- int [WordOffset](#) [get]
Networked Property Word Offset

6.174.1 Detailed Description

Networked Weaved Attribute

Networked Property Attribute

6.174.2 Constructor & Destructor Documentation

6.174.2.1 NetworkedWeavedAttribute()

```
NetworkedWeavedAttribute (
    int wordOffset,
    int wordCount )
```

[NetworkedWeavedAttribute](#) Constructor

Parameters

wordOffset	WordOffset
wordCount	WordCount

6.174.3 Property Documentation

6.174.3.1 WordCount

```
int WordCount [get]
```

Networked Property Word Count

6.174.3.2 WordOffset

```
int WordOffset [get]
```

Networked Property Word Offset

6.175 NetworkEvents Class Reference

Companion component for [NetworkRunner](#). Exposes [INetworkRunnerCallbacks](#) as UnityEvents, which can be wired up to other components in the inspector.

Inherits [Behaviour](#), and [INetworkRunnerCallbacks](#).

Classes

- class [ConnectFailedEvent](#)
UnityEvent for ConnectFailed
- class [ConnectRequestEvent](#)
UnityEvent for ConnectRequest
- class [CustomAuthenticationResponse](#)
UnityEvent for Custom Authentication
- class [DisconnectFromServerEvent](#)
UnityEvent for DisconnectFromServer
- class [HostMigrationEvent](#)
UnityEvent for HostMigration
- class [InputEvent](#)
UnityEvent for NetworkInput
- class [InputPlayerEvent](#)
UnityEvent for NetworkInput with PlayerRef
- class [ObjectEvent](#)
UnityEvent for NetworkObject
- class [ObjectPlayerEvent](#)
UnityEvent for NetworkObject with PlayerRef
- class [PlayerEvent](#)
UnityEvent for PlayerRef

- class [ReliableDataEvent](#)
UnityEvent for Reliable Data
- class [ReliableProgressEvent](#)
UnityEvent for Reliable Data Progress
- class [RunnerEvent](#)
UnityEvent for NetworkRunner
- class [SessionListUpdateEvent](#)
UnityEvent for SessionInfo List
- class [ShutdownEvent](#)
UnityEvent for Shutdown
- class [SimulationMessageEvent](#)
UnityEvent for SimulationMessage

Public Attributes

- [RunnerEvent OnConnectedToServer](#)
- [ConnectFailedEvent OnConnectFailed](#)
- [ConnectRequestEvent OnConnectRequest](#)
- [CustomAuthenticationResponse OnCustomAuthenticationResponse](#)
- [DisconnectFromServerEvent OnDisconnectedFromServer](#)
- [HostMigrationEvent OnHostMigration](#)
- [InputEvent OnInput](#)
- [InputPlayerEvent OnInputMissing](#)
- [ObjectPlayerEvent OnObjectEnterAOI](#)
- [ObjectPlayerEvent OnObjectExitAOI](#)
- [ReliableDataEvent OnReliableData](#)
- [ReliableProgressEvent OnReliableProgress](#)
- [RunnerEvent OnSceneLoadDone](#)
- [RunnerEvent OnSceneLoadStart](#)
- [SessionListUpdateEvent OnSessionListUpdate](#)
- [ShutdownEvent OnShutdown](#)
- [SimulationMessageEvent OnSimulationMessage](#)
- [PlayerEvent PlayerJoined](#)
- [PlayerEvent PlayerLeft](#)

Additional Inherited Members

6.175.1 Detailed Description

Companion component for [NetworkRunner](#). Exposes [INetworkRunnerCallbacks](#) as UnityEvents, which can be wired up to other components in the inspector.

6.176 NetworkEvents.ConnectFailedEvent Class Reference

UnityEvent for ConnectFailed

Inherits UnityEvent< NetworkRunner, NetAddress, NetConnectFailedReason >.

6.176.1 Detailed Description

UnityEvent for ConnectFailed

6.177 NetworkEvents.ConnectRequestEvent Class Reference

UnityEvent for ConnectRequest

Inherits UnityEvent< NetworkRunner, NetworkRunnerCallbackArgs.ConnectRequest, byte[]>.

6.177.1 Detailed Description

UnityEvent for ConnectRequest

6.178 NetworkEvents.CustomAuthenticationResponse Class Reference

UnityEvent for Custom Authentication

Inherits UnityEvent< NetworkRunner, Dictionary< string, object >>.

6.178.1 Detailed Description

UnityEvent for Custom Authentication

6.179 NetworkEvents.DisconnectFromServerEvent Class Reference

UnityEvent for DisconnectFromServer

Inherits UnityEvent< NetworkRunner, NetDisconnectReason >.

6.179.1 Detailed Description

UnityEvent for DisconnectFromServer

6.180 NetworkEvents.HostMigrationEvent Class Reference

UnityEvent for HostMigration

Inherits UnityEvent< NetworkRunner, HostMigrationToken >.

6.180.1 Detailed Description

UnityEvent for HostMigration

6.181 NetworkEvents.InputEvent Class Reference

UnityEvent for [NetworkInput](#)

Inherits UnityEvent< NetworkRunner, NetworkInput >.

6.181.1 Detailed Description

UnityEvent for [NetworkInput](#)

6.182 NetworkEvents.InputPlayerEvent Class Reference

UnityEvent for [NetworkInput](#) with [PlayerRef](#)

Inherits UnityEvent< NetworkRunner, PlayerRef, NetworkInput >.

6.182.1 Detailed Description

UnityEvent for [NetworkInput](#) with [PlayerRef](#)

6.183 NetworkEvents.ObjectEvent Class Reference

UnityEvent for [NetworkObject](#)

Inherits UnityEvent< NetworkRunner, NetworkObject >.

6.183.1 Detailed Description

UnityEvent for [NetworkObject](#)

6.184 NetworkEvents.ObjectPlayerEvent Class Reference

UnityEvent for [NetworkObject](#) with [PlayerRef](#)

Inherits UnityEvent< NetworkRunner, NetworkObject, PlayerRef >.

6.184.1 Detailed Description

UnityEvent for [NetworkObject](#) with [PlayerRef](#)

6.185 NetworkEvents.PlayerEvent Class Reference

UnityEvent for [PlayerRef](#)

Inherits UnityEvent< NetworkRunner, PlayerRef >.

6.185.1 Detailed Description

UnityEvent for [PlayerRef](#)

6.186 NetworkEvents.ReliableDataEvent Class Reference

UnityEvent for Reliable Data

Inherits UnityEvent< NetworkRunner, PlayerRef, ReliableKey, ArraySegment< byte >>.

6.186.1 Detailed Description

UnityEvent for Reliable Data

6.187 NetworkEvents.ReliableProgressEvent Class Reference

UnityEvent for Reliable Data Progress

Inherits UnityEvent< NetworkRunner, PlayerRef, ReliableKey, float >.

6.187.1 Detailed Description

UnityEvent for Reliable Data Progress

6.188 NetworkEvents.RunnerEvent Class Reference

UnityEvent for [NetworkRunner](#)

Inherits UnityEvent< NetworkRunner >.

6.188.1 Detailed Description

UnityEvent for [NetworkRunner](#)

6.189 NetworkEvents.SessionListUpdateEvent Class Reference

UnityEvent for [SessionInfo](#) List

Inherits UnityEvent< NetworkRunner, List< SessionInfo >>.

6.189.1 Detailed Description

UnityEvent for [SessionInfo](#) List

6.190 NetworkEvents.ShutdownEvent Class Reference

UnityEvent for Shutdown

Inherits UnityEvent< NetworkRunner, ShutdownReason >.

6.190.1 Detailed Description

UnityEvent for Shutdown

6.191 NetworkEvents.SimulationMessageEvent Class Reference

UnityEvent for [SimulationMessage](#)

Inherits UnityEvent< NetworkRunner, SimulationMessagePtr >.

6.191.1 Detailed Description

UnityEvent for [SimulationMessage](#)

6.192 NetworkId Struct Reference

The unique identifier for a network entity.

Inherits [INetworkStruct](#), [IEquatable< NetworkId >](#), [IComparable](#), and [IComparable< NetworkId >](#).

Classes

- class [EqualityComparer](#)
IEqualityComparer interface for NetworkId objects.

Public Member Functions

- int [CompareTo](#) (NetworkId other)
Compares the current NetworkId object with another NetworkId object.
- int [IComparable.CompareTo](#) (object obj)
- bool [Equals](#) (NetworkId other)
Determines whether the current NetworkId object is equal to another NetworkId object.
- override bool [Equals](#) (object obj)
Determines whether the specified object is equal to the current NetworkId object.
- override int [GetHashCode](#) ()
Get the hash code for this NetworkId.
- string [ToNamePrefixString](#) ()
String conversion specifically for use in prefixing names of GameObjects.
- override string [ToString](#) ()
String representation of the NetworkId.
- void [Write](#) (NetBitBuffer *buffer)
Writes this NetworkId to the provided NetBitBuffer.

Static Public Member Functions

- static implicit operator bool (NetworkId id)
Converts the NetworkId object to a boolean value.
- static bool [operator!=](#) (NetworkId a, NetworkId b)
Determines whether two NetworkId objects are not equal.
- static bool [operator==](#) (NetworkId a, NetworkId b)
Determines whether two NetworkId objects are equal.
- static NetworkId [Read](#) (NetBitBuffer *buffer)
Reads a NetworkId from the provided NetBitBuffer.
- static void [Write](#) (NetBitBuffer *buffer, NetworkId id)
Writes the NetworkId to the provided NetBitBuffer.

Public Attributes

- uint [Raw](#)
The raw value of the network id.

Static Public Attributes

- const int [ALIGNMENT](#) = 4
The alignment of the network id in bytes.
- const int [BLOCK_SIZE](#) = 8
The size of the network id block in bytes.
- const uint [RAW_PHYSICS_INFO](#) = 4u
- const uint [RAW_PLAYER_REF_DATA_ARRAY](#) = 2u
- const uint [RAW_RUNTIME_CONFIG](#) = 1u
- const uint [RAW_SCENE_INFO](#) = 3u
- const int [SIZE](#) = 4
The size of the network id in bytes.

Properties

- static EqualityComparer Comparer = new EqualityComparer() [get]
The IEqualityComparer for NetworkId objects.
- bool IsReserved [get]
Signal if the network id is reserved.
- bool IsValid [get]
Signal if the network id is valid.

6.192.1 Detailed Description

The unique identifier for a network entity.

6.192.2 Member Function Documentation

6.192.2.1 CompareTo()

```
int CompareTo (
    NetworkId other )
```

Compares the current NetworkId object with another NetworkId object.

Parameters

other	A NetworkId object to compare with this object.
-------	---

Returns

A value that indicates the relative order of the objects being compared.

6.192.2.2 Equals() [1/2]

```
bool Equals (
    NetworkId other )
```

Determines whether the current NetworkId object is equal to another NetworkId object.

Parameters

other	A NetworkId object to compare with this object.
-------	---

Returns

true if the current object is equal to the other parameter; otherwise, false.

6.192.2.3 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Determines whether the specified object is equal to the current [NetworkId](#) object.

Parameters

<i>obj</i>	The object to compare with the current object.
------------	--

Returns

true if the specified object is equal to the current object; otherwise, false.

6.192.2.4 GetHashCode()

```
override int GetHashCode ( )
```

Get the hash code for this [NetworkId](#).

6.192.2.5 operator bool()

```
static implicit operator bool (
    NetworkId id ) [static]
```

Converts the [NetworkId](#) object to a boolean value.

6.192.2.6 operator"!=()

```
static bool operator!= (
    NetworkId a,
    NetworkId b ) [static]
```

Determines whether two [NetworkId](#) objects are not equal.

6.192.2.7 operator==()

```
static bool operator== (
    NetworkId a,
    NetworkId b ) [static]
```

Determines whether two [NetworkId](#) objects are equal.

6.192.2.8 Read()

```
static NetworkId Read (
    NetBitBuffer * buffer ) [static]
```

Reads a [NetworkId](#) from the provided NetBitBuffer.

Parameters

<i>buffer</i>	The buffer to read the NetworkId from.
---------------	--

Returns

The [NetworkId](#) read from the buffer.

6.192.2.9 ToNamePrefixString()

```
string ToNamePrefixString ( )
```

String conversion specifically for use in prefixing names of GameObjects.

6.192.2.10 ToString()

```
override string ToString ( )
```

String representation of the [NetworkId](#).

6.192.2.11 Write() [1/2]

```
void Write (
    NetBitBuffer * buffer )
```

Writes this [NetworkId](#) to the provided NetBitBuffer.

Parameters

<i>buffer</i>	The buffer to write this NetworkId to.
---------------	--

6.192.2.12 Write() [2/2]

```
static void Write (
    NetBitBuffer * buffer,
    NetworkId id ) [static]
```

Writes the [NetworkId](#) to the provided NetBitBuffer.

Parameters

<i>buffer</i>	The buffer to write the NetworkId to.
<i>id</i>	The NetworkId to write.

6.192.3 Member Data Documentation**6.192.3.1 ALIGNMENT**

```
const int ALIGNMENT = 4 [static]
```

The alignment of the network id in bytes.

6.192.3.2 BLOCK_SIZE

```
const int BLOCK_SIZE = 8 [static]
```

The size of the network id block in bytes.

6.192.3.3 Raw

```
uint Raw
```

The raw value of the network id.

6.192.3.4 SIZE

```
const int SIZE = 4 [static]
```

The size of the network id in bytes.

6.192.4 Property Documentation

6.192.4.1 Comparer

```
IEqualityComparer Comparer = new EqualityComparer() [static], [get]
```

The IEqualityComparer for [NetworkId](#) objects.

6.192.4.2 IsReserved

```
bool IsReserved [get]
```

Signal if the network id is reserved.

6.192.4.3 IsValid

```
bool IsValid [get]
```

Signal if the network id is valid.

6.193 NetworkId.EqualityComparer Class Reference

IEqualityComparer interface for [NetworkId](#) objects.

Inherits IEqualityComparer< NetworkId >.

Public Member Functions

- bool [Equals \(NetworkId a, NetworkId b\)](#)
Determines whether the specified NetworkId objects are equal.
- int [GetHashCode \(NetworkId id\)](#)
Returns a hash code for the specified NetworkId object.

6.193.1 Detailed Description

IEqualityComparer interface for [NetworkId](#) objects.

6.193.2 Member Function Documentation

6.193.2.1 Equals()

```
bool Equals (
    NetworkId a,
    NetworkId b )
```

Determines whether the specified [NetworkId](#) objects are equal.

6.193.2.2 GetHashCode()

```
int GetHashCode (
    NetworkId id )
```

Returns a hash code for the specified [NetworkId](#) object.

6.194 NetworkInput Struct Reference

[NetworkInput](#) Struct

Public Member Functions

- bool [Convert](#) ([Type](#) type)
• bool [Convert< T >](#) ()
Converts the Type of this [INetworkInput](#) to another type
- T [Get< T >](#) ()
Gets the content of this [INetworkInput](#) as another type
- bool [Is< T >](#) ()
Checks if this [INetworkInput](#) is of a certain type
- bool [Set< T >](#) (T value)
Sets the content of this [INetworkInput](#) to another type
- bool [TryGet< T >](#) (out T input)
Tries to export data as the indicated T [INetworkInput](#) struct.
- bool [TrySet< T >](#) (T input)
Tries to import data from a [INetworkInput](#) struct.

Properties

- `uint * Data` [get]
Data pointer of the NetworkInput
- `bool IsValid` [get]
Signal if the NetworkInput is valid or not
- `Type Type` [get]
Get the Type associated with this NetworkInput
- `int WordCount` [get]
Number of Words for the NetworkInput

6.194.1 Detailed Description

[NetworkInput Struct](#)

6.194.2 Member Function Documentation

6.194.2.1 Convert< T >()

```
bool Convert< T > ( )
```

Converts the Type of this [INetworkInput](#) to another type

Type Constraints

T : unmanaged
T : INetworkInput
T : Convert
T : typeof
T : T

6.194.2.2 Get< T >()

```
T Get< T > ( )
```

Gets the content of this [INetworkInput](#) as another type

Type Constraints

T : unmanaged
T : INetworkInput

6.194.2.3 Is< T >()

```
bool Is< T > ( )
```

Checks if this [INetworkInput](#) is of a certain type

Type Constraints

T : **unmanaged**

T : [INetworkInput](#)

6.194.2.4 Set< T >()

```
bool Set< T > (  
    T value )
```

Sets the content of this [INetworkInput](#) to another type

Type Constraints

T : **unmanaged**

T : [INetworkInput](#)

6.194.2.5 TryGet< T >()

```
bool TryGet< T > (  
    out T input )
```

Tries to export data as the indicated *T* [INetworkInput](#) struct.

Type Constraints

T : **unmanaged**

T : [INetworkInput](#)

6.194.2.6 TrySet< T >()

```
bool TrySet< T > (  
    T input )
```

Tries to import data from a [INetworkInput](#) struct.

Type Constraints

T : **unmanaged**

T : [INetworkInput](#)

6.194.3 Property Documentation

6.194.3.1 Data

```
uint* Data [get]
```

Data pointer of the [NetworkInput](#)

6.194.3.2 IsValid

```
bool IsValid [get]
```

Signal if the [NetworkInput](#) is valid or not

6.194.3.3 Type

```
Type Type [get]
```

Get the Type associated with this [NetworkInput](#)

6.194.3.4 WordCount

```
int WordCount [get]
```

Number of Words for the [NetworkInput](#)

6.195 NetworkInputUtils Class Reference

Utility methods for [NetworkInput](#)

Static Public Member Functions

- static int [GetMaxWordCount \(\)](#)
Get the max word count from all registered types
- static Type [GetType \(int typeKey\)](#)
Get the Type based on its associate Key
- static int [GetTypeKey \(Type type\)](#)
Get the Key associate with the argument Type
- static int [GetWordCount \(Type type\)](#)
Get Type Word Count if it is of type [NetworkInput](#)

6.195.1 Detailed Description

Utility methods for [NetworkInput](#)

6.195.2 Member Function Documentation

6.195.2.1 GetMaxWordCount()

```
static int GetMaxWordCount ( ) [static]
```

Get the max word count from all registered types

6.195.2.2 GetType()

```
static Type GetType ( int typeKey ) [static]
```

Get the Type based on its associate Key

Parameters

<i>typeKey</i>	Key associated with a Type
----------------	----------------------------

Returns

Type associated with the Key, or null otherwise

6.195.2.3 GetTypeKey()

```
static int GetTypeKey ( Type type ) [static]
```

Get the Key associate with the argument Type

Parameters

<i>type</i>	Type to check for the key
-------------	---------------------------

Returns

Associated Type Key, or an exception if not found

6.195.2.4 GetWordCount()

```
static int GetWordCount (
    Type type ) [static]
```

Get Type Word Count if it is of type [NetworkInput](#)

Parameters

<code>type</code>	Type to check for word count
-------------------	------------------------------

Returns

Number of words for the [NetworkInput](#)

6.196 NetworkInputWeavedAttribute Class Reference

Network Input Weaved Attribute

Inherits Attribute.

Public Member Functions

- [NetworkInputWeavedAttribute](#) (int wordCount)
NetworkInputWeavedAttribute Constructor

Properties

- int [WordCount](#) [get]
NetworkInput Word Count

6.196.1 Detailed Description

Network Input Weaved Attribute

6.196.2 Constructor & Destructor Documentation

6.196.2.1 NetworkInputWeavedAttribute()

```
NetworkInputWeavedAttribute (
    int wordCount )
```

[NetworkInputWeavedAttribute](#) Constructor

Parameters

<code>wordCount</code>	WordCount
------------------------	---------------------------

6.196.3 Property Documentation

6.196.3.1 WordCount

`int WordCount [get]`

[NetworkInput](#) Word Count

6.197 NetworkLinkedList< T > Struct Template Reference

[Fusion](#) type for networking LinkedLists. Maximum capacity is fixed, and is set with the [CapacityAttribute](#).

Typical Usage:

Inherits [IEnumerable< T >](#), and [INetworkLinkedList](#).

Classes

- struct [Enumerator](#)
Enumerator for NetworkLinkedList< T >.

Public Member Functions

- void [INetworkLinkedList.Add](#) (object item)
Adds an item to the networked linked list.
- void [Add](#) (T value)
Adds a value to the end of the list.
- void [Clear](#) ()
Removes and clears all list elements.
- bool [Contains](#) (T value)
Returns true if the value already exists in the list.
- bool [Contains](#) (T value, [IEqualityComparer< T >](#) comparer)
Returns true if the value already exists in the list.
- int * [Entry](#) (int index)
- int * [FindFreeEntry](#) (out int index)
- T [Get](#) (int index)
Returns the value at supplied index.
- int * [GetEntryByListIndex](#) (int listIndex)
- [IEnumerator](#) [GetEnumerator](#) ()
Get the enumerator for the list.

- `IEnumerator< T > IEnumerable< T >. GetEnumerator ()`
- `IEnumerator IEnumerable. GetEnumerator ()`
- `int IndexOf (T value)`

Returns the index with this value. Returns -1 if not found.
- `int IndexOf (T value, IEqualityComparer< T > equalityComparer)`

Returns the index of the first occurrence of a value in the NetworkLinkedList.
- `NetworkLinkedList (byte *data, int capacity, IElementReaderWriter< T > rw)`

Initializes a new instance of the NetworkLinkedList struct with the specified data, capacity, and reader/writer.
- `T Read (int *entry)`
- `NetworkLinkedList< T > Remap (void *list)`

Remaps the current NetworkLinkedList to a new memory location.
- `bool Remove (T value)`

Removes the first found element with indicated value.
- `bool Remove (T value, IEqualityComparer< T > equalityComparer)`

Removes the first found element with indicated value.
- `void RemoveEntry (int *entry, int entryIndex)`
- `T Set (int index, T value)`

Sets the value at supplied index.
- `void Write (int *entry, T value)`

Public Attributes

- `int _capacity`
- `int * _data`
- `IElementReaderWriter< T > _rw`
- `int _stride`

Static Public Attributes

- `const int COUNT = 0`
- `const int ELEMENT_WORDS = 2`

Returns the number of words required to store a single element.
- `const int HEAD = 1`
- `const int INVALID = 0`
- `const int META_WORDS = 3`

Returns the number of words required to store the list metadata.
- `const int NEXT = 1`
- `const int OFFSET = 1`
- `const int PREV = 0`
- `const int TAIL = 2`

Properties

- `int Capacity [get]`

Returns the max element count.
- `int Count [get]`

Returns the current element count.
- `int Head [get, set]`
- `int Tail [get, set]`
- `T this[int index] [get, set]`

Element indexer.

6.197.1 Detailed Description

Fusion type for networking LinkedLists. Maximum capacity is fixed, and is set with the [CapacityAttribute](#).

Typical Usage:

```
[Networked, Capacity(10)]
NetworkLinkedList<int> syncedLinkedList => default;

Optional usage (for NetworkBehaviours ONLY - this is not legal in INetworkStructs): [Networked,
Capacity(4)]
NetworkLinkedList<int> syncedLinkedList { get; } = MakeInitializer(new int[]
{ 1, 2, 3, 4 });
```

Usage for modifying data: var list = syncedLinkedList; list.Add(123); list[0] = 456; list.Remove(0);

Template Parameters

<i>T</i>	T can be a primitive, or an INetworkStruct .
----------	--

6.197.2 Constructor & Destructor Documentation

6.197.2.1 NetworkLinkedList()

```
NetworkLinkedList (
    byte * data,
    int capacity,
    IElementReaderWriter< T > rw )
```

Initializes a new instance of the [NetworkLinkedList](#) struct with the specified data, capacity, and reader/writer.

Parameters

<i>data</i>	The pointer to the data of the list.
<i>capacity</i>	The capacity of the list.
<i>rw</i>	The reader/writer for the elements of the list.

6.197.3 Member Function Documentation

6.197.3.1 Add() [1/2]

```
void INetworkLinkedList. Add (
    object item )
```

Adds an item to the networked linked list.

Parameters

<i>item</i>	The item to add to the linked list.
-------------	-------------------------------------

Implements [INetworkLinkedList](#).

6.197.3.2 Add() [2/2]

```
void Add (
    T value )
```

Adds a value to the end of the list.

Parameters

<i>value</i>	Value to add.
--------------	---------------

6.197.3.3 Clear()

```
void Clear ( )
```

Removes and clears all list elements.

6.197.3.4 Contains() [1/2]

```
bool Contains (
    T value )
```

Returns true if the value already exists in the list.

6.197.3.5 Contains() [2/2]

```
bool Contains (
    T value,
    IEqualityComparer< T > comparer )
```

Returns true if the value already exists in the list.

6.197.3.6 Get()

```
T Get (
    int index )
```

Returns the value at supplied index.

6.197.3.7 GetEnumerator()

```
Enumerator GetEnumerator ( )
```

Get the enumerator for the list.

6.197.3.8 IndexOf() [1/2]

```
int IndexOf (
    T value )
```

Returns the index with this value. Returns -1 if not found.

6.197.3.9 IndexOf() [2/2]

```
int IndexOf (
    T value,
    IEqualityComparer< T > equalityComparer )
```

Returns the index of the first occurrence of a value in the [NetworkLinkedList](#).

Parameters

<code>value</code>	The value to locate in the NetworkLinkedList . The value can be null for reference types.
<code>equalityComparer</code>	An equality comparer to compare values. Must not be null.

Returns

The zero-based index of the first occurrence of value within the entire [NetworkLinkedList](#), if found; otherwise, -1.

This method performs a linear search; therefore, this method is an O(n) operation, where n is Capacity.

6.197.3.10 Remap()

```
NetworkLinkedList<T> Remap (
    void * list )
```

Remaps the current [NetworkLinkedList](#) to a new memory location.

Parameters

<i>list</i>	The pointer to the new memory location.
-------------	---

Returns

A new instance of [NetworkLinkedList](#) with the same capacity and reader/writer, but mapped to the new memory location.

6.197.3.11 Remove() [1/2]

```
bool Remove (
    T value )
```

Removes the first found element with indicated value.

6.197.3.12 Remove() [2/2]

```
bool Remove (
    T value,
    IEqualityComparer< T > equalityComparer )
```

Removes the first found element with indicated value.

6.197.3.13 Set()

```
T Set (
    int index,
    T value )
```

Sets the value at supplied index.

6.197.4 Member Data Documentation

6.197.4.1 ELEMENT_WORDS

```
const int ELEMENT_WORDS = 2 [static]
```

Returns the number of words required to store a single element.

6.197.4.2 META_WORDS

```
const int META_WORDS = 3 [static]
```

Returns the number of words required to store the list metadata.

6.197.5 Property Documentation

6.197.5.1 Capacity

```
int Capacity [get]
```

Returns the max element count.

6.197.5.2 Count

```
int Count [get]
```

Returns the current element count.

6.197.5.3 this[int index]

```
T this[int index] [get], [set]
```

Element indexer.

6.198 NetworkLinkedList< T >.Enumerator Struct Reference

Enumerator for [NetworkLinkedList<T>](#).

Inherits [IEnumerator< T >](#).

Public Member Functions

- void [Dispose \(\)](#)
Releases all resources used by the [NetworkLinkedList< T >.Enumerator](#).
- bool [MoveNext \(\)](#)
Advances the enumerator to the next element of the [NetworkLinkedList< T >](#).
- void [Reset \(\)](#)
Resets the enumerator to its initial position, which is before the first element in the collection.

Public Attributes

- bool `_first`
- int `_head`
- `NetworkLinkedList< T > _list`

Properties

- T `Current` [get]
Gets the current element in the collection.
- object `IEnumerator`. `Current` [get]

6.198.1 Detailed Description

Enumerator for `NetworkLinkedList<T>`.

6.198.2 Member Function Documentation

6.198.2.1 `Dispose()`

```
void Dispose ( )
```

Releases all resources used by the `NetworkLinkedList<T>.Enumerator`.

6.198.2.2 `MoveNext()`

```
bool MoveNext ( )
```

Advances the enumerator to the next element of the `NetworkLinkedList<T>`.

Returns

Returns true if the enumerator advanced to the next element.

6.198.2.3 `Reset()`

```
void Reset ( )
```

Resets the enumerator to its initial position, which is before the first element in the collection.

6.198.3 Property Documentation

6.198.3.1 `Current`

```
T Current [get]
```

Gets the current element in the collection.

Returns

The current element in the collection.

Exceptions

<i>InvalidOperationException</i>	Thrown when the enumerator is positioned before the first element or after the last element.
----------------------------------	--

6.199 NetworkLinkedListReadOnly< T > Struct Template Reference

Read-only version of NetworkLinkedList<T>.

Public Member Functions

- bool **Contains** (T value)

Returns true if the value already exists in the list.
- bool **Contains** (T value, IEqualityComparer< T > comparer)

Returns true if the value already exists in the list.
- int * **Entry** (int index)
- T **Get** (int index)

Returns the value at supplied index.
- int * **GetEntryByListIndex** (int listIndex)
- int **IndexOf** (T value)

Returns the index with this value. Returns -1 if not found.
- int **IndexOf** (T value, IEqualityComparer< T > equalityComparer)

Returns the index of the first occurrence of a value in the [FixedArray](#).
- T **Read** (int *entry)

Public Attributes

- int **_capacity**
- int * **_data**
- [IElementReaderWriter](#)< T > **_rw**
- int **_stride**

Static Public Attributes

- const int **COUNT** = 0
- const int **ELEMENT_WORDS** = 2

Returns the number of words required to store a single element.
- const int **HEAD** = 1
- const int **INVALID** = 0
- const int **META_WORDS** = 3

Returns the number of words required to store the list metadata.
- const int **NEXT** = 1
- const int **OFFSET** = 1
- const int **PREV** = 0
- const int **TAIL** = 2

Properties

- int **Capacity** [get]
Returns the max element count.
- int **Count** [get]
Returns the current element count.
- int **Head** [get]
- int **Tail** [get]
- T **this[int index]** [get]
Element indexer.

6.199.1 Detailed Description

Read-only version of NetworkLinkedList<T>.

Template Parameters

<i>T</i>	Custom struct type.
----------	---------------------

6.199.2 Member Function Documentation

6.199.2.1 Contains() [1/2]

```
bool Contains (
    T value )
```

Returns true if the value already exists in the list.

6.199.2.2 Contains() [2/2]

```
bool Contains (
    T value,
    IEqualityComparer< T > comparer )
```

Returns true if the value already exists in the list.

6.199.2.3 Get()

```
T Get (
    int index )
```

Returns the value at supplied index.

6.199.2.4 IndexOf() [1/2]

```
int IndexOf (
    T value )
```

Returns the index with this value. Returns -1 if not found.

6.199.2.5 IndexOf() [2/2]

```
int IndexOf (
    T value,
    IEqualityComparer< T > equalityComparer )
```

Returns the index of the first occurrence of a value in the [FixedArray](#).

Parameters

<code>value</code>	The value to locate in the FixedArray . The value can be null for reference types.
<code>equalityComparer</code>	An equality comparer to compare values. Must not be null.

Returns

The zero-based index of the first occurrence of value within the entire [FixedArray](#), if found; otherwise, -1.

This method performs a linear search; therefore, this method is an O(n) operation, where n is Capacity.

6.199.3 Member Data Documentation

6.199.3.1 ELEMENT_WORDS

```
const int ELEMENT_WORDS = 2 [static]
```

Returns the number of words required to store a single element.

6.199.3.2 META_WORDS

```
const int META_WORDS = 3 [static]
```

Returns the number of words required to store the list metadata.

6.199.4 Property Documentation

6.199.4.1 Capacity

```
int Capacity [get]
```

Returns the max element count.

6.199.4.2 Count

```
int Count [get]
```

Returns the current element count.

6.199.4.3 this[int index]

```
T this[int index] [get]
```

Element indexer.

6.200 NetworkLoadSceneParameters Struct Reference

Parameters for loading a scene

Inherits IEquatable< NetworkLoadSceneParameters >.

Public Member Functions

- bool [Equals \(NetworkLoadSceneParameters other\)](#)
Compares two NetworkLoadSceneParameters for equality
- override bool [Equals \(object obj\)](#)
Compares two NetworkLoadSceneParameters for equality
- override int [GetHashCode \(\)](#)
Returns the hash code of the NetworkLoadSceneParameters
- override string [ToString \(\)](#)
Returns a string representation of the NetworkLoadSceneParameters

Static Public Member Functions

- static bool `operator!=` (`NetworkLoadSceneParameters` left, `NetworkLoadSceneParameters` right)
Compares two `NetworkLoadSceneParameters` for inequality
- static bool `operator==` (`NetworkLoadSceneParameters` left, `NetworkLoadSceneParameters` right)
Compares two `NetworkLoadSceneParameters` for equality

Public Attributes

- readonly `NetworkSceneLoadId LoadId`
The unique id of the scene load operation

Properties

- bool `IsActiveOnLoad` [get]
Signals if the scene should be active on load
- bool `IsLocalPhysics2D` [get]
Signals if the scene should have local 2D physics
- bool `IsLocalPhysics3D` [get]
Signals if the scene should have local 3D physics
- bool `IsSingleLoad` [get]
Signals if the scene should be single loaded
- `LoadSceneMode LoadSceneMode` [get]
The `LoadSceneMode` to use when loading the scene
- `LoadSceneParameters LoadSceneParameters` [get]
The `LoadSceneParameters` to use when loading the scene
- `LocalPhysicsMode LocalPhysicsMode` [get]
The `LocalPhysicsMode` to use when loading the scene

6.200.1 Detailed Description

Parameters for loading a scene

6.200.2 Member Function Documentation

6.200.2.1 Equals() [1/2]

```
bool Equals (
    NetworkLoadSceneParameters other )
```

Compares two `NetworkLoadSceneParameters` for equality

Parameters

<code>other</code>	The other <code>NetworkLoadSceneParameters</code>
--------------------	---

Returns

Returns true if the two [NetworkLoadSceneParameters](#) are equal

6.200.2.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Compares two [NetworkLoadSceneParameters](#) for equality

Parameters

<i>obj</i>	The other NetworkLoadSceneParameters
------------	--

Returns

Returns true if the two [NetworkLoadSceneParameters](#) are equal

6.200.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code of the [NetworkLoadSceneParameters](#)

6.200.2.4 operator"!=()

```
static bool operator!= (
    NetworkLoadSceneParameters left,
    NetworkLoadSceneParameters right ) [static]
```

Compares two [NetworkLoadSceneParameters](#) for inequality

Parameters

<i>left</i>	Left NetworkLoadSceneParameters
<i>right</i>	Right NetworkLoadSceneParameters

Returns

Returns true if the two [NetworkLoadSceneParameters](#) are not equal

6.200.2.5 operator==()

```
static bool operator== (
    NetworkLoadSceneParameters left,
    NetworkLoadSceneParameters right ) [static]
```

Compares two [NetworkLoadSceneParameters](#) for equality

Parameters

<i>left</i>	Left NetworkLoadSceneParameters
<i>right</i>	Right NetworkLoadSceneParameters

Returns

Returns true if the two [NetworkLoadSceneParameters](#) are equal

6.200.2.6 ToString()

```
override string ToString ( )
```

Returns a string representation of the [NetworkLoadSceneParameters](#)

6.200.3 Member Data Documentation

6.200.3.1 LoadId

```
readonly NetworkSceneLoadId LoadId
```

The unique id of the scene load operation

6.200.4 Property Documentation

6.200.4.1 IsActiveOnLoad

```
bool IsActiveOnLoad [get]
```

Signals if the scene should be active on load

6.200.4.2 IsLocalPhysics2D

```
bool IsLocalPhysics2D [get]
```

Signals if the scene should have local 2D physics

6.200.4.3 IsLocalPhysics3D

```
bool IsLocalPhysics3D [get]
```

Signals if the scene should have local 3D physics

6.200.4.4 IsSingleLoad

```
bool IsSingleLoad [get]
```

Signals if the scene should be single loaded

6.200.4.5 LoadSceneMode

```
LoadSceneMode LoadSceneMode [get]
```

The [LoadSceneMode](#) to use when loading the scene

6.200.4.6 LoadSceneParameters

```
LoadSceneParameters LoadSceneParameters [get]
```

The [LoadSceneParameters](#) to use when loading the scene

6.200.4.7 LocalPhysicsMode

```
LocalPhysicsMode LocalPhysicsMode [get]
```

The [LocalPhysicsMode](#) to use when loading the scene

6.201 NetworkMecanimAnimator Class Reference

A component for synchronizing the Animator controller state from the State Authority to network proxies. Requires a Unity Animator component, and a [NetworkObject](#) component. NOTE: Animator Root Motion is not compatible with re-simulation and prediction.

Inherits [NetworkBehaviour](#), and [IAfterAllTicks](#).

Public Member Functions

- override void [Render](#) ()
Post simulation frame rendering callback. Runs after all simulations have finished. Use in place of Unity's Update when Fusion is handling Physics.
- void [SetTrigger](#) (int triggerHash, bool passThroughOnInputAuthority=false)
Queues a [SetTrigger\(\)](#) call for the associated Animator on the State Authority. Call this instead of Animator.SetTrigger() for the State Authority to ensure that triggers are captured. On State Authority, this call will defer the [SetTrigger\(\)](#) pass-through to the Animator until [FixedUpdateNetwork\(\)](#) is called, where all queued triggers will be executed (this is to ensure tick agreement between server and clients).
- void [SetTrigger](#) (string trigger, bool passThroughOnInputAuthority=false)
Queues a [SetTrigger\(\)](#) call for the associated Animator on the State Authority. Call this instead of Animator.SetTrigger() for the State Authority to ensure that triggers are captured. On State Authority, this call will defer the [SetTrigger\(\)](#) pass-through to the Animator until [FixedUpdateNetwork\(\)](#) is called, where all queued triggers will be executed (this is to ensure tick agreement between server and clients).
- override void [Spawned](#) ()
Post spawn callback.

Public Attributes

- Animator [Animator](#)
The Animator being synced. If unset, will attempt to find one on this GameObject.
- [RenderSource](#) [ApplyTiming](#) = [RenderSource.To](#)
The source of the State which is applied in Render.

Properties

- override? int [DynamicWordCount](#) [get]
Gets the dynamic word count for the [NetworkMecanimAnimator](#).

Additional Inherited Members

6.201.1 Detailed Description

A component for synchronizing the Animator controller state from the State Authority to network proxies. Requires a Unity Animator component, and a [NetworkObject](#) component. NOTE: Animator Root Motion is not compatible with re-simulation and prediction.

6.201.2 Member Function Documentation

6.201.2.1 SetTrigger() [1/2]

```
void SetTrigger (
    int triggerHash,
    bool passThroughOnInputAuthority = false )
```

Queues a [SetTrigger\(\)](#) call for the associated Animator on the State Authority. Call this instead of `Animator.SetTrigger()` for the State Authority to ensure that triggers are captured. On State Authority, this call will defer the [SetTrigger\(\)](#) pass-through to the Animator until [FixedUpdateNetwork\(\)](#) is called, where all queued triggers will be executed (this is to ensure tick agreement between server and clients).

Parameters

<code>triggerHash</code>	Trigger hash to set
<code>passThroughOnInputAuthority</code>	Will call <code>Animator.SetTrigger()</code> immediately on the InputAuthority. If false, SetTrigger() will not be called on the Input Authority at all and <code>Animator.SetTrigger()</code> should be called explicitly as needed.

6.201.2.2 SetTrigger() [2/2]

```
void SetTrigger (
    string trigger,
    bool passThroughOnInputAuthority = false )
```

Queues a [SetTrigger\(\)](#) call for the associated Animator on the State Authority. Call this instead of `Animator.SetTrigger()` for the State Authority to ensure that triggers are captured. On State Authority, this call will defer the [SetTrigger\(\)](#) pass-through to the Animator until [FixedUpdateNetwork\(\)](#) is called, where all queued triggers will be executed (this is to ensure tick agreement between server and clients).

Parameters

<code>trigger</code>	Trigger name to set
<code>passThroughOnInputAuthority</code>	Will call <code>Animator.SetTrigger()</code> immediately on the InputAuthority. If false, SetTrigger() will not be called on the Input Authority at all and <code>Animator.SetTrigger()</code> should be called explicitly as needed.

6.201.3 Member Data Documentation

6.201.3.1 Animator

`Animator` `Animator`

The Animator being synced. If unset, will attempt to find one on this GameObject.

6.201.3.2 ApplyTiming

```
RenderSource ApplyTiming = RenderSource.To
```

The source of the State which is applied in Render.

6.201.4 Property Documentation

6.201.4.1 DynamicWordCount

```
override? int DynamicWordCount [get]
```

Gets the dynamic word count for the [NetworkMecanimAnimator](#).

The dynamic word count, which is the maximum of the current total words and the runtime counts, if the application is playing.

Exceptions

System.InvalidOperationException	Thrown when this property is accessed outside of playing.
--	---

6.202 NetworkObject Class Reference

The primary [Fusion](#) component for networked GameObject entities. This stores the object's network identity and manages the object's state and input authority.

Inherits [Behaviour](#).

Public Member Functions

- void [AssignInputAuthority](#) ([PlayerRef](#) player)
Sets which [PlayerRef](#) has Input Authority for this Object.
- void [CopyStateFrom](#) ([NetworkObject](#) source)
Copies the entire State from another [NetworkObject](#)
- void [CopyStateFrom](#) ([NetworkObjectHeaderPtr](#) source)
Copies the entire State from another [NetworkObject](#) based on the [NetworkObjectHeaderPtr](#)
- int [GetLocalAuthorityMask](#) ()
Gets a bitmask of [AuthorityMasks](#) flags, representing the current local authority over this [NetworkObject](#).
- delegate [PriorityLevel PriorityLevelDelegate](#) ([NetworkObject](#) networkObject, [PlayerRef](#) player)
Delegate for determining the priority level of a network object for a specific player.
- void [ReleaseStateAuthority](#) ()
Release the state authority over this [NetworkObject](#) on shared mode.
- void [RemoveInputAuthority](#) ()
Removes input authority from whichever player has it for this object. Only valid when called on a Host or Server peer.

- delegate bool [ReplicateToDelegate](#) ([NetworkObject](#) networkObject, [PlayerRef](#) player)
Delegate for determining if a network object should be replicated to a specific player.
- void [RequestStateAuthority](#) ()
Request state authority over this [NetworkObject](#) on shared mode.
- void [SetPlayerAlwaysInterested](#) ([PlayerRef](#) player, bool alwaysInterested)
Add or remove specific player interest in this [NetworkObject](#). Only the [NetworkObject](#) State Authority can set interest.

Static Public Member Functions

- static int [GetWordCount](#) ([NetworkObject](#) obj)
Calculates the total word count for a given [NetworkObject](#).
- static void [NetworkUnwrap](#) ([NetworkRunner](#) runner, [NetworkId](#) wrapper, ref [NetworkObject](#) result)
Return the [NetworkObject](#) reference on result that matches the provided [NetworkId](#)
- static [NetworkId](#) [NetworkWrap](#) ([NetworkObject](#) obj)
Return the obj [NetworkId](#).
- static [NetworkId](#) [NetworkWrap](#) ([NetworkRunner](#) runner, [NetworkObject](#) obj)
Return the obj [NetworkId](#).
- static implicit operator [NetworkId](#) ([NetworkObject](#) obj)
Converts the Network Object to it's [NetworkId](#).

Public Attributes

- [NetworkObjectFlags](#) Flags
Flags used for network object prefabs and similar
- bool [IsResume](#)
Signal that this [NetworkObject](#) comes from a Resume Spawn
- [NetworkObject](#)[] [NestedObjects](#)
Array of initial child nested [NetworkObject](#) entities, that are children of this Object.
- [NetworkBehaviour](#)[] [NetworkedBehaviours](#)
Array of all [NetworkBehaviours](#) associated with this network entity.
- [NetworkObjectTypeld](#) [NetworkTypeld](#)
The type ID for this prefab or scene object, set when adding to the prefab table and registering scene objects, respectively. All spawned instances of this object will retain this value. Use [NetworkId](#) for the unique ID of network entries.
- [PriorityLevelDelegate](#) [PriorityCallback](#)
Delegate callback used to override priority value for a specific object-player pair
- [ReplicateToDelegate](#) [ReplicateTo](#)
Delegate callback used to override if an object should be replicate to a client or not
- uint [SortKey](#)
Used for whenever objects need to be sorted in a deterministic order, like when registering scene objects.

Protected Member Functions

- virtual void [Awake](#) ()
Awake is called when the script instance is being loaded.
- virtual void [OnDestroy](#) ()
OnDestroy is called when the script instance is being destroyed.

Properties

- bool `HasInputAuthority` [get]
Returns if `Simulation.LocalPlayer` is the designated Input Source for this network entity.
- bool `HasStateAuthority` [get]
Returns if `Simulation.LocalPlayer` is the designated State Source for this network entity.
- `NetworkId? Id` [get]
The unique identifier for this network entity.
- `PlayerRef InputAuthority` [get]
Returns the `PlayerRef` that has Input Authority over this network entity. PlayerRefs are assigned in order from 0 to `MaxPlayers-1` and are re-used as players join and leave. The only caveat is that the server player (if one exists), always gets the last index no matter how many clients are connected.
- bool `IsInSimulation` [get]
If this object is inserted into the simulation
- bool `IsProxy` [get]
Returns if `Simulation.LocalPlayer` is neither the Input nor State Source for this network entity.
- bool `IsSpawnable` [get, set]
Toggles if this `NetworkObject` is included in the `NetworkProjectConfig.PrefabTable`, which will include the prefab in builds as a Spawnable object.
- bool `IsValid` [get]
Returns if this network entity is associated with its `NetworkRunner`, and that runner is not null.
- `Tick LastReceiveTick` [get]
Last tick this object received an update.
- string `Name` [get]
The ID + Unity GameObject name for this entity.
- `RenderSource RenderSource` [get, set]
Returns the `Fusion.RenderSource` for this `Fusion.NetworkBehaviour` instance, indicating how snapshot data will be used to render it.
- float `RenderTime` [get]
Returns the current interpolation time for this object
- `RenderTimeframe RenderTimeframe` [get]
Returns the `Fusion.RenderTimeframe` for this `Fusion.NetworkBehaviour` instance, indicating what snapshot data will be used to render it.
- `NetworkRunner Runner` [get]
The `NetworkRunner` this entity is associated with.
- `PlayerRef StateAuthority` [get]
Returns the `PlayerRef` that has State Authority over this network entity. PlayerRefs are assigned in order from 0 to `MaxPlayers-1` and are re-used as players join and leave. The only caveat is that the server player (if one exists), always gets the last index no matter how many clients are connected.

6.202.1 Detailed Description

The primary `Fusion` component for networked GameObject entities. This stores the object's network identity and manages the object's state and input authority.

6.202.2 Member Function Documentation

6.202.2.1 AssignInputAuthority()

```
void AssignInputAuthority (
    PlayerRef player )
```

Sets which [PlayerRef](#) has Input Authority for this Object.

6.202.2.2 Awake()

```
virtual void Awake ( ) [protected], [virtual]
```

Awake is called when the script instance is being loaded.

6.202.2.3 CopyStateFrom() [1/2]

```
void CopyStateFrom (
    NetworkObject source )
```

Copies the entire State from another [NetworkObject](#)

Parameters

<code>source</code>	NetworkObject to copy the State from
---------------------	--

6.202.2.4 CopyStateFrom() [2/2]

```
void CopyStateFrom (
    NetworkObjectHeaderPtr source )
```

Copies the entire State from another [NetworkObject](#) based on the [NetworkObjectHeaderPtr](#)

Parameters

<code>source</code>	NetworkObjectHeaderPtr to copy the state from
---------------------	---

6.202.2.5 GetLocalAuthorityMask()

```
int GetLocalAuthorityMask ( )
```

Gets a bitmask of [AuthorityMasks](#) flags, representing the current local authority over this [NetworkObject](#).

6.202.2.6 GetWordCount()

```
static int GetWordCount (
    NetworkObject obj ) [static]
```

Calculates the total word count for a given [NetworkObject](#).

Parameters

<i>obj</i>	The NetworkObject for which the word count is to be calculated.
------------	---

Returns

The total word count of the [NetworkObject](#). Returns 0 if the [NetworkObject](#) is not alive.

Exceptions

System.Exception	Thrown when a NetworkBehaviour reference is missing in the NetworkBehaviour array of the NetworkObject .
----------------------------------	--

6.202.2.7 NetworkUnwrap()

```
static void NetworkUnwrap (
    NetworkRunner runner,
    NetworkId wrapper,
    ref NetworkObject result ) [static]
```

Return the [NetworkObject](#) reference on *result* that matches the provided [NetworkId](#)

Parameters

<i>runner</i>	The NetworkRunner that will be used to try to find a NetworkObject with ID equals to <i>wrapper</i>
<i>wrapper</i>	The NetworkId to be searched
<i>result</i>	The found NetworkObject . null if the provided NetworkId is not valid

6.202.2.8 NetworkWrap() [1/2]

```
static NetworkId NetworkWrap (
    NetworkObject obj ) [static]
```

Return the *obj* [NetworkId](#).

Parameters

<i>obj</i>	The NetworkObject to get the ID from
------------	--

Returns

The [NetworkId](#) of the object. Default if the object is not alive (null or destroyed)

6.202.2.9 NetworkWrap() [2/2]

```
static NetworkId NetworkWrap (
    NetworkRunner runner,
    NetworkObject obj ) [static]
```

Return the *obj* [NetworkId](#).

Parameters

<i>runner</i>	The NetworkRunner that <i>obj</i> is assigned to
<i>obj</i>	The NetworkObject to get the ID from

Returns

The [NetworkId](#) of the object. Default if the object is not alive (null or destroyed)

6.202.2.10 OnDestroy()

```
virtual void OnDestroy ( ) [protected], [virtual]
```

OnDestroy is called when the script instance is being destroyed.

6.202.2.11 operator NetworkId()

```
static implicit operator NetworkId (
    NetworkObject obj ) [static]
```

Converts the Network Object to it's [NetworkId](#).

Parameters

<i>obj</i>	The object to convert
------------	-----------------------

Returns

The [NetworkId](#) of the object. Default if the object is not alive (null or destroyed)

6.202.2.12 PriorityLevelDelegate()

```
delegate PriorityLevel PriorityLevelDelegate (
    NetworkObject networkObject,
    PlayerRef player )
```

Delegate for determining the priority level of a network object for a specific player.

Parameters

<i>networkObject</i>	The network object in question.
<i>player</i>	The player for whom the priority level is being determined.

Returns

The priority level of the network object for the player.

6.202.2.13 ReleaseStateAuthority()

```
void ReleaseStateAuthority ( )
```

Release the state authority over this [NetworkObject](#) on shared mode.

6.202.2.14 RemoveInputAuthority()

```
void RemoveInputAuthority ( )
```

Removes input authority from whichever player has it for this object. Only valid when called on a Host or Server peer.

6.202.2.15 ReplicateToDelegate()

```
delegate bool ReplicateToDelegate (
    NetworkObject networkObject,
    PlayerRef player )
```

Delegate for determining if a network object should be replicated to a specific player.

Parameters

<i>networkObject</i>	The network object in question.
<i>player</i>	The player to potentially replicate to.

Returns

True if the object should be replicated to the player, false otherwise.

6.202.2.16 RequestStateAuthority()

```
void RequestStateAuthority ( )
```

Request state authority over this [NetworkObject](#) on shared mode.

6.202.2.17 SetPlayerAlwaysInterested()

```
void SetPlayerAlwaysInterested (
    PlayerRef player,
    bool alwaysInterested )
```

Add or remove specific player interest in this [NetworkObject](#). Only the [NetworkObject](#) State Authority can set interest.

Parameters

<i>player</i>	The player to set interest for
<i>alwaysInterested</i>	If the player should always be interested in this object

6.202.3 Member Data Documentation**6.202.3.1 Flags**

[NetworkObjectFlags](#) Flags

Flags used for network object prefabs and similar

6.202.3.2 IsResume

```
bool IsResume
```

Signal that this [NetworkObject](#) comes from a Resume Spawn

6.202.3.3 NestedObjects

```
NetworkObject [ ] NestedObjects
```

Array of initial child nested [NetworkObject](#) entities, that are children of this Object.

6.202.3.4 NetworkedBehaviours

```
NetworkBehaviour [ ] NetworkedBehaviours
```

Array of all [NetworkBehaviour](#)s associated with this network entity.

6.202.3.5 NetworkTypeId

```
NetworkObjectTypeID NetworkTypeID
```

The type ID for this prefab or scene object, set when adding to the prefab table and registering scene objects, respectively. All spawned instances of this object will retain this value. Use [NetworkId](#) for the unique ID of network entries.

6.202.3.6 PriorityCallback

```
PriorityLevelDelegate PriorityCallback
```

Delegate callback used to override priority value for a specific object-player pair

6.202.3.7 ReplicateTo

```
ReplicateToDelegate ReplicateTo
```

Delegate callback used to override if an object should be replicate to a client or not

6.202.3.8 SortKey

```
uint SortKey
```

Used for whenever objects need to be sorted in a deterministic order, like when registering scene objects.

6.202.4 Property Documentation

6.202.4.1 HasInputAuthority

```
bool HasInputAuthority [get]
```

Returns if [Simulation.LocalPlayer](#) is the designated Input Source for this network entity.

6.202.4.2 HasStateAuthority

```
bool HasStateAuthority [get]
```

Returns if [Simulation.LocalPlayer](#) is the designated State Source for this network entity.

6.202.4.3 Id

```
NetworkId? Id [get]
```

The unique identifier for this network entity.

6.202.4.4 InputAuthority

```
PlayerRef InputAuthority [get]
```

Returns the [PlayerRef](#) that has Input Authority over this network entity. PlayerRefs are assigned in order from 0 to MaxPlayers-1 and are re-used as players join and leave. The only caveat is that the server player (if one exists), always gets the last index no matter how many clients are connected.

6.202.4.5 IsInSimulation

```
bool IsInSimulation [get]
```

If this object is inserted into the simulation

6.202.4.6 IsProxy

```
bool IsProxy [get]
```

Returns if [Simulation.LocalPlayer](#) is neither the Input nor State Source for this network entity.

6.202.4.7 IsSpawnable

```
bool IsSpawnable [get], [set]
```

Toggles if this [NetworkObject](#) is included in the [NetworkProjectConfig.PrefabTable](#), which will include the prefab in builds as a Spawnable object.

6.202.4.8 IsValid

```
bool IsValid [get]
```

Returns if this network entity is associated with its [NetworkRunner](#), and that runner is not null.

6.202.4.9 LastReceiveTick

```
Tick LastReceiveTick [get]
```

Last tick this object received an update.

6.202.4.10 Name

```
string Name [get]
```

The ID + Unity GameObject name for this entity.

6.202.4.11 RenderSource

```
RenderSource RenderSource [get], [set]
```

Returns the [Fusion.RenderSource](#) for this [Fusion.NetworkBehaviour](#) instance, indicating how snapshot data will be used to render it.

6.202.4.12 RenderTime

`float RenderTime [get]`

Returns the current interpolation time for this object

6.202.4.13 RenderTimeframe

`RenderTimeframe RenderTimeframe [get]`

Returns the [Fusion.RenderTimeframe](#) for this [Fusion.NetworkBehaviour](#) instance, indicating what snapshot data will be used to render it.

6.202.4.14 Runner

`NetworkRunner Runner [get]`

The [NetworkRunner](#) this entity is associated with.

6.202.4.15 StateAuthority

`PlayerRef StateAuthority [get]`

Returns the [PlayerRef](#) that has State Authority over this network entity. PlayerRefs are assigned in order from 0 to MaxPlayers-1 and are re-used as players join and leave. The only caveat is that the server player (if one exists), always gets the last index no matter how many clients are connected.

6.203 NetworkObjectFlagsExtensions Class Reference

Extension methods for the NetworkObjectFlags enum.

Static Public Member Functions

- static int [GetVersion](#) (this [NetworkObjectFlags](#) flags)
Returns the version of the flags.
- static bool [IsIgnored](#) (this [NetworkObjectFlags](#) flags)
Check if the flags are ignored.
- static bool [IsVersionCurrent](#) (this [NetworkObjectFlags](#) flags)
Check if the flags are the current version.
- static [NetworkObjectFlags](#) [SetCurrentVersion](#) (this [NetworkObjectFlags](#) flags)
Sets the flags to the current version.
- static [NetworkObjectFlags](#) [SetIgnored](#) (this [NetworkObjectFlags](#) flags, bool value)
Sets the ignored flag on the flags.

6.203.1 Detailed Description

Extension methods for the NetworkObjectFlags enum.

6.203.2 Member Function Documentation

6.203.2.1 GetVersion()

```
static int GetVersion (
    this NetworkObjectFlags flags ) [static]
```

Returns the version of the flags.

Parameters

<i>flags</i>	The flags to get the version of.
--------------	----------------------------------

Returns

The version of the flags.

6.203.2.2 IsIgnored()

```
static bool IsIgnored (
    this NetworkObjectFlags flags ) [static]
```

Check if the flags are ignored.

Parameters

<i>flags</i>	The flags to check.
--------------	---------------------

Returns

True if the flags are ignored, false otherwise.

6.203.2.3 IsVersionCurrent()

```
static bool IsVersionCurrent (
    this NetworkObjectFlags flags ) [static]
```

Check if the flags are the current version.

Parameters

<i>flags</i>	The flags to check.
--------------	---------------------

Returns

True if the flags are the current version, false otherwise.

6.203.2.4 SetCurrentVersion()

```
static NetworkObjectFlags SetCurrentVersion (
    this NetworkObjectFlags flags ) [static]
```

Sets the flags to the current version.

Parameters

<i>flags</i>	The flags to set.
--------------	-------------------

Returns

The flags with the version set to the current version.

6.203.2.5 SetIgnored()

```
static NetworkObjectFlags SetIgnored (
    this NetworkObjectFlags flags,
    bool value ) [static]
```

Sets the ignored flag on the flags.

Parameters

<i>flags</i>	Flags to set the ignored flag on.
<i>value</i>	Ignored flag value.

Returns

The flags with the ignored flag set to the given value.

6.204 NetworkObjectGuid Struct Reference

[NetworkObjectGuid](#)

Inherits [INetworkStruct](#), [IEquatable< NetworkObjectGuid >](#), and [IComparable< NetworkObjectGuid >](#).

Classes

- class [EqualityComparer](#)
EqualityComparer for NetworkObjectGuid

Public Member Functions

- int [CompareTo \(NetworkObjectGuid other\)](#)
Compare the NetworkObjectGuid to another NetworkObjectGuid
- bool [Equals \(NetworkObjectGuid other\)](#)
Check if the NetworkObjectGuid is equal to another NetworkObjectGuid
- override bool [Equals \(object obj\)](#)
Check if the NetworkObjectGuid is equal to another object
- override int [GetHashCode \(\)](#)
Get the hashcode for a NetworkObjectGuid
- [NetworkObjectGuid \(byte *guid\)](#)
*Create a NetworkObjectGuid from a byte**
- [NetworkObjectGuid \(byte\[\] guid\)](#)
Create a NetworkObjectGuid from a byte array
- [NetworkObjectGuid \(long data0, long data1\)](#)
Create a new NetworkObjectGuid
- [NetworkObjectGuid \(string guid\)](#)
Create a new NetworkObjectGuid
- override string [ToString \(\)](#)
Returns a string representation of the NetworkObjectGuid.
- string [ToString \(string format\)](#)
Returns a string representation of the NetworkObjectGuid.
- string [ToUnityGuidString \(\)](#)
Returns a string representation of the NetworkObjectGuid.

Static Public Member Functions

- static implicit [operator Guid \(NetworkObjectGuid guid\)](#)
Implicit conversion from NetworkObjectGuid to Guid
- static implicit [operator NetworkObjectGuid \(Guid guid\)](#)
Implicit conversion from Guid to NetworkObjectGuid
- static [operator NetworkPrefabRef \(NetworkObjectGuid t\)](#)
Explicit conversion from NetworkObjectGuid to NetworkPrefabRef
- static bool [operator!= \(NetworkObjectGuid a, NetworkObjectGuid b\)](#)
Compare two NetworkObjectGuid
- static bool [operator== \(NetworkObjectGuid a, NetworkObjectGuid b\)](#)
Compare two NetworkObjectGuid
- static [NetworkObjectGuid Parse \(string str\)](#)
Parse a NetworkObjectGuid from a string.
- static bool [TryParse \(string str, out NetworkObjectGuid guid\)](#)
Try to parse a string into a NetworkObjectGuid

Public Attributes

- fixed long [RawGuidValue](#) [2]

The Raw Guid Value of the NetworkObjectGuid

Static Public Attributes

- const int [ALIGNMENT](#) = 4
The alignment of the NetworkObjectGuid
- const int [SIZE](#) = 16
The Size of the NetworkObjectGuid in bytes

Properties

- static [NetworkObjectGuid](#) [Empty](#) [get]
The default value of a NetworkObjectGuid
- bool [IsValid](#) [get]
Signal if the NetworkObjectGuid is valid.

6.204.1 Detailed Description

[NetworkObjectGuid](#)

6.204.2 Constructor & Destructor Documentation

6.204.2.1 [NetworkObjectGuid\(\)](#) [1/4]

```
NetworkObjectGuid (
    string guid )
```

Create a new [NetworkObjectGuid](#)

Parameters

<code>guid</code>	The guid to use
-------------------	-----------------

6.204.2.2 [NetworkObjectGuid\(\)](#) [2/4]

```
NetworkObjectGuid (
    long data0,
    long data1 )
```

Create a new [NetworkObjectGuid](#)

Parameters

<i>data0</i>	Data0 of the Guid
<i>data1</i>	Data1 of the Guid

6.204.2.3 NetworkObjectGuid() [3/4]

```
NetworkObjectGuid (
    byte[] guid )
```

Create a [NetworkObjectGuid](#) from a byte array

Parameters

<i>guid</i>	The byte array to create the NetworkObjectGuid from
-------------	---

6.204.2.4 NetworkObjectGuid() [4/4]

```
NetworkObjectGuid (
    byte * guid )
```

Create a [NetworkObjectGuid](#) from a byte*

Parameters

<i>guid</i>	The byte* to create the NetworkObjectGuid from
-------------	--

6.204.3 Member Function Documentation**6.204.3.1 CompareTo()**

```
int CompareTo (
    NetworkObjectGuid other )
```

Compare the [NetworkObjectGuid](#) to another [NetworkObjectGuid](#)

Parameters

<i>other</i>	The other NetworkObjectGuid to compare against
--------------	--

Returns

0 if the [NetworkObjectGuid](#) are equal, -1 if this [NetworkObjectGuid](#) is less than the other, 1 if this [NetworkObjectGuid](#) is greater than the other

6.204.3.2 Equals() [1/2]

```
bool Equals (  
    NetworkObjectGuid other )
```

Check if the [NetworkObjectGuid](#) is equal to another [NetworkObjectGuid](#)

Parameters

<i>other</i>	The other NetworkObjectGuid to check against
--------------	--

Returns

True if the [NetworkObjectGuids](#) are equal, false otherwise

6.204.3.3 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Check if the [NetworkObjectGuid](#) is equal to another object

Parameters

<i>obj</i>	The other object to check against
------------	-----------------------------------

Returns

True if the objects are equal, false otherwise

6.204.3.4 GetHashCode()

```
override int GetHashCode ( )
```

Get the hashcode for a [NetworkObjectGuid](#)

6.204.3.5 operator Guid()

```
static implicit operator Guid (
    NetworkObjectGuid guid ) [static]
```

Implicit conversion from [NetworkObjectGuid](#) to Guid

Parameters

<i>guid</i>	NetworkObjectGuid to convert from
-------------	---

Returns

Guid

6.204.3.6 operator NetworkObjectGuid()

```
static implicit operator NetworkObjectGuid (
    Guid guid ) [static]
```

Implicit conversion from Guid to [NetworkObjectGuid](#)

Parameters

<i>guid</i>	Guid to convert from
-------------	----------------------

Returns

[NetworkObjectGuid](#)

6.204.3.7 operator NetworkPrefabRef()

```
static operator NetworkPrefabRef (
    NetworkObjectGuid t ) [explicit], [static]
```

Explicit conversion from [NetworkObjectGuid](#) to [NetworkPrefabRef](#)

Parameters

<i>t</i>	NetworkObjectGuid to convert from
----------	---

Returns

[NetworkPrefabRef](#)

6.204.3.8 operator"!=()

```
static bool operator!= (
    NetworkObjectGuid a,
    NetworkObjectGuid b ) [static]
```

Compare two NetworkObjectGuid

Returns

True if the NetworkObjectGuid are not equal, false otherwise

6.204.3.9 operator==()

```
static bool operator== (
    NetworkObjectGuid a,
    NetworkObjectGuid b ) [static]
```

Compare two NetworkObjectGuid

Returns

True if the NetworkObjectGuid are equal, false otherwise

6.204.3.10 Parse()

```
static NetworkObjectGuid Parse (
    string str ) [static]
```

Parse a NetworkObjectGuid from a string.

Parameters

<i>str</i>	The string to parse.
------------	----------------------

Returns

The parsed NetworkObjectGuid.

6.204.3.11 ToString() [1/2]

```
override string ToString ( )
```

Returns a string representation of the [NetworkObjectGuid](#).

6.204.3.12 ToString() [2/2]

```
string ToString (
    string format )
```

Returns a string representation of the [NetworkObjectGuid](#).

6.204.3.13 ToUnityGuidString()

```
string ToUnityGuidString ( )
```

Returns a string representation of the [NetworkObjectGuid](#).

6.204.3.14 TryParse()

```
static bool TryParse (
    string str,
    out NetworkObjectGuid guid ) [static]
```

Try to parse a string into a [NetworkObjectGuid](#)

Parameters

<i>str</i>	String to parse
<i>guid</i>	Parsed NetworkObjectGuid

Returns

True if the string was parsed successfully, false otherwise

6.204.4 Member Data Documentation

6.204.4.1 ALIGNMENT

```
const int ALIGNMENT = 4 [static]
```

The alignment of the [NetworkObjectGuid](#)

6.204.4.2 RawGuidValue

```
fixed long RawGuidValue[2]
```

The Raw Guid Value of the [NetworkObjectGuid](#)

6.204.4.3 SIZE

```
const int SIZE = 16 [static]
```

The Size of the [NetworkObjectGuid](#) in bytes

6.204.5 Property Documentation

6.204.5.1 Empty

```
NetworkObjectGuid Empty [static], [get]
```

The default value of a [NetworkObjectGuid](#)

6.204.5.2 IsValid

```
bool IsValid [get]
```

Signal if the [NetworkObjectGuid](#) is valid.

6.205 NetworkObjectGuid.EqualityComparer Class Reference

[EqualityComparer](#) for [NetworkObjectGuid](#)

Inherits [IEqualityComparer< NetworkObjectGuid >](#).

Public Member Functions

- bool [Equals \(NetworkObjectGuid x, NetworkObjectGuid y\)](#)
Check if two NetworkObjectGuid are equals
- int [GetHashCode \(NetworkObjectGuid obj\)](#)
Get the hashcode for a NetworkObjectGuid

6.205.1 Detailed Description

EqualityComparer for NetworkObjectGuid

6.205.2 Member Function Documentation

6.205.2.1 Equals()

```
bool Equals (
    NetworkObjectGuid x,
    NetworkObjectGuid y )
```

Check if two NetworkObjectGuid are equals

6.205.2.2 GetHashCode()

```
int GetHashCode (
    NetworkObjectGuid obj )
```

Get the hashcode for a NetworkObjectGuid

6.206 NetworkObjectHeader Struct Reference

Network object header information for a NetworkObject.

Inherits [INetworkStruct](#), and [IEquatable< NetworkObjectHeader >](#).

Public Member Functions

- bool [Equals \(NetworkObjectHeader other\)](#)
Checks if the current instance of NetworkObjectHeader is equal to another instance of the same type.
- override bool [Equals \(object obj\)](#)
Checks if the current instance of NetworkObjectHeader is equal to another object.
- override int [GetHashCode \(\)](#)
Generates a hash code for the current instance of NetworkObjectHeader.
- override string [ToString \(\)](#)
The string representation of the NetworkObjectHeader.

Static Public Member Functions

- static int * [GetBehaviourChangedTickArray](#) (NetworkObjectHeader *header)
Returns a pointer to the array of behaviour change ticks in a NetworkObjectHeader.
- static int * [GetDataPointer](#) (NetworkObjectHeader *header)
Returns a pointer to the data of a NetworkObjectHeader.
- static int [GetDataWordCount](#) (NetworkObjectHeader *header)
Returns the count of data words in a NetworkObjectHeader.
- static NetworkTRSPData * [GetMainNetworkTRSPData](#) (NetworkObjectHeader *header)
Returns a pointer to the main network TRSP data of a NetworkObjectHeader, if it exists.
- static bool [HasMainNetworkTRSP](#) (NetworkObjectHeader *header)
Checks if a NetworkObjectHeader has a main network TRSP.
- static bool [operator!=](#) (NetworkObjectHeader left, NetworkObjectHeader right)
Determines if two instances of NetworkObjectHeader are not equal.
- static bool [operator==](#) (NetworkObjectHeader left, NetworkObjectHeader right)
Determines if two instances of NetworkObjectHeader are equal.

Public Attributes

- fixed int [_reserved](#) [10]
Reserved space for future use.
- short [BehaviourCount](#)
The number of behaviours in the network object.
- NetworkObjectHeaderFlags [Flags](#)
The flags indicating various states or properties of the network object.
- NetworkId [Id](#)
The unique identifier of the network object.
- PlayerRef [InputAuthority](#)
The player reference who has input authority over the network object.
- NetworkObjectNestingKey [NestingKey](#)
The nesting key of the network object.
- NetworkId [NestingRoot](#)
The unique identifier of the root network object in the nesting hierarchy.
- PlayerRef [StateAuthority](#)
The player reference who has state authority over the network object.
- NetworkObjectTypeid [Type](#)
The type identifier of the network object.
- short [WordCount](#)
The size of the network object's data in words.

Static Public Attributes

- const int [PLAYER_DATA_WORD](#) = 36 / Allocator.REPLICATE_WORD_SIZE
The word index of the player data in the NetworkObjectHeader.
- const int [SIZE](#) = 80
The size of the NetworkObjectHeader in bytes.
- const int [WORDS](#) = SIZE / Allocator.REPLICATE_WORD_SIZE
The size of the NetworkObjectHeader in words.

Properties

- int `ByteCount` [get]
The size of the network object's data in bytes.

6.206.1 Detailed Description

Network object header information for a [NetworkObject](#).

6.206.2 Member Function Documentation

6.206.2.1 Equals() [1/2]

```
bool Equals (
    NetworkObjectHeader other )
```

Checks if the current instance of [NetworkObjectHeader](#) is equal to another instance of the same type.

6.206.2.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Checks if the current instance of [NetworkObjectHeader](#) is equal to another object.

6.206.2.3 GetBehaviourChangedTickArray()

```
static int* GetBehaviourChangedTickArray (
    NetworkObjectHeader * header ) [static]
```

Returns a pointer to the array of behaviour change ticks in a [NetworkObjectHeader](#).

Parameters

<code>header</code>	Pointer to the NetworkObjectHeader .
---------------------	--

Returns

Pointer to the array of behaviour change ticks in the [NetworkObjectHeader](#).

6.206.2.4 GetDataPointer()

```
static int* GetDataPointer (
    NetworkObjectHeader * header ) [static]
```

Returns a pointer to the data of a [NetworkObjectHeader](#).

Parameters

<i>header</i>	Pointer to the NetworkObjectHeader .
---------------	--

Returns

Pointer to the data of the [NetworkObjectHeader](#).

6.206.2.5 GetDataWordCount()

```
static int GetDataWordCount (
    NetworkObjectHeader * header ) [static]
```

Returns the count of data words in a [NetworkObjectHeader](#).

Parameters

<i>header</i>	Pointer to the NetworkObjectHeader .
---------------	--

Returns

The count of data words in the [NetworkObjectHeader](#).

6.206.2.6 GetHashCode()

```
override int GetHashCode ( )
```

Generates a hash code for the current instance of [NetworkObjectHeader](#).

6.206.2.7 GetMainNetworkTRSPData()

```
static NetworkTRSPData* GetMainNetworkTRSPData (
    NetworkObjectHeader * header ) [static]
```

Returns a pointer to the main network TRSP data of a [NetworkObjectHeader](#), if it exists.

Parameters

<i>header</i>	Pointer to the NetworkObjectHeader .
---------------	--

Returns

Pointer to the main network TRSP data of the [NetworkObjectHeader](#) if it exists, null otherwise.

6.206.2.8 HasMainNetworkTRSP()

```
static bool HasMainNetworkTRSP (
    NetworkObjectHeader * header ) [static]
```

Checks if a [NetworkObjectHeader](#) has a main network TRSP.

Parameters

<i>header</i>	Pointer to the NetworkObjectHeader .
---------------	--

Returns

True if the [NetworkObjectHeader](#) has a main network TRSP, false otherwise.

6.206.2.9 operator"!=()

```
static bool operator!= (
    NetworkObjectHeader left,
    NetworkObjectHeader right ) [static]
```

Determines if two instances of [NetworkObjectHeader](#) are not equal.

Returns

True if the instances are not equal; otherwise, false.

6.206.2.10 operator==()

```
static bool operator== (
    NetworkObjectHeader left,
    NetworkObjectHeader right ) [static]
```

Determines if two instances of [NetworkObjectHeader](#) are equal.

Returns

True if the instances are equal; otherwise, false.

6.206.2.11 ToString()

```
override string ToString ( )
```

The string representation of the [NetworkObjectHeader](#).

6.206.3 Member Data Documentation

6.206.3.1 _reserved

```
fixed int _reserved[10]
```

Reserved space for future use.

6.206.3.2 BehaviourCount

```
short BehaviourCount
```

The number of behaviours in the network object.

6.206.3.3 Flags

```
NetworkObjectHeaderFlags Flags
```

The flags indicating various states or properties of the network object.

6.206.3.4 Id

```
NetworkId Id
```

The unique identifier of the network object.

6.206.3.5 InputAuthority

```
PlayerRef InputAuthority
```

The player reference who has input authority over the network object.

6.206.3.6 NestingKey

```
NetworkObjectNestingKey NestingKey
```

The nesting key of the network object.

6.206.3.7 NestingRoot

```
NetworkId NestingRoot
```

The unique identifier of the root network object in the nesting hierarchy.

6.206.3.8 PLAYER_DATA_WORD

```
const int PLAYER_DATA_WORD = 36 / Allocator.REPLICATE_WORD_SIZE [static]
```

The word index of the player data in the [NetworkObjectHeader](#).

6.206.3.9 SIZE

```
const int SIZE = 80 [static]
```

The size of the [NetworkObjectHeader](#) in bytes.

6.206.3.10 StateAuthority

```
PlayerRef StateAuthority
```

The player reference who has state authority over the network object.

6.206.3.11 Type

```
NetworkObjectTypeID Type
```

The type identifier of the network object.

6.206.3.12 WordCount

```
short WordCount
```

The size of the network object's data in words.

6.206.3.13 WORDS

```
const int WORDS = SIZE / Allocator.REPLICATE_WORD_SIZE [static]
```

The size of the [NetworkObjectHeader](#) in words.

6.206.4 Property Documentation

6.206.4.1 ByteCount

```
int ByteCount [get]
```

The size of the network object's data in bytes.

6.207 NetworkObjectHeaderPtr Struct Reference

Represents a pointer to a [NetworkObjectHeader](#). This struct is unsafe because it uses pointers.

Public Attributes

- [NetworkObjectHeader * Ptr](#)
Pointer to a [NetworkObjectHeader](#).

Properties

- [NetworkId Id \[get\]](#)
Gets the Id of the [NetworkObjectHeader](#) this struct points to.
- [NetworkObjectTypeld Type \[get\]](#)
Gets the Type of the [NetworkObjectHeader](#) this struct points to.

6.207.1 Detailed Description

Represents a pointer to a [NetworkObjectHeader](#). This struct is unsafe because it uses pointers.

6.207.2 Member Data Documentation

6.207.2.1 Ptr

`NetworkObjectHeader* Ptr`

Pointer to a [NetworkObjectHeader](#).

6.207.3 Property Documentation

6.207.3.1 Id

`NetworkId Id [get]`

Gets the Id of the [NetworkObjectHeader](#) this struct points to.

6.207.3.2 Type

`NetworkObjectType Id [get]`

Gets the Type of the [NetworkObjectHeader](#) this struct points to.

6.208 NetworkObjectInitializerUnity Class Reference

Initializes network objects for Unity.

Inherits [INetworkObjectInitializer](#).

Public Member Functions

- void [InitializeNetworkState](#) (`NetworkObject` `networkObject`)
Initializes the network object.

6.208.1 Detailed Description

Initializes network objects for Unity.

6.208.2 Member Function Documentation

6.208.2.1 InitializeNetworkState()

```
void InitializeNetworkState (
    NetworkObject networkObject )
```

Initializes the network object.

Parameters

<code>networkObject</code>	The network object to initialize.
----------------------------	-----------------------------------

Implements [INetworkObjectInitializer](#).

6.209 NetworkObjectMeta Class Reference

Meta information about a network object.

Properties

- `NetworkId Id` [get]
Get the [NetworkId](#) of this object.
- `PlayerRef InputAuthority` [get]
Get the Player that has input authority over this object.
- `PlayerRef StateAuthority` [get]
Get the Player that has state authority over this object.
- `NetworkObjectTypeId Type` [get]
Get the [NetworkObjectType](#) of this object.

6.209.1 Detailed Description

Meta information about a network object.

6.209.2 Property Documentation

6.209.2.1 Id

`NetworkId Id` [get]

Get the [NetworkId](#) of this object.

6.209.2.2 InputAuthority

`PlayerRef InputAuthority` [get]

Get the Player that has input authority over this object.

6.209.2.3 StateAuthority

`PlayerRef StateAuthority [get]`

Get the Player that has state authority over this object.

6.209.2.4 Type

`NetworkObjectTypeId Type [get]`

Get the `NetworkObjectTypeid` of this object.

6.210 NetworkObjectNestingKey Struct Reference

A key used to identify a network object nesting.

Inherits `INetworkStruct`, and `IEquatable< NetworkObjectNestingKey >`.

Classes

- class `EqualityComparer`
Implements the `IEqualityComparer` interface.

Public Member Functions

- bool `Equals (NetworkObjectNestingKey other)`
Checks if the current instance of `NetworkObjectNestingKey` is equal to another instance of the same type.
- override bool `Equals (object obj)`
Checks if the current instance of `NetworkObjectNestingKey` is equal to another object.
- override int `GetHashCode ()`
Serves as the default hash function.
- `NetworkObjectNestingKey (int value)`
Initializes a new instance of the `NetworkObjectNestingKey` struct with a specified value.
- override string `ToString ()`
Returns a string that represents the current object.

Public Attributes

- int `Value`
The value of the `NetworkObjectNestingKey`.

Static Public Attributes

- const int **ALIGNMENT** = 4
The alignment of the NetworkObjectNestingKey in bytes.
- const int **SIZE** = 4
The size of the NetworkObjectNestingKey in bytes.

Properties

- bool **IsNone** [get]
Checks if the NetworkObjectNestingKey is none.
- bool **IsValid** [get]
Checks if the NetworkObjectNestingKey is valid.

6.210.1 Detailed Description

A key used to identify a network object nesting.

6.210.2 Constructor & Destructor Documentation

6.210.2.1 NetworkObjectNestingKey()

```
NetworkObjectNestingKey (
    int value )
```

Initializes a new instance of the NetworkObjectNestingKey struct with a specified value.

Parameters

<code>value</code>	The value of the NetworkObjectNestingKey.
--------------------	---

6.210.3 Member Function Documentation

6.210.3.1 Equals() [1/2]

```
bool Equals (
    NetworkObjectNestingKey other )
```

Checks if the current instance of NetworkObjectNestingKey is equal to another instance of the same type.

Parameters

<i>other</i>	An instance of NetworkObjectNestingKey to compare with the current instance.
--------------	--

Returns

True if the current instance is equal to the other parameter; otherwise, false.

6.210.3.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Checks if the current instance of [NetworkObjectNestingKey](#) is equal to another object.

Parameters

<i>obj</i>	An object to compare with the current instance.
------------	---

Returns

True if the current instance is equal to the obj parameter; otherwise, false.

6.210.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current object.

6.210.3.4 ToString()

```
override string ToString ( )
```

Returns a string that represents the current object.

Returns

A string that represents the current object.

6.210.4 Member Data Documentation

6.210.4.1 ALIGNMENT

```
const int ALIGNMENT = 4 [static]
```

The alignment of the [NetworkObjectNestingKey](#) in bytes.

6.210.4.2 SIZE

```
const int SIZE = 4 [static]
```

The size of the [NetworkObjectNestingKey](#) in bytes.

6.210.4.3 Value

```
int Value
```

The value of the [NetworkObjectNestingKey](#).

6.210.5 Property Documentation

6.210.5.1 IsNone

```
bool IsNone [get]
```

Checks if the [NetworkObjectNestingKey](#) is none.

Returns

True if the value of the [NetworkObjectNestingKey](#) is 0; otherwise, false.

6.210.5.2 IsValid

```
bool IsValid [get]
```

Checks if the [NetworkObjectNestingKey](#) is valid.

Returns

True if the value of the [NetworkObjectNestingKey](#) is greater than 0; otherwise, false.

6.211 NetworkObjectNestingKey.EqualityComparer Class Reference

Implements the [IEqualityComparer](#) interface.

Inherits [IEqualityComparer< NetworkObjectNestingKey >](#).

Public Member Functions

- bool [Equals \(NetworkObjectNestingKey x, NetworkObjectNestingKey y\)](#)
Determines whether two [NetworkObjectNestingKey](#) objects are equal.
- int [GetHashCode \(NetworkObjectNestingKey obj\)](#)
Returns a hash code for the specified [NetworkObjectNestingKey](#).

6.211.1 Detailed Description

Implements the [IEqualityComparer](#) interface.

6.211.2 Member Function Documentation

6.211.2.1 Equals()

```
bool Equals (
    NetworkObjectNestingKey x,
    NetworkObjectNestingKey y )
```

Determines whether two [NetworkObjectNestingKey](#) objects are equal.

6.211.2.2 GetHashCode()

```
int GetHashCode (
    NetworkObjectNestingKey obj )
```

Returns a hash code for the specified [NetworkObjectNestingKey](#).

6.212 NetworkObjectPrefabData Class Reference

This class represents the data for a network object prefab.

Inherits [Behaviour](#).

Public Attributes

- [NetworkObjectGuid Guid](#)
The unique identifier for the network object.

Additional Inherited Members

6.212.1 Detailed Description

This class represents the data for a network object prefab.

6.212.2 Member Data Documentation

6.212.2.1 Guid

[NetworkObjectGuid](#) Guid

The unique identifier for the network object.

6.213 NetworkObjectProviderDummy Class Reference

A dummy implementation of the [INetworkObjectProvider](#) interface. This class is used for testing purposes and throws a [NotImplementedException](#) for all its methods.

Inherits [INetworkObjectProvider](#).

Public Member Functions

- [NetworkObjectAcquireResult AcquirePrefabInstance](#) ([NetworkRunner](#) runner, in [NetworkPrefabAcquireContext](#) context, out [NetworkObject](#) instance)
Acquires an instance of a prefab for a network object.
- [void ReleaseInstance](#) ([NetworkRunner](#) runner, in [NetworkObjectReleaseContext](#) context)
Releases an instance of a network object.

6.213.1 Detailed Description

A dummy implementation of the [INetworkObjectProvider](#) interface. This class is used for testing purposes and throws a [NotImplementedException](#) for all its methods.

6.214 NetworkObjectReleaseContext Struct Reference

Represents the context for releasing a network object. This struct is unsafe because it uses pointers.

Public Member Functions

- [NetworkObjectReleaseContext](#) ([NetworkObject](#) obj, [NetworkObjectTypeID](#) typeId, bool isBeingDestroyed, bool isNested)
Initializes a new instance of the [NetworkObjectReleaseContext](#) struct with the specified parameters.
- override string [ToString](#) ()
Returns a string that represents the current object.

Public Attributes

- readonly bool [IsBeingDestroyed](#)
Indicates whether the network object is being destroyed.
- readonly bool [IsNestedObject](#)
Indicates whether the network object is a nested object.
- readonly [NetworkObject](#) Object
The network object to be released.
- readonly [NetworkObjectTypeID](#) Typeld
The type identifier of the network object.

6.214.1 Detailed Description

Represents the context for releasing a network object. This struct is unsafe because it uses pointers.

6.214.2 Constructor & Destructor Documentation

6.214.2.1 NetworkObjectReleaseContext()

```
NetworkObjectReleaseContext (
    NetworkObject obj,
    NetworkObjectTypeID typeId,
    bool isBeingDestroyed,
    bool isNested )
```

Initializes a new instance of the [NetworkObjectReleaseContext](#) struct with the specified parameters.

Parameters

<i>obj</i>	The network object to be released.
<i>typeid</i>	The type identifier of the network object.
<i>isBeingDestroyed</i>	Indicates whether the network object is being destroyed.
<i>isNested</i>	Indicates whether the network object is a nested object.

6.214.3 Member Function Documentation

6.214.3.1 ToString()

```
override string ToString ( )
```

Returns a string that represents the current object.

Returns

A string that represents the current object.

6.214.4 Member Data Documentation

6.214.4.1 IsBeingDestroyed

```
readonly bool IsBeingDestroyed
```

Indicates whether the network object is being destroyed.

6.214.4.2 IsNestedObject

```
readonly bool IsNestedObject
```

Indicates whether the network object is a nested object.

6.214.4.3 Object

```
readonly NetworkObject Object
```

The network object to be released.

6.214.4.4 TypeId

```
readonly NetworkObjectTypeID TypeId
```

The type identifier of the network object.

6.215 NetworkObjectSortKeyComparer Class Reference

This class is used to compare two [NetworkObject](#) instances based on their SortKey. It implements the IComparer interface.

Inherits [IComparer< NetworkObject >](#).

Public Member Functions

- int [Compare \(NetworkObject x, NetworkObject y\)](#)
Compares two NetworkObject instances based on their SortKey.

Static Public Attributes

- static readonly [NetworkObjectSortKeyComparer Instance](#) = new [NetworkObjectSortKeyComparer\(\)](#)
An instance of the NetworkObjectSortKeyComparer class.

6.215.1 Detailed Description

This class is used to compare two [NetworkObject](#) instances based on their SortKey. It implements the IComparer interface.

6.215.2 Member Function Documentation

6.215.2.1 Compare()

```
int Compare (
    NetworkObject x,
    NetworkObject y )
```

Compares two [NetworkObject](#) instances based on their SortKey.

Parameters

x	The first NetworkObject to compare.
y	The second NetworkObject to compare.

Returns

A signed integer that indicates the relative values of x and y.

6.215.3 Member Data Documentation

6.215.3.1 Instance

```
readonly NetworkObjectSortKeyComparer Instance = new NetworkObjectSortKeyComparer() [static]
```

An instance of the [NetworkObjectSortKeyComparer](#) class.

6.216 NetworkObjectSpawnException Class Reference

Network Object Spawn Exception

Inherits Exception.

Public Member Functions

- [NetworkObjectSpawnException](#) ([NetworkSpawnStatus](#) status, [NetworkObjectTypeId?](#) id=null)
Network Object Spawn Exception Constructor

Properties

- override string [Message](#) [get]
Exception Message
- [NetworkSpawnStatus](#) [Status](#) [get]
Network Spawn Status
- [NetworkObjectTypeId?](#) [Typeld](#) [get]
Network Object Type Id

6.216.1 Detailed Description

Network Object Spawn Exception

6.216.2 Constructor & Destructor Documentation

6.216.2.1 NetworkObjectSpawnException()

```
NetworkObjectSpawnException (
    NetworkSpawnStatus status,
    NetworkObjectTypeid? id = null )
```

Network Object Spawn Exception Constructor

Parameters

<code>status</code>	Network Spawn Status
<code>id</code>	Network Object Type Id

6.216.3 Property Documentation

6.216.3.1 Message

```
override string Message [get]
```

Exception Message

6.216.3.2 Status

```
NetworkSpawnStatus Status [get]
```

Network Spawn Status

6.216.3.3 Typeld

```
NetworkObjectType? Typeld [get]
```

Network Object Type Id

6.217 NetworkObjectTypeld Struct Reference

ID for a [NetworkObject](#) Prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).

Inherits [INetworkStruct](#), and [IEquatable<NetworkObjectTypeld>](#).

Classes

- class [EqualityComparer](#)
NetworkObjectTypeld Comparer

Public Member Functions

- bool [Equals \(NetworkObjectTypeld other\)](#)
Checks if the current `NetworkObjectTypeld` instance is equal to another `NetworkObjectTypeld` instance.
- override bool [Equals \(object obj\)](#)
Determines whether the specified object is equal to the current `NetworkObjectTypeld` instance.
- override int [GetHashCode \(\)](#)
Generates a hash code for the current `NetworkObjectTypeld` instance.
- override string [ToString \(\)](#)
Returns a string that represents the current `NetworkObjectTypeld` instance.

Static Public Member Functions

- static [NetworkObjectTypeld FromCustom \(uint raw\)](#)
Creates a `NetworkObjectTypeld` from a raw uint value representing a Custom type.
- static [NetworkObjectTypeld FromPrefabId \(NetworkPrefabId prefabId\)](#)
Creates a `NetworkObjectTypeld` from a `NetworkPrefabId`.
- static [NetworkObjectTypeld FromSceneRefAndObjectIndex \(SceneRef sceneRef, int objIndex, NetworkSceneLoadId loadId=default\)](#)
Creates a `NetworkObjectTypeld` from a `SceneRef`, an object index, and an optional `NetworkSceneLoadId`.
- static [NetworkObjectTypeld FromStruct \(ushort structId\)](#)
Creates a `NetworkObjectTypeld` from a ushort value representing an InternalStruct type.
- static implicit operator [NetworkObjectTypeld \(NetworkPrefabId prefabId\)](#)
Converts a `NetworkPrefabId` instance to a `NetworkObjectTypeld` instance.
- static bool [operator!= \(NetworkObjectTypeld a, NetworkObjectTypeld b\)](#)
Determines whether two `NetworkObjectTypeld` instances are not equal.
- static bool [operator== \(NetworkObjectTypeld a, NetworkObjectTypeld b\)](#)
Determines whether two `NetworkObjectTypeld` instances are equal.

Public Attributes

- uint [_value0](#)
Represents the first part of the value of a `NetworkObjectTypeld`.
- uint [_value1](#)
Represents the second part of the value of a `NetworkObjectTypeld`.

Static Public Attributes

- const int [ALIGNMENT = 4](#)
Represents the alignment of a `NetworkObjectTypeld` in memory.
- const int [MAX_SCENE_OBJECT_INDEX = \(1 << SCENE_OBJECT_INDEX_BITS\) - 1](#)
Represents the maximum number of SceneObjects that can be represented by a `NetworkObjectTypeld`.
- const int [SIZE = 8](#)
Represents the size of a `NetworkObjectTypeld` in bytes.
- const ushort [STRUCT_TYPE_PLAYERDATA = 1](#)

Properties

- uint `AsCustom` [get]
Gets the raw uint value representation of the `NetworkObjectTypeId` assuming it is a Custom type.
- ushort `AsInternalStructId` [get]
Gets the ushort value representation of the `NetworkObjectTypeId` assuming it is an InternalStruct type.
- `NetworkPrefabId AsPrefabId` [get]
Gets the `NetworkPrefabId` representation of the `NetworkObjectTypeId` assuming it is a Prefab.
- `NetworkSceneObjectId AsSceneObjectId` [get]
Gets the `NetworkSceneObjectId` representation of the `NetworkObjectTypeId` assuming it is a SceneObject.
- static `EqualityComparer Comparer = new EqualityComparer()` [get]
An instance of the `NetworkObjectTypeId EqualityComparer` class.
- bool `IsCustom` [get]
Checks if the `NetworkObjectTypeId` is a Custom type.
- bool `IsNone` [get]
Checks if the `NetworkObjectTypeId` is invalid.
- bool `IsPrefab` [get]
Checks if the `NetworkObjectTypeId` is a Prefab.
- bool `IsSceneObject` [get]
Checks if the `NetworkObjectTypeId` is a SceneObject.
- bool `IsStruct` [get]
Checks if the `NetworkObjectTypeId` is an InternalStruct.
- bool `IsValid` [get]
Checks if the `NetworkObjectTypeId` is valid.
- `NetworkTypeIdKind Kind` [get]
Gets the kind of the `NetworkObjectTypeId`.
- static `NetworkObjectTypeId PlayerData` [get]
Represents a `NetworkObjectTypeId` for the PlayerData.

6.217.1 Detailed Description

ID for a `NetworkObject` Prefab which has been cataloged in a `NetworkProjectConfig.PrefabTable`.

6.217.2 Member Function Documentation

6.217.2.1 Equals() [1/2]

```
bool Equals (
    NetworkObjectTypeId other )
```

Checks if the current `NetworkObjectTypeId` instance is equal to another `NetworkObjectTypeId` instance.

Parameters

<code>other</code>	The other <code>NetworkObjectTypeId</code> instance to compare with the current instance.
--------------------	---

6.217.2.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Determines whether the specified object is equal to the current [NetworkObjectTypeId](#) instance.

Parameters

<i>obj</i>	The object to compare with the current NetworkObjectTypeId instance.
------------	--

6.217.2.3 FromCustom()

```
static NetworkObjectTypeId FromCustom (
    uint raw ) [static]
```

Creates a [NetworkObjectTypeId](#) from a raw uint value representing a Custom type.

Parameters

<i>raw</i>	The raw uint value to use.
------------	----------------------------

Returns

A [NetworkObjectTypeId](#) that represents a Custom type with the given raw value.

6.217.2.4 FromPrefabId()

```
static NetworkObjectTypeId FromPrefabId (
    NetworkPrefabId prefabId ) [static]
```

Creates a [NetworkObjectTypeId](#) from a [NetworkPrefabId](#).

Parameters

<i>prefabId</i>	The NetworkPrefabId to use.
-----------------	---

Returns

A [NetworkObjectTypeId](#) that represents a Prefab with the given [NetworkPrefabId](#).

Exceptions

<i>ArgumentException</i>	Thrown when the provided NetworkPrefabId is not valid.
--------------------------	--

6.217.2.5 FromSceneRefAndObjectIndex()

```
static NetworkObjectTypeId FromSceneRefAndObjectIndex (
    SceneRef sceneRef,
    int objIndex,
    NetworkSceneLoadId loadId = default ) [static]
```

Creates a [NetworkObjectTypeld](#) from a [SceneRef](#), an object index, and an optional [NetworkSceneLoadId](#).

Parameters

<i>sceneRef</i>	The SceneRef to use.
<i>objIndex</i>	The object index to use.
<i>loadId</i>	The NetworkSceneLoadId to use. Defaults to default(NetworkSceneLoadId).

Returns

A [NetworkObjectTypeld](#) that represents a SceneObject with the given [SceneRef](#), object index, and [NetworkSceneLoadId](#).

Exceptions

<i>ArgumentException</i>	Thrown when the provided SceneRef is not valid.
<i>ArgumentOutOfRangeException</i>	Thrown when the provided object index is out of range.

6.217.2.6 FromStruct()

```
static NetworkObjectTypeId FromStruct (
    ushort structId ) [static]
```

Creates a [NetworkObjectTypeld](#) from a ushort value representing an InternalStruct type.

Parameters

<i>struct</i> ↪ <i>Id</i>	The ushort value to use.
------------------------------	--------------------------

Returns

A [NetworkObjectTypeId](#) that represents an InternalStruct type with the given ushort value.

6.217.2.7 GetHashCode()

```
override int GetHashCode ( )
```

Generates a hash code for the current [NetworkObjectTypeId](#) instance.

6.217.2.8 operator NetworkObjectTypeId()

```
static implicit operator NetworkObjectTypeId (
    NetworkPrefabId prefabId ) [static]
```

Converts a [NetworkPrefabId](#) instance to a [NetworkObjectTypeId](#) instance.

Parameters

<i>prefab</i> <i>Id</i>	The NetworkPrefabId instance to convert.
----------------------------	--

Returns

A [NetworkObjectTypeId](#) instance that represents a Prefab with the given [NetworkPrefabId](#).

6.217.2.9 operator"!=()

```
static bool operator!= (
    NetworkObjectTypeID a,
    NetworkObjectTypeID b ) [static]
```

Determines whether two [NetworkObjectTypeId](#) instances are not equal.

6.217.2.10 operator==()

```
static bool operator== (
    NetworkObjectTypeID a,
    NetworkObjectTypeID b ) [static]
```

Determines whether two [NetworkObjectTypeId](#) instances are equal.

6.217.2.11 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [NetworkObjectTypeld](#) instance.

6.217.3 Member Data Documentation

6.217.3.1 _value0

```
uint _value0
```

Represents the first part of the value of a [NetworkObjectTypeld](#).

6.217.3.2 _value1

```
uint _value1
```

Represents the second part of the value of a [NetworkObjectTypeld](#).

6.217.3.3 ALIGNMENT

```
const int ALIGNMENT = 4 [static]
```

Represents the alignment of a [NetworkObjectTypeld](#) in memory.

6.217.3.4 MAX_SCENE_OBJECT_INDEX

```
const int MAX_SCENE_OBJECT_INDEX = (1 << SCENE_OBJECT_INDEX_BITS) - 1 [static]
```

Represents the maximum number of SceneObjects that can be represented by a [NetworkObjectTypeld](#).

6.217.3.5 SIZE

```
const int SIZE = 8 [static]
```

Represents the size of a [NetworkObjectTypeld](#) in bytes.

6.217.4 Property Documentation

6.217.4.1 AsCustom

```
uint AsCustom [get]
```

Gets the raw uint value representation of the [NetworkObjectTypeld](#) assuming it is a Custom type.

The raw uint value representation of the [NetworkObjectTypeld](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObjectTypeId is not a Custom type.
----------------------------------	---

6.217.4.2 AsInternalStructId

```
ushort AsInternalStructId [get]
```

Gets the ushort value representation of the [NetworkObjectTypeId](#) assuming it is an InternalStruct type.

The ushort value representation of the [NetworkObjectTypeId](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObjectTypeId is not an InternalStruct type.
----------------------------------	--

6.217.4.3 AsPrefabId

```
NetworkPrefabId AsPrefabId [get]
```

Gets the [NetworkPrefabId](#) representation of the [NetworkObjectTypeId](#) assuming it is a Prefab.

The [NetworkPrefabId](#) representation of the [NetworkObjectTypeId](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObjectTypeId is not a Prefab.
----------------------------------	--

6.217.4.4 AsSceneObjectId

```
NetworkSceneObjectId AsSceneObjectId [get]
```

Gets the [NetworkSceneObjectId](#) representation of the [NetworkObjectTypeId](#) assuming it is a SceneObject.

The [NetworkSceneObjectId](#) representation of the [NetworkObjectTypeId](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObjectTypeId is not a SceneObject.
----------------------------------	---

6.217.4.5 Comparer

```
EqualityComparer Comparer = new EqualityComparer() [static], [get]
```

An instance of the [NetworkObjectTypeId EqualityComparer](#) class.

6.217.4.6 IsCustom

```
bool IsCustom [get]
```

Checks if the [NetworkObjectTypeId](#) is a Custom type.

6.217.4.7 IsNone

```
bool IsNone [get]
```

Checks if the [NetworkObjectTypeId](#) is invalid.

6.217.4.8 IsPrefab

```
bool IsPrefab [get]
```

Checks if the [NetworkObjectTypeId](#) is a Prefab.

6.217.4.9 IsSceneObject

```
bool IsSceneObject [get]
```

Checks if the [NetworkObjectTypeId](#) is a SceneObject.

6.217.4.10 IsStruct

```
bool IsStruct [get]
```

Checks if the [NetworkObjectTypeId](#) is an InternalStruct.

6.217.4.11 IsValid

```
bool IsValid [get]
```

Checks if the [NetworkObjectTypeId](#) is valid.

6.217.4.12 Kind

```
NetworkTypeIdKind Kind [get]
```

Gets the kind of the [NetworkObjectTypeId](#).

6.217.4.13 PlayerData

```
NetworkObjectTypeId PlayerData [static], [get]
```

Represents a [NetworkObjectTypeId](#) for the PlayerData.

6.218 NetworkObjectTypeId.EqualityComparer Class Reference

[NetworkObjectTypeId](#) Comparer

Inherits [IEqualityComparer< NetworkObjectTypeId >](#).

Public Member Functions

- bool [Equals \(NetworkObjectTypeId x, NetworkObjectTypeId y\)](#)
Checks if two [NetworkObjectTypeId](#) instances are equal.
- int [GetHashCode \(NetworkObjectTypeId obj\)](#)
Gets the hash code of a [NetworkObjectTypeId](#) instance.

6.218.1 Detailed Description

[NetworkObjectTypeId](#) Comparer

6.218.2 Member Function Documentation

6.218.2.1 Equals()

```
bool Equals (
    NetworkObjectTypeId x,
    NetworkObjectTypeId y )
```

Checks if two `NetworkObjectTypeId` instances are equal.

6.218.2.2 GetHashCode()

```
int GetHashCode (
    NetworkObjectTypeId obj )
```

Gets the hash code of a `NetworkObjectTypeId` instance.

Parameters

<i>obj</i>	The NetworkObjectTypeld instance.
------------	---

6.219 NetworkPhysicsInfo Struct Reference

Network Physics [INetworkStruct](#)

Inherits [INetworkStruct](#).

Public Attributes

- float [TimeScale](#)
NetworkPhysicsInfo Time Scale

Static Public Attributes

- const int [SIZE](#) = 40
NetworkPhysicsInfo Total Size
- const int [WORD_COUNT](#) = 10
NetworkPhysicsInfo Word Count

6.219.1 Detailed Description

Network Physics [INetworkStruct](#)

6.219.2 Member Data Documentation

6.219.2.1 SIZE

```
const int SIZE = 40 [static]
```

[NetworkPhysicsInfo](#) Total Size

6.219.2.2 TimeScale

```
float TimeScale
```

[NetworkPhysicsInfo](#) Time Scale

6.219.2.3 WORD_COUNT

```
const int WORD_COUNT = 10 [static]
```

[NetworkPhysicsInfo](#) Word Count

6.220 NetworkPrefabAcquireContext Struct Reference

Represents the context for acquiring a prefab instance for a network object. This struct is unsafe because it uses pointers.

Public Member Functions

- [NetworkPrefabAcquireContext](#) ([NetworkPrefabId](#) prefabId, [NetworkObjectMeta](#) meta=null, bool isSyncronous=true, bool dontDestroyOnLoad=false)

Initializes a new instance of the [NetworkPrefabAcquireContext](#) struct with the specified parameters.

Public Attributes

- readonly bool [DontDestroyOnLoad](#)
Indicates whether the network object should not be destroyed on load.
- readonly bool [IsSyncronous](#)
Indicates whether the operation is syncronous.
- readonly [NetworkObjectMeta](#) [Meta](#)
The metadata of the network object.
- readonly [NetworkPrefabId](#) [PrefabId](#)
The identifier of the prefab.

Properties

- int * [Data](#) [get]
Gets the data pointer to the first word of this [NetworkObject](#)'s data block.
- bool [HasHeader](#) [get]
Checks if the Header is not null.

6.220.1 Detailed Description

Represents the context for acquiring a prefab instance for a network object. This struct is unsafe because it uses pointers.

6.220.2 Constructor & Destructor Documentation

6.220.2.1 NetworkPrefabAcquireContext()

```
NetworkPrefabAcquireContext (
    NetworkPrefabId prefabId,
    NetworkObjectMeta meta = null,
    bool isSyncronous = true,
    bool dontDestroyOnLoad = false )
```

Initializes a new instance of the [NetworkPrefabAcquireContext](#) struct with the specified parameters.

Parameters

<i>prefabId</i>	The identifier of the prefab.
<i>meta</i>	The metadata of the network object.
<i>isSynchronous</i>	Indicates whether the operation is synchronous.
<i>dontDestroyOnLoad</i>	Indicates whether the network object should not be destroyed on load.

6.220.3 Member Data Documentation

6.220.3.1 DontDestroyOnLoad

```
readonly bool DontDestroyOnLoad
```

Indicates whether the network object should not be destroyed on load.

6.220.3.2 IsSynchronous

```
readonly bool IsSynchronous
```

Indicates whether the operation is synchronous.

6.220.3.3 Meta

```
readonly NetworkObjectMeta Meta
```

The metadata of the network object.

6.220.3.4 PrefabId

```
readonly NetworkPrefabId PrefabId
```

The identifier of the prefab.

6.220.4 Property Documentation

6.220.4.1 Data

```
int* Data [get]
```

Gets the data pointer to the first word of this [NetworkObject](#)'s data block.

Returns

Data pointer to the first word of this [NetworkObject](#)'s data block.

Exceptions

<i>InvalidOperationException</i>	Thrown when the Header is null.
----------------------------------	---------------------------------

6.220.4.2 HasHeader

bool HasHeader [get]

Checks if the Header is not null.

Returns

True if the Header is not null; otherwise, false.

6.221 NetworkPrefabAttribute Class Reference

Network Prefab Attribute

Inherits PropertyAttribute.

6.221.1 Detailed Description

Network Prefab Attribute

6.222 NetworkPrefabId Struct Reference

ID for a [NetworkObject](#) Prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).

Inherits [INetworkStruct](#), [IEquatable<NetworkPrefabId>](#), [IComparable](#), and [IComparable<NetworkPrefabId>](#).

Classes

- class [EqualityComparer](#)

Equality comparer for NetworkPrefabId.

Public Member Functions

- int [CompareTo](#) ([NetworkPrefabId](#) other)
Compares the [NetworkPrefabId](#) to another [NetworkPrefabId](#).
- int [IComparable](#).[CompareTo](#) (object obj)
Compares the [NetworkPrefabId](#) to another object.
- bool [Equals](#) ([NetworkPrefabId](#) other)
Checks if the [NetworkPrefabId](#) is equal to another [NetworkPrefabId](#).
- override bool [Equals](#) (object obj)
Checks if the [NetworkPrefabId](#) is equal to another object.
- override int [GetHashCode](#) ()
Gets the hash code of the [NetworkPrefabId](#).
- override string [ToString](#) ()
Converts the [NetworkPrefabId](#) to a string.
- string [ToString](#) (bool brackets, bool prefix)
Converts the [NetworkPrefabId](#) to a string with optional brackets and prefix.

Static Public Member Functions

- static [NetworkPrefabId](#) [FromIndex](#) (int index)
Creates a [NetworkPrefabId](#) from an index.
- static [NetworkPrefabId](#) [FromRaw](#) (uint value)
Creates a [NetworkPrefabId](#) from a raw value.
- static bool [operator!=](#) ([NetworkPrefabId](#) a, [NetworkPrefabId](#) b)
Checks if two [NetworkPrefabId](#) are not equal.
- static bool [operator==](#) ([NetworkPrefabId](#) a, [NetworkPrefabId](#) b)
Checks if two [NetworkPrefabId](#) are equal.

Public Attributes

- uint [RawValue](#)
The raw value of the [NetworkPrefabId](#).

Static Public Attributes

- const int [ALIGNMENT](#) = 4
The alignment of a [NetworkPrefabId](#).
- const int [MAX_INDEX](#) = int.MaxValue - 1
The maximum index value of a [NetworkPrefabId](#).
- const int [SIZE](#) = 4
The size of a [NetworkPrefabId](#).

Properties

- int [AsIndex](#) [get]
Converts the [NetworkPrefabId](#) to an index.
- bool [IsNone](#) [get]
Checks if the [NetworkPrefabId](#) is none.
- bool [IsValid](#) [get]
Checks if the [NetworkPrefabId](#) is valid.

6.222.1 Detailed Description

ID for a [NetworkObject](#) Prefab which has been cataloged in a [NetworkProjectConfig.PrefabTable](#).

6.222.2 Member Function Documentation

6.222.2.1 CompareTo() [1/2]

```
int CompareTo (
    NetworkPrefabId other )
```

Compares the [NetworkPrefabId](#) to another [NetworkPrefabId](#).

6.222.2.2 CompareTo() [2/2]

```
int IComparable.CompareTo (
    object obj )
```

Compares the [NetworkPrefabId](#) to another object.

6.222.2.3 Equals() [1/2]

```
bool Equals (
    NetworkPrefabId other )
```

Checks if the [NetworkPrefabId](#) is equal to another [NetworkPrefabId](#).

6.222.2.4 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Checks if the [NetworkPrefabId](#) is equal to another object.

6.222.2.5 FromIndex()

```
static NetworkPrefabId FromIndex (
    int index ) [static]
```

Creates a [NetworkPrefabId](#) from an index.

6.222.2.6 FromRaw()

```
static NetworkPrefabId FromRaw (
    uint value ) [static]
```

Creates a [NetworkPrefabId](#) from a raw value.

6.222.2.7 GetHashCode()

```
override int GetHashCode ( )
```

Gets the hash code of the [NetworkPrefabId](#).

6.222.2.8 operator"!=()

```
static bool operator!= (
    NetworkPrefabId a,
    NetworkPrefabId b ) [static]
```

Checks if two [NetworkPrefabId](#) are not equal.

6.222.2.9 operator==()

```
static bool operator== (
    NetworkPrefabId a,
    NetworkPrefabId b ) [static]
```

Checks if two [NetworkPrefabId](#) are equal.

6.222.2.10 ToString() [1/2]

```
override string ToString ( )
```

Converts the [NetworkPrefabId](#) to a string.

6.222.2.11 ToString() [2/2]

```
string ToString (
    bool brackets,
    bool prefix )
```

Converts the [NetworkPrefabId](#) to a string with optional brackets and prefix.

6.222.3 Member Data Documentation

6.222.3.1 ALIGNMENT

```
const int ALIGNMENT = 4 [static]
```

The alignment of a [NetworkPrefabId](#).

6.222.3.2 MAX_INDEX

```
const int MAX_INDEX = int.MaxValue - 1 [static]
```

The maximum index value of a [NetworkPrefabId](#).

6.222.3.3 RawValue

```
uint RawValue
```

The raw value of the [NetworkPrefabId](#).

6.222.3.4 SIZE

```
const int SIZE = 4 [static]
```

The size of a [NetworkPrefabId](#).

6.222.4 Property Documentation

6.222.4.1 AsIndex

```
int AsIndex [get]
```

Converts the [NetworkPrefabId](#) to an index.

6.222.4.2 IsNone

```
bool IsNone [get]
```

Checks if the [NetworkPrefabId](#) is none.

6.222.4.3 IsValid

```
bool IsValid [get]
```

Checks if the [NetworkPrefabId](#) is valid.

6.223 NetworkPrefabId.EqualityComparer Class Reference

Equality comparer for [NetworkPrefabId](#).

Inherits [IEqualityComparer< NetworkPrefabId >](#).

Public Member Functions

- bool [Equals \(NetworkPrefabId x, NetworkPrefabId y\)](#)
Checks if two [NetworkPrefabId](#) are equal.
- int [GetHashCode \(NetworkPrefabId obj\)](#)
Gets the hash code of a [NetworkPrefabId](#).

6.223.1 Detailed Description

Equality comparer for [NetworkPrefabId](#).

6.223.2 Member Function Documentation

6.223.2.1 Equals()

```
bool Equals (
    NetworkPrefabId x,
    NetworkPrefabId y )
```

Checks if two `NetworkPrefabId` are equal.

6.223.2.2 GetHashCode()

```
int GetHashCode (
    NetworkPrefabId obj )
```

Gets the hash code of a `NetworkPrefabId`.

6.224 NetworkPrefabInfo Struct Reference

Meta data for a `NetworkObject` prefab which has been cataloged in a `NetworkProjectConfig.PrefabTable`.

Public Attributes

- readonly `NetworkObjectHeader * Header`
Header data for the `NetworkObject` prefab.
- readonly bool `IsSynchronous`
Is the prefab supposed to be loaded in a synchronous way. `Fusion` will report an error if this field is set to true and no prefab is returned by `INetworkObjectProvider`.
- readonly `NetworkPrefabId Prefab`
Prefab ID. Use `NetworkPrefabTable.TryAddSource` to look up the actual prefab reference in the `NetworkProjectConfig.PrefabTable`.

Properties

- int * `Data` [get]
Data pointer to the first word of this `NetworkObject`'s data block.
- bool `HasHeader` [get]
If the Header is not null.

6.224.1 Detailed Description

Meta data for a `NetworkObject` prefab which has been cataloged in a `NetworkProjectConfig.PrefabTable`.

6.224.2 Member Data Documentation

6.224.2.1 Header

```
readonly NetworkObjectHeader* Header
```

Header data for the [NetworkObject](#) prefab.

6.224.2.2 IsSynchronous

```
readonly bool IsSynchronous
```

Is the prefab supposed to be loaded in a synchronous way. [Fusion](#) will report an error if this field is set to true and no prefab is returned by [INetworkObjectProvider](#).

6.224.2.3 Prefab

```
readonly NetworkPrefabId Prefab
```

Prefab ID. Use [NetworkPrefabTable.TryAddSource](#) to look up the actual prefab reference in the [NetworkProjectConfig.PrefabTable](#).

6.224.3 Property Documentation

6.224.3.1 Data

```
int* Data [get]
```

Data pointer to the first word of this [NetworkObject](#)'s data block.

6.224.3.2 HasHeader

```
bool HasHeader [get]
```

If the Header is not null.

6.225 NetworkPrefabRef Struct Reference

NetworkPrefabRef

Inherits [INetworkStruct](#), [IEquatable< NetworkPrefabRef >](#), and [IComparable< NetworkPrefabRef >](#).

Classes

- class [EqualityComparer](#)
EqualityComparer for NetworkPrefabRef

Public Member Functions

- int [CompareTo \(NetworkPrefabRef other\)](#)
Compare the NetworkPrefabRef to another NetworkPrefabRef
- bool [Equals \(NetworkPrefabRef other\)](#)
Check if the NetworkPrefabRef is equal to another NetworkPrefabRef
- override bool [Equals \(object obj\)](#)
Check if the NetworkPrefabRef is equal to another object
- override int [GetHashCode \(\)](#)
Get the hashcode for a NetworkPrefabRef
- [NetworkPrefabRef \(byte *guid\)](#)
*Create a NetworkPrefabRef from a byte**
- [NetworkPrefabRef \(byte\[\] guid\)](#)
Create a NetworkPrefabRef from a byte array
- [NetworkPrefabRef \(long data0, long data1\)](#)
Create a new NetworkPrefabRef
- [NetworkPrefabRef \(string guid\)](#)
Create a new NetworkPrefabRef
- override string [ToString \(\)](#)
Returns a string representation of the NetworkPrefabRef.
- string [ToString \(string format\)](#)
Returns a string representation of the NetworkPrefabRef.
- string [ToUnityGuidString \(\)](#)
Returns a string representation of the NetworkPrefabRef.

Static Public Member Functions

- static implicit operator Guid ([NetworkPrefabRef guid](#))
Implicit conversion from NetworkPrefabRef to Guid
- static operator [NetworkObjectGuid \(NetworkPrefabRef t\)](#)
Explicit conversion from NetworkPrefabRef to NetworkObjectGuid
- static implicit operator [NetworkPrefabRef \(Guid guid\)](#)
Implicit conversion from Guid to NetworkPrefabRef
- static bool [operator!= \(NetworkPrefabRef a, NetworkPrefabRef b\)](#)
Compare two NetworkPrefabRef
- static bool [operator== \(NetworkPrefabRef a, NetworkPrefabRef b\)](#)
Compare two NetworkPrefabRef
- static [NetworkPrefabRef Parse \(string str\)](#)
Parse a NetworkPrefabRef from a string.
- static bool [TryParse \(string str, out NetworkPrefabRef guid\)](#)
Try to parse a string into a NetworkPrefabRef

Public Attributes

- fixed long [RawGuidValue](#) [2]

The Raw Guid Value of the NetworkPrefabRef

Static Public Attributes

- const int [ALIGNMENT](#) = 4
The alignment of the NetworkPrefabRef
- const int [SIZE](#) = 16
The Size of the NetworkPrefabRef in bytes

Properties

- static [NetworkPrefabRef](#) [Empty](#) [get]
The default value of a NetworkPrefabRef
- bool [IsValid](#) [get]
Signal if the NetworkPrefabRef is valid.

6.225.1 Detailed Description

[NetworkPrefabRef](#)

A decoupled [NetworkObject](#) prefab reference. Internally stored as a GUID.

6.225.2 Constructor & Destructor Documentation

6.225.2.1 [NetworkPrefabRef\(\)](#) [1/4]

```
NetworkPrefabRef (
    string guid )
```

Create a new [NetworkPrefabRef](#)

Parameters

<code>guid</code>	The guid to use
-------------------	-----------------

6.225.2.2 [NetworkPrefabRef\(\)](#) [2/4]

```
NetworkPrefabRef (
```

```
long data0,  
long data1 )
```

Create a new [NetworkPrefabRef](#)

Parameters

<i>data0</i>	Data0 of the Guid
<i>data1</i>	Data1 of the Guid

6.225.2.3 NetworkPrefabRef() [3/4]

```
NetworkPrefabRef (   
    byte[ ] guid )
```

Create a [NetworkPrefabRef](#) from a byte array

Parameters

<i>guid</i>	The byte array to create the NetworkPrefabRef from
-------------	--

6.225.2.4 NetworkPrefabRef() [4/4]

```
NetworkPrefabRef (   
    byte * guid )
```

Create a [NetworkPrefabRef](#) from a byte*

Parameters

<i>guid</i>	The byte* to create the NetworkPrefabRef from
-------------	---

6.225.3 Member Function Documentation

6.225.3.1 CompareTo()

```
int CompareTo (   
    NetworkPrefabRef other )
```

Compare the [NetworkPrefabRef](#) to another [NetworkPrefabRef](#)

Parameters

<i>other</i>	The other NetworkPrefabRef to compare against
--------------	---

Returns

0 if the [NetworkPrefabRef](#) are equal, -1 if this [NetworkPrefabRef](#) is less than the other, 1 if this [NetworkPrefabRef](#) is greater than the other

6.225.3.2 Equals() [1/2]

```
bool Equals (  
    NetworkPrefabRef other )
```

Check if the [NetworkPrefabRef](#) is equal to another [NetworkPrefabRef](#)

Parameters

<i>other</i>	The other NetworkPrefabRef to check against
--------------	---

Returns

True if the NetworkPrefabRefs are equal, false otherwise

6.225.3.3 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Check if the [NetworkPrefabRef](#) is equal to another object

Parameters

<i>obj</i>	The other object to check against
------------	-----------------------------------

Returns

True if the objects are equal, false otherwise

6.225.3.4 GetHashCode()

```
override int GetHashCode ( )
```

Get the hashCode for a [NetworkPrefabRef](#)

6.225.3.5 operator Guid()

```
static implicit operator Guid (
    NetworkPrefabRef guid ) [static]
```

Implicit conversion from [NetworkPrefabRef](#) to Guid

Parameters

<i>guid</i>	NetworkPrefabRef to convert from
-------------	--

Returns

Guid

6.225.3.6 operator NetworkObjectGuid()

```
static operator NetworkObjectGuid (
    NetworkPrefabRef t ) [explicit], [static]
```

Explicit conversion from [NetworkPrefabRef](#) to NetworkObjectGuid

Parameters

<i>t</i>	NetworkPrefabRef to convert from
----------	--

Returns

NetworkObjectGuid

6.225.3.7 operator NetworkPrefabRef()

```
static implicit operator NetworkPrefabRef (
    Guid guid ) [static]
```

Implicit conversion from Guid to [NetworkPrefabRef](#)

Parameters

<i>guid</i>	Guid to convert from
-------------	----------------------

Returns

[NetworkPrefabRef](#)

6.225.3.8 operator"!=()

```
static bool operator!= (
    NetworkPrefabRef a,
    NetworkPrefabRef b ) [static]
```

Compare two [NetworkPrefabRef](#)

Returns

True if the [NetworkPrefabRef](#) are not equal, false otherwise

6.225.3.9 operator==()

```
static bool operator== (
    NetworkPrefabRef a,
    NetworkPrefabRef b ) [static]
```

Compare two [NetworkPrefabRef](#)

Returns

True if the [NetworkPrefabRef](#) are equal, false otherwise

6.225.3.10 Parse()

```
static NetworkPrefabRef Parse (
    string str ) [static]
```

Parse a [NetworkPrefabRef](#) from a string.

Parameters

<code>str</code>	The string to parse.
------------------	----------------------

Returns

The parsed [NetworkPrefabRef](#).

6.225.3.11 ToString() [1/2]

```
override string ToString ( )
```

Returns a string representation of the [NetworkPrefabRef](#).

6.225.3.12 ToString() [2/2]

```
string ToString (
    string format )
```

Returns a string representation of the [NetworkPrefabRef](#).

6.225.3.13 ToUnityGuidString()

```
string ToUnityGuidString ( )
```

Returns a string representation of the [NetworkPrefabRef](#).

6.225.3.14 TryParse()

```
static bool TryParse (
    string str,
    out NetworkPrefabRef guid ) [static]
```

Try to parse a string into a [NetworkPrefabRef](#)

Parameters

<i>str</i>	String to parse
<i>guid</i>	Parsed NetworkPrefabRef

Returns

True if the string was parsed successfully, false otherwise

6.225.4 Member Data Documentation

6.225.4.1 ALIGNMENT

```
const int ALIGNMENT = 4 [static]
```

The alignment of the [NetworkPrefabRef](#)

6.225.4.2 RawGuidValue

```
fixed long RawGuidValue[2]
```

The Raw Guid Value of the [NetworkPrefabRef](#)

6.225.4.3 SIZE

```
const int SIZE = 16 [static]
```

The Size of the [NetworkPrefabRef](#) in bytes

6.225.5 Property Documentation

6.225.5.1 Empty

```
NetworkPrefabRef Empty [static], [get]
```

The default value of a [NetworkPrefabRef](#)

6.225.5.2 IsValid

```
bool IsValid [get]
```

Signal if the [NetworkPrefabRef](#) is valid.

6.226 NetworkPrefabRef.EqualityComparer Class Reference

[EqualityComparer](#) for [NetworkPrefabRef](#)

Inherits [IEqualityComparer< NetworkPrefabRef >](#).

Public Member Functions

- bool [Equals \(NetworkPrefabRef x, NetworkPrefabRef y\)](#)
Check if two NetworkPrefabRef are equals
- int [GetHashCode \(NetworkPrefabRef obj\)](#)
Get the hashCode for a NetworkPrefabRef

6.226.1 Detailed Description

EqualityComparer for NetworkPrefabRef

6.226.2 Member Function Documentation

6.226.2.1 Equals()

```
bool Equals (
    NetworkPrefabRef x,
    NetworkPrefabRef y )
```

Check if two NetworkPrefabRef are equals

6.226.2.2 GetHashCode()

```
int GetHashCode (
    NetworkPrefabRef obj )
```

Get the hashCode for a NetworkPrefabRef

6.227 NetworkPrefabTable Class Reference

Class representing a table of network prefabs.

Public Member Functions

- int [AddInstance \(NetworkPrefabId prefabId\)](#)
Add an instance of a prefab id.
- void [AddSource \(INetworkPrefabSource source\)](#)
Adds a prefab source to the table.
- void [Clear \(\)](#)
Clear the prefab table.
- bool [Contains \(NetworkPrefabId prefabId\)](#)
Returns true if the prefab table contains a prefab with the given id.
- IEnumerable<(NetworkPrefabId, INetworkPrefabSource)> [GetEntries \(\)](#)
Returns all entries in the table.
- [NetworkObjectGuid GetGuid \(NetworkPrefabId prefabId\)](#)
Gets a prefab guid by id.
- [NetworkPrefabId GetId \(NetworkObjectGuid guid\)](#)
Gets a prefab id by guid.
- int [GetInstanceCount \(NetworkPrefabId prefabId\)](#)
Get the instance count of a prefab id.
- [INetworkPrefabSource GetSource \(NetworkObjectGuid guid\)](#)
Gets a prefab source by guid.
- [INetworkPrefabSource GetSource \(NetworkPrefabId prefabId\)](#)
Gets a prefab source by id.
- bool [IsAcquired \(NetworkPrefabId prefabId\)](#)
Signal if a prefab id has been acquired.
- [NetworkObject Load \(NetworkPrefabId prefabId, bool isSynchronous\)](#)
Load a prefab by id.
- int [RemoveInstance \(NetworkPrefabId prefabId\)](#)
Remove an instance of a prefab id.
- bool [TryAddSource \(INetworkPrefabSource source, out NetworkPrefabId id\)](#)
Tries to add a prefab source to the table.
- bool [Unload \(NetworkPrefabId prefabId\)](#)
Unload a prefab by id.
- void [UnloadAll \(\)](#)
Unload all prefabs.
- int [UnloadUnreferenced \(bool includeIncompleteLoads=false\)](#)
Unload all unreferenced prefabs.

Public Attributes

- [NetworkPrefabTableOptions Options = NetworkPrefabTableOptions.Default](#)
Options for the NetworkPrefabTable.

Properties

- IReadOnlyList<INetworkPrefabSource> [Prefabs \[get\]](#)
All prefab sources.
- int [Version \[get\]](#)
Prefab table version. Incremented every time a change occurs.

6.227.1 Detailed Description

Class representing a table of network prefabs.

6.227.2 Member Function Documentation

6.227.2.1 AddInstance()

```
int AddInstance (
    NetworkPrefabId prefabId )
```

Add an instance of a prefab id.

Parameters

<i>prefab</i> ↗ <i>Id</i>	Id of the prefab.
------------------------------	-------------------

Returns

The new instance count, or 0 if not found.

6.227.2.2 AddSource()

```
void AddSource (
    INetworkPrefabSource source )
```

Adds a prefab source to the table.

Parameters

<i>source</i>	Prefab source to add.
---------------	-----------------------

Exceptions

<i>ArgumentException</i>	Thrown if a prefab source with the same guid already exists.
--------------------------	--

6.227.2.3 Clear()

```
void Clear ( )
```

Clear the prefab table.

6.227.2.4 Contains()

```
bool Contains (
    NetworkPrefabId prefabId )
```

Returns true if the prefab table contains a prefab with the given id.

Parameters

<i>prefab</i> ↳ <i>Id</i>	Id of the prefab.
---------------------------------	-------------------

Returns

True if the prefab table contains a prefab with the given id.

6.227.2.5 GetEntries()

```
IEnumerable<(NetworkPrefabId, INetworkPrefabSource)> GetEntries ( )
```

Returns all entries in the table.

6.227.2.6 GetGuid()

```
NetworkObjectGuid GetGuid (
    NetworkPrefabId prefabId )
```

Gets a prefab guid by id.

Parameters

<i>prefab</i> ↳ <i>Id</i>	Id of the prefab source.
---------------------------------	--------------------------

Returns

The prefab guid, or default if not found.

6.227.2.7 GetById()

```
NetworkPrefabId GetById (
    NetworkObjectGuid guid )
```

Gets a prefab id by guid.

Parameters

<i>guid</i>	Guid of the prefab source.
-------------	----------------------------

Returns

The prefab id, or default if not found.

6.227.2.8 GetInstancesCount()

```
int GetInstancesCount (
    NetworkPrefabId prefabId )
```

Get the instance count of a prefab id.

Parameters

<i>prefabId</i>	Id of the prefab.
-----------------	-------------------

Returns

The instance count, or 0 if not found.

6.227.2.9 GetSource() [1/2]

```
INetworkPrefabSource GetSource (
    NetworkObjectGuid guid )
```

Gets a prefab source by guid.

Parameters

<i>guid</i>	Guid of the prefab source.
-------------	----------------------------

Returns

The prefab source, or default if not found.

6.227.2.10 GetSource() [2/2]

```
INetworkPrefabSource GetSource (
    NetworkPrefabId prefabId )
```

Gets a prefab source by id.

Parameters

<i>prefab</i> <i>Id</i>	Id of the prefab source.
----------------------------	--------------------------

Returns

The prefab source, or default if not found.

6.227.2.11 IsAcquired()

```
bool IsAcquired (
    NetworkPrefabId prefabId )
```

Signal if a prefab id has been acquired.

Parameters

<i>prefab</i> <i>Id</i>	Id of the prefab.
----------------------------	-------------------

Returns

True if the prefab id has been acquired.

6.227.2.12 Load()

```
NetworkObject Load (
    NetworkPrefabId prefabId,
    bool isSynchronous )
```

Load a prefab by id.

Parameters

<i>prefabId</i>	Id of the prefab.
<i>isSynchronous</i>	If true, the load will be synchronous.

Returns

The loaded prefab, or null if not found.

6.227.2.13 RemoveInstance()

```
int RemoveInstance (
    NetworkPrefabId prefabId )
```

Remove an instance of a prefab id.

Parameters

<i>prefabId</i>	Id of the prefab.
-----------------	-------------------

Returns

The new instance count, or 0 if not found.

6.227.2.14 TryAddSource()

```
bool TryAddSource (
    INetworkPrefabSource source,
    out NetworkPrefabId id )
```

Tries to add a prefab source to the table.

Parameters

<i>source</i>	Prefab source to add.
<i>id</i>	Id of the prefab source.

Returns

True if the prefab source was added, false otherwise.

Exceptions

<i>ArgumentNullException</i>	Thrown if <i>source</i> is null.
------------------------------	----------------------------------

6.227.2.15 Unload()

```
bool Unload (
    NetworkPrefabId prefabId )
```

Unload a prefab by id.

Parameters

<i>prefab</i> ↪ <i>Id</i>	Id of the prefab.
------------------------------	-------------------

Returns

True if the prefab was unloaded, false otherwise.

6.227.2.16 UnloadAll()

```
void UnloadAll ( )
```

Unload all prefabs.

6.227.2.17 UnloadUnreferenced()

```
int UnloadUnreferenced (
    bool includeIncompleteLoads = false )
```

Unload all unreferenced prefabs.

Parameters

<i>includeIncompleteLoads</i>	If true, incomplete loads will be unloaded as well.
-------------------------------	---

Returns

The number of prefabs unloaded.

6.227.3 Member Data Documentation

6.227.3.1 Options

```
NetworkPrefabTableOptions Options = NetworkPrefabTableOptions.Default
```

Options for the [NetworkPrefabTable](#).

6.227.4 Property Documentation

6.227.4.1 Prefabs

```
IReadOnlyList<INetworkPrefabSource> Prefabs [get]
```

All prefab sources.

6.227.4.2 Version

```
int Version [get]
```

Prefab table version. Incremented every time a change occurs.

6.228 NetworkPrefabTableOptions Struct Reference

Options for the [NetworkPrefabTable](#).

Public Attributes

- bool [UnloadPrefabOnReleasingLastInstance](#)
If true, prefabs will be unloaded when the last instance is released.
- bool [UnloadUnusedPrefabsOnShutdown](#)
If true, all prefabs will be unloaded on shutdown.

Static Public Attributes

- static [NetworkPrefabTableOptions Default](#)
Default options.

6.228.1 Detailed Description

Options for the [NetworkPrefabTable](#).

6.228.2 Member Data Documentation

6.228.2.1 Default

```
NetworkPrefabTableOptions Default [static]
```

Initial value:

```
= new NetworkPrefabTableOptions() {
    UnloadPrefabOnReleasingLastInstance = false,
    UnloadUnusedPrefabsOnShutdown = true,
}
```

Default options.

6.228.2.2 UnloadPrefabOnReleasingLastInstance

```
bool UnloadPrefabOnReleasingLastInstance
```

If true, prefabs will be unloaded when the last instance is released.

6.228.2.3 UnloadUnusedPrefabsOnShutdown

```
bool UnloadUnusedPrefabsOnShutdown
```

If true, all prefabs will be unloaded on shutdown.

6.229 NetworkProjectConfig Class Reference

The core [Fusion](#) config file that is shared with all peers at startup.

Public Types

- enum class [PeerModes](#)

Options for running one or multiple peers in one Unity instance. Multiple is useful for testing multiple players/clients inside of the Unity editor without needing to build executables. Each peer is assigned its own independent physics scene and [NetworkRunner](#) instance.

- enum class [ReplicationFeatures](#)

Public Member Functions

- int? [GetExecutionOrder](#) (Type type)
Get the execution order for a given type. If the type is registered, returns null.
- override string [ToString](#) ()
ToString() implementation.

Static Public Member Functions

- static [NetworkProjectConfig Deserialize](#) (string data)
De-serialize a NetworkProjectConfig from a JSON string (typically sent by the Room's Creator).
- static string [Serialize](#) (NetworkProjectConfig config)
Serialize a NetworkProjectConfig into a JSON string.
- static void [UnloadGlobal](#) ()
Unloads Global, if already loaded. If loading Global has faulted, resets the state and next call to the Global accessor will attempt to load the config again.

Public Attributes

- string[] [AssembliesToWeave](#)
Names of assemblies Fusion is going to weave. Not case sensitive.
- bool [CheckNetworkedPropertiesBeingEmpty](#) = false
If set, the weaver will check if NetworkedAttribute properties getters and setters are empty.
- bool [CheckRpcAttributeUsage](#) = false
If set, the weaver will check if RpcAttribute is used in types that do not support it. This requires all types to be scanned and can increase weaving duration.
- EncryptionConfig [EncryptionConfig](#) = new EncryptionConfig()
Reference to EncryptionConfig settings for this NetworkProjectConfig
- bool [EnqueueIncompleteSynchronousSpawns](#)
This flag changes the behaviour of NetworkRunner.Spawn<T> to return null (instead of throwing an exception) and NetworkRunner.TrySpawn<T>) to return NetworkSpawnStatus.Queued if Fusion was unable to load a prefab synchronously (e.g. because it was Addressable). Fusion will enqueue the spawn and attempt to perform it the next frame, until successful. Useful for transition from Fusion 1.x.
- HeapConfiguration [Heap](#) = new HeapConfiguration()
Heap Settings
- bool [HideNetworkObjectInactivityGuard](#) = false
Inactive NetworkObject need special handling in case they get destroyed without ever being activated. This is achieved with adding a nested GameObject called "NetworkObjectInactivityGuard" that tracks the OnDestroy message. HideNetworkObjectInactivityGuard can be used to control whether these guards are visible in the hierarchy or not.
- HostMigrationConfig [HostMigration](#) = new HostMigrationConfig()
Reference to HostMigration settings for this NetworkProjectConfig
- bool [InvokeRenderInBatchMode](#) = true
Signal if the SimulationBehaviour.Render callbacks should be invoked in Batch Mode.
- LagCompensationSettings [LagCompensation](#) = new LagCompensationSettings()
Advanced lag compensation buffer settings.
- NetworkConfiguration [Network](#) = new NetworkConfiguration()
Reference to NetworkConfiguration settings for this NetworkProjectConfig.
- NetworkSimulationConfiguration [NetworkConditions](#) = new NetworkSimulationConfiguration()
Settings for simulating network conditions of latency and loss.
- bool [NetworkIdIsObjectName](#)

- **bool NullChecksForNetworkedProperties = true**
If set, the weaver will add a check to all [Networked] properties on each NetworkBehaviour to verify if owing NetworkObject has been attached to.
- **PeerModes PeerMode**
Setting for whether multiple peers can run per Unity instance (typically to allow easy testing of multiple peers inside of the editor).
- **NetworkPrefabTable PrefabTable = new NetworkPrefabTable()**
Reference to the NetworkPrefabTable instance for this NetworkProjectConfig.
- **SimulationConfig Simulation = new SimulationConfig()**
Reference to SimulationConfig settings for this NetworkProjectConfig.
- **TimeSyncConfiguration TimeSynchronizationOverride**
this can be used to override the time synchronization from code
- **string Typeld = CurrentTypeld**
Current NetworkProjectConfig Type ID
- **bool UseSerializableDictionary = true**
Use Fusion.SerializableDictionary to store [Networked] dictionary properties initial value. If unchecked, the weaver will emit System.Generic.Dictionary instead - a type that's not Unity-serializable, but custom serializers (e.g. Odin) may support it.
- **int Version = CurrentVersion**
Current NetworkProjectConfig version

Static Public Attributes

- **static NetworkRunner BuildTypes**
Get the version information for the Fusion.Runtime.dll.
- **const string CurrentTypeld = nameof(NetworkProjectConfig)**
Current NetworkProjectConfig Type ID
- **const int CurrentVersion = 1**
Current NetworkProjectConfig version
- **const string DefaultResourceName = nameof(NetworkProjectConfig)**
Default file name for the NetworkProjectConfig asset

Properties

- **static NetworkProjectConfig Global [get]**
Reference for the default NetworkProjectConfig. By default, loads a resource named "NetworkProjectConfig". This behaviour can be changed with an attribute FusionGlobalScriptableObjectLoaderMethodAttribute.

6.229.1 Detailed Description

The core **Fusion** config file that is shared with all peers at startup.

6.229.2 Member Enumeration Documentation

6.229.2.1 PeerModes

```
enum PeerModes [strong]
```

Options for running one or multiple peers in one Unity instance. Multiple is useful for testing multiple players/clients inside of the Unity editor without needing to build executables. Each peer is assigned its own independent physics scene and **NetworkRunner** instance.

Enumerator

Single	This is the normal use case, where every build and the editor run a single server, host or client peer.
Multiple	This is the optional use case, which allows running multiple peers in the Unity editor, or in a build.

6.229.2.2 ReplicationFeatures

```
enum ReplicationFeatures [strong]
```

Eventual Consistency [NetworkObject](#) state replication options.

Scheduling enables automatic prioritization of objects when culling occurs (when Object's are not replicated due to exceeding per tick data limits, they increase in priority on the following [Tick](#)).

Interest Management enables [NetworkObject](#) Area Of Interest and Explicit Interest features.

Enumerator

None	No special replication handling. This setting is ideal if your project never exceeds per tick data limits during gameplay.
Scheduling	When changed Network Objects are not replicated by the server to a client due to culling (data per tick limit was reached) the server increases the priority of that Network Object for the next outgoing Tick update to that client.
SchedulingAndInterestManagement	In addition to scheduling, Interest Management features are also enabled (Area Of Interest and Explicit Interest).

6.229.3 Member Function Documentation**6.229.3.1 Deserialize()**

```
static NetworkProjectConfig Deserialize (
    string data ) [static]
```

De-serialize a [NetworkProjectConfig](#) from a JSON string (typically sent by the Room's Creator).

Parameters

<i>data</i>	JSON string of a serialized NetworkProjectConfig
-------------	--

Returns

[NetworkProjectConfig](#) reference de-serialized from JSON string

6.229.3.2 GetExecutionOrder()

```
int? GetExecutionOrder (
    Type type )
```

Get the execution order for a given type. If the type is registered, returns null.

Parameters

<i>type</i>	Type to check for the execution order.
-------------	--

Returns

Execution order for the type, or null if not registered.

6.229.3.3 Serialize()

```
static string Serialize (
    NetworkProjectConfig config ) [static]
```

Serialize a [NetworkProjectConfig](#) into a JSON string.

Parameters

<i>config</i>	NetworkProjectConfig reference
---------------	--

Returns

JSON String

6.229.3.4 ToString()

```
override string ToString ( )
```

[ToString\(\)](#) implementation.

6.229.3.5 UnloadGlobal()

```
static void UnloadGlobal ( ) [static]
```

Unloads [Global](#), if already loaded. If loading [Global](#) has faulted, resets the state and next call to the [Global](#) accessor will attempt to load the config again.

6.229.4 Member Data Documentation

6.229.4.1 AssembliesToWeave

```
string [] AssembliesToWeave
```

Initial value:

```
= new string[] {  
    "Fusion.Unity",  
    "Assembly-CSharp",  
    "Assembly-CSharp-firstpass",  
    "Fusion.Addons.Physics",  
    "Fusion.Addons.FSM",  
}
```

Names of assemblies [Fusion](#) is going to weave. Not case sensitive.

6.229.4.2 BuildTypes

```
NetworkRunner. BuildTypes [static]
```

Get the version information for the Fusion.Runntime.dll.

6.229.4.3 CheckNetworkedPropertiesBeingEmpty

```
bool CheckNetworkedPropertiesBeingEmpty = false
```

If set, the weaver will check if [NetworkedAttribute](#) properties getters and setters are empty.

6.229.4.4 CheckRpcAttributeUsage

```
bool CheckRpcAttributeUsage = false
```

If set, the weaver will check if [RpcAttribute](#) is used in types that do not support it. This requires all types to be scanned and can increase weaving duration.

6.229.4.5 CurrentTypeId

```
const string CurrentTypeId = nameof(NetworkProjectConfig) [static]
```

Current [NetworkProjectConfig](#) Type ID

6.229.4.6 CurrentVersion

```
const int CurrentVersion = 1 [static]
```

Current [NetworkProjectConfig](#) version

6.229.4.7 DefaultResourceName

```
const string DefaultResourceName = nameof(NetworkProjectConfig) [static]
```

Default file name for the [NetworkProjectConfig](#) asset

6.229.4.8 EncryptionConfig

```
EncryptionConfig EncryptionConfig = new EncryptionConfig()
```

Reference to [EncryptionConfig](#) settings for this [NetworkProjectConfig](#)

6.229.4.9 EnqueueIncompleteSynchronousSpawns

```
bool EnqueueIncompleteSynchronousSpawns
```

This flag changes the behaviour of [NetworkRunner.Spawn<T>](#) to return null (instead of throwing an exception) and [NetworkRunner.TrySpawn<T>](#) to return [NetworkSpawnStatus.Queued](#) if [Fusion](#) was unable to load a prefab synchronously (e.g. because it was Addressable). [Fusion](#) will enqueue the spawn and attempt to perform it the next frame, until successful. Useful for transition from [Fusion](#) 1.x.

6.229.4.10 Heap

```
HeapConfiguration Heap = new HeapConfiguration()
```

Heap Settings

6.229.4.11 HideNetworkObjectInactivityGuard

```
bool HideNetworkObjectInactivityGuard = false
```

Inactive [NetworkObject](#) need special handling in case they get destroyed without ever being activated. This is achieved with adding a nested GameObject called "NetworkObjectInactivityGuard" that tracks the OnDestroy message. [HideNetworkObjectInactivityGuard](#) can be used to control whether these guards are visible in the hierarchy or not.

6.229.4.12 HostMigration

```
HostMigrationConfig HostMigration = new HostMigrationConfig()
```

Reference to [HostMigration](#) settings for this [NetworkProjectConfig](#)

6.229.4.13 InvokeRenderInBatchMode

```
bool InvokeRenderInBatchMode = true
```

Signal if the [SimulationBehaviour.Render](#) callbacks should be invoked in Batch Mode.

6.229.4.14 LagCompensation

```
LagCompensationSettings LagCompensation = new LagCompensationSettings()
```

Advanced lag compensation buffer settings.

6.229.4.15 Network

```
NetworkConfiguration Network = new NetworkConfiguration()
```

Reference to [NetworkConfiguration](#) settings for this [NetworkProjectConfig](#).

6.229.4.16 NetworkConditions

```
NetworkSimulationConfiguration NetworkConditions = new NetworkSimulationConfiguration()
```

Settings for simulating network conditions of latency and loss.

6.229.4.17 NetworkIdIsObjectName

```
bool NetworkIdIsObjectName
```

Signal if the [NetworkId](#) of the [NetworkObject](#) should be included on the name of the GameObject.

6.229.4.18 NullChecksForNetworkedProperties

```
bool NullChecksForNetworkedProperties = true
```

If set, the weaver will add a check to all [Networked] properties on each [NetworkBehaviour](#) to verify if owing [NetworkObject](#) has been attached to.

6.229.4.19 PeerMode

```
PeerModes PeerMode
```

Setting for whether multiple peers can run per Unity instance (typically to allow easy testing of multiple peers inside of the editor).

6.229.4.20 PrefabTable

```
NetworkPrefabTable PrefabTable = new NetworkPrefabTable()
```

Reference to the [NetworkPrefabTable](#) instance for this [NetworkProjectConfig](#).

6.229.4.21 Simulation

```
SimulationConfig Simulation = new SimulationConfig()
```

Reference to [SimulationConfig](#) settings for this [NetworkProjectConfig](#).

6.229.4.22 TimeSynchronizationOverride

```
TimeSyncConfiguration TimeSynchronizationOverride
```

this can be used to override the time synchronization from code

6.229.4.23 TypeId

```
string TypeId = CurrentTypeId
```

Current [NetworkProjectConfig](#) Type ID

6.229.4.24 UseSerializableDictionary

```
bool UseSerializableDictionary = true
```

Use [Fusion.SerializableDictionary](#) to store [Networked] dictionary properties initial value. If unchecked, the weaver will emit System.Generic.Dictionary instead - a type that's not Unity-serializable, but custom serializers (e.g. Odin) may support it.

6.229.4.25 Version

```
int Version = CurrentVersion
```

Current [NetworkProjectConfig](#) version

6.229.5 Property Documentation

6.229.5.1 Global

```
NetworkProjectConfig Global [static], [get]
```

Reference for the default [NetworkProjectConfig](#). By default, loads a resource named "NetworkProjectConfig". This behaviour can be changed with an attribute [FusionGlobalScriptableObjectLoaderMethodAttribute](#).

6.230 NetworkProjectConfigAsset Class Reference

Manages and references the current instance of [NetworkProjectConfig](#)

Inherits [FusionGlobalScriptableObject< NetworkProjectConfigAsset >](#).

Classes

- struct [SerializableSimulationBehaviourMeta](#)

An auto-generated list containing meta information about all the [SimulationBehaviours](#) in the project, e.g. execution order.

Static Public Member Functions

- static bool [TryGetGlobal](#) (out [NetworkProjectConfigAsset](#) global)

Try to get the current [NetworkProjectConfig](#) instance.

- static void [UnloadGlobal](#) ()

Unload the current [NetworkProjectConfig](#) instance.

Public Attributes

- `SerializableSimulationBehaviourMeta[] BehaviourMeta = Array.Empty<SerializableSimulationBehaviourMeta>()`
An auto-generated list containing meta information about all the `SimulationBehaviours` in the project, e.g. execution order.
- `NetworkProjectConfig Config = new NetworkProjectConfig()`
The current `NetworkProjectConfig` instance.
- `NetworkPrefabTableOptions PrefabOptions = NetworkPrefabTableOptions.Default`
Options for the `NetworkPrefabTable`.
- `List< INetworkPrefabSource > Prefabs = new List<INetworkPrefabSource>()`
*An auto-generated list containing source information (e.g. Resource path, address, static reference) for all the prefabs that can be spawned, i.e. the ones with `NetworkObject` component and `NetworkObject.IsSpawnable` enabled.
Additional prefabs can be registered at runtime with `NetworkPrefabTable.TryAddSource`.*

Protected Member Functions

- `override void OnDisable ()`
Unloads all prefabs.

Properties

- `static NetworkProjectConfigAsset Global [get]`
The current `NetworkProjectConfig` instance.
- `static bool IsGlobalLoaded [get]`
True if the `NetworkProjectConfig` instance exists, otherwise false.

Additional Inherited Members

6.230.1 Detailed Description

Manages and references the current instance of `NetworkProjectConfig`

6.230.2 Member Function Documentation

6.230.2.1 OnDisable()

```
override void OnDisable ( ) [protected], [virtual]
```

Unloads all prefabs.

Reimplemented from `FusionGlobalScriptableObject< NetworkProjectConfigAsset >`.

6.230.2.2 TryGetGlobal()

```
static bool TryGetGlobal (
    out NetworkProjectConfigAsset global ) [static]
```

Try to get the current `NetworkProjectConfig` instance.

Parameters

<code>global</code>	<code>NetworkProjectConfig</code> instance if it exists, otherwise null.
---------------------	--

Returns

True if the `NetworkProjectConfig` instance exists, otherwise false.

6.230.2.3 UnloadGlobal()

```
static void UnloadGlobal ( ) [static]
```

Unload the current `NetworkProjectConfig` instance.

6.230.3 Member Data Documentation

6.230.3.1 BehaviourMeta

```
SerializableSimulationBehaviourMeta [ ] BehaviourMeta = Array.Empty<SerializableSimulationBehaviourMeta>()
```

An auto-generated list containing meta information about all the `SimulationBehaviour`s in the project, e.g. execution order.

6.230.3.2 Config

```
NetworkProjectConfig Config = new NetworkProjectConfig()
```

The current `NetworkProjectConfig` instance.

6.230.3.3 PrefabOptions

```
NetworkPrefabTableOptions PrefabOptions = NetworkPrefabTableOptions.Default
```

Options for the `NetworkPrefabTable`.

6.230.3.4 Prefabs

```
List<INetworkPrefabSource> Prefabs = new List<INetworkPrefabSource>()
```

An auto-generated list containing source information (e.g. Resource path, address, static reference) for all the prefabs that can be spawned, i.e. the ones with [NetworkObject](#) component and [NetworkObject.IsSpawnable](#) enabled. Additional prefabs can be registered at runtime with [NetworkPrefabTable.TryAddSource](#).

6.230.4 Property Documentation

6.230.4.1 Global

```
NetworkProjectConfigAsset Global [static], [get]
```

The current [NetworkProjectConfig](#) instance.

6.230.4.2 IsGlobalLoaded

```
bool IsGlobalLoaded [static], [get]
```

True if the [NetworkProjectConfig](#) instance exists, otherwise false.

6.231 NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta Struct Reference

An auto-generated list containing meta information about all the [SimulationBehaviour](#)s in the project, e.g. execution order.

Public Attributes

- int [ExecutionOrder](#)
The execution order of the [SimulationBehaviour](#).
- [SerializableType< SimulationBehaviour > Type](#)
The type of the [SimulationBehaviour](#).

6.231.1 Detailed Description

An auto-generated list containing meta information about all the [SimulationBehaviour](#)s in the project, e.g. execution order.

6.231.2 Member Data Documentation

6.231.2.1 ExecutionOrder

`int ExecutionOrder`

The execution order of the [SimulationBehaviour](#).

6.231.2.2 Type

`SerializableType<SimulationBehaviour> Type`

The type of the [SimulationBehaviour](#).

6.232 NetworkRNG Struct Reference

PCG32 random generator, 16 bytes in size. <http://www.pcg-random.org>

Inherits [INetworkStruct](#).

Public Member Functions

- [NetworkRNG \(Int32 seed\)](#)

Creates a new instance of [NetworkRNG](#) with a random seed.
- [double Next \(\)](#)

Generates a random double value within the inclusive range [0, 1].
- [double NextExclusive \(\)](#)

Generates a random double value within the exclusive range [0, 1).
- [int NextInt32 \(\)](#)

Generates a random integer value within the range of int.MinValue to int.MaxValue.
- [float NextSingle \(\)](#)

Generates a random float value within the inclusive range [0, 1].
- [float NextSingleExclusive \(\)](#)

Generates a random float value within the exclusive range [0, 1).
- [uint NextUInt32 \(\)](#)

Generates a random unsigned integer value within the range of 0 to uint.MaxValue.
- [Int32 RangeExclusive \(Int32 minInclusive, Int32 maxExclusive\)](#)

Returns a random Int32 within [minInclusive, maxExclusive) (range is exclusive). If minInclusive and maxExclusive are equal, then the "exclusive rule" is ignored and minInclusive will be returned. If minInclusive is greater than maxExclusive, then the numbers are automatically swapped.
- [UInt32 RangeExclusive \(UInt32 minInclusive, UInt32 maxExclusive\)](#)

Returns a random UInt32 within [minInclusive, maxExclusive) (range is exclusive). If minInclusive and maxExclusive are equal, then the "exclusive rule" is ignored and minInclusive will be returned. If minInclusive is greater than maxExclusive, then the numbers are automatically swapped.

- Double [RangeInclusive](#) (Double minInclusive, Double maxInclusive)
Returns a random Double within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.
- Int32 [RangeInclusive](#) (Int32 minInclusive, Int32 maxInclusive)
Returns a random Int32 within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.
- Single [RangeInclusive](#) (Single minInclusive, Single maxInclusive)
Returns a random Single within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.
- UInt32 [RangeInclusive](#) (UInt32 minInclusive, UInt32 maxInclusive)
Returns a random UInt32 within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.
- override string [ToString](#) ()
String representation of the RNG state.

Static Public Attributes

- const UInt32 **MAX** = UInt32.MaxValue
Maximum allowed value
- const int **SIZE** = 16
Size of the struct in bytes.

Properties

- [NetworkRNG Peek](#) [get]
Returns the same RNG instance.

6.232.1 Detailed Description

PCG32 random generator, 16 bytes in size. <http://www.pcg-random.org>

6.232.2 Constructor & Destructor Documentation

6.232.2.1 NetworkRNG()

```
NetworkRNG (
    Int32 seed )
```

Creates a new instance of [NetworkRNG](#) with a random seed.

Parameters

<code>seed</code>	Seed value.
-------------------	-------------

6.232.3 Member Function Documentation

6.232.3.1 Next()

```
double Next ( )
```

Generates a random double value within the inclusive range [0, 1].

Returns

A random double value between 0 and 1, inclusive.

6.232.3.2 NextExclusive()

```
double NextExclusive ( )
```

Generates a random double value within the exclusive range [0, 1).

Returns

A random double value between 0 (inclusive) and 1 (exclusive).

6.232.3.3 NextInt32()

```
int NextInt32 ( )
```

Generates a random integer value within the range of int.MinValue to int.MaxValue.

Returns

A random integer value between int.MinValue and int.MaxValue, inclusive.

6.232.3.4 NextSingle()

```
float NextSingle ( )
```

Generates a random float value within the inclusive range [0, 1].

Returns

A random float value between 0 and 1, inclusive.

6.232.3.5 NextSingleExclusive()

```
float NextSingleExclusive ( )
```

Generates a random float value within the exclusive range [0, 1).

Returns

A random float value between 0 (inclusive) and 1 (exclusive).

6.232.3.6 NextUInt32()

```
uint NextUInt32 ( )
```

Generates a random unsigned integer value within the range of 0 to uint.MaxValue.

Returns

A random unsigned integer value between 0 and uint.MaxValue, inclusive.

6.232.3.7 RangeExclusive() [1/2]

```
Int32 RangeExclusive (
    Int32 minInclusive,
    Int32 maxExclusive )
```

Returns a random Int32 within [minInclusive, maxExclusive) (range is exclusive). If minInclusive and maxExclusive are equal, then the "exclusive rule" is ignored and minInclusive will be returned. If minInclusive is greater than maxExclusive, then the numbers are automatically swapped.

6.232.3.8 RangeExclusive() [2/2]

```
UInt32 RangeExclusive (
    UInt32 minInclusive,
    UInt32 maxExclusive )
```

Returns a random UInt32 within [minInclusive, maxExclusive) (range is exclusive). If minInclusive and maxExclusive are equal, then the "exclusive rule" is ignored and minInclusive will be returned. If minInclusive is greater than maxExclusive, then the numbers are automatically swapped.

6.232.3.9 RangeInclusive() [1/4]

```
Double RangeInclusive (
    Double minInclusive,
    Double maxInclusive )
```

Returns a random Double within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.

6.232.3.10 RangeInclusive() [2/4]

```
Int32 RangeInclusive (
    Int32 minInclusive,
    Int32 maxInclusive )
```

Returns a random Int32 within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.

6.232.3.11 RangeInclusive() [3/4]

```
Single RangeInclusive (
    Single minInclusive,
    Single maxInclusive )
```

Returns a random Single within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.

6.232.3.12 RangeInclusive() [4/4]

```
UInt32 RangeInclusive (
    UInt32 minInclusive,
    UInt32 maxInclusive )
```

Returns a random UInt32 within [minInclusive, maxInclusive] (range is inclusive). If minInclusive is greater than maxInclusive, then the numbers are automatically swapped.

6.232.3.13 ToString()

```
override string ToString ( )
```

String representation of the RNG state.

6.232.4 Member Data Documentation

6.232.4.1 MAX

```
const UInt32 MAX = UInt32.MaxValue [static]
```

Maximum allowed value

6.232.4.2 SIZE

```
const int SIZE = 16 [static]
```

Size of the struct in bytes.

6.232.5 Property Documentation

6.232.5.1 Peek

```
NetworkRNG Peek [get]
```

Returns the same RNG instance.

6.233 NetworkRpcStaticWeavedInvokerAttribute Class Reference

Network Rpc Static Weaved Invoker Attribute Contains info about a static weaved RPC Method

Inherits Attribute.

Public Member Functions

- [NetworkRpcStaticWeavedInvokerAttribute \(string key\)](#)
NetworkRpcStaticWeavedInvokerAttribute Constructor

Properties

- string [Key](#) [get]
RPC Key

6.233.1 Detailed Description

Network Rpc Static Weaved Invoker Attribute Contains info about a static weaved RPC Method

6.233.2 Constructor & Destructor Documentation

6.233.2.1 NetworkRpcStaticWeavedInvokerAttribute()

```
NetworkRpcStaticWeavedInvokerAttribute (
    string key )
```

NetworkRpcStaticWeavedInvokerAttribute Constructor

Parameters

key	Key
-----	-----

6.233.3 Property Documentation

6.233.3.1 Key

```
string Key [get]
```

RPC Key

6.234 NetworkRpcWeavedInvokerAttribute Class Reference

Network Rpc Weaved Invoker Attribute Contains info about a weaved RPC Method

Inherits Attribute.

Public Member Functions

- *NetworkRpcWeavedInvokerAttribute* (int key, int sources, int targets)
NetworkRpcWeavedInvokerAttribute Constructor

Properties

- int **Key** [get]
RPC Key
- int **Sources** [get]
RPC Sources
- int **Targets** [get]
RPC Targets

6.234.1 Detailed Description

Network Rpc Weaved Invoker Attribute Contains info about a weaved RPC Method

6.234.2 Constructor & Destructor Documentation

6.234.2.1 NetworkRpcWeavedInvokerAttribute()

```
NetworkRpcWeavedInvokerAttribute (
    int key,
    int sources,
    int targets )
```

[NetworkRpcWeavedInvokerAttribute](#) Constructor

Parameters

<i>key</i>	Key
<i>sources</i>	Sources
<i>targets</i>	Targets

6.234.3 Property Documentation

6.234.3.1 Key

```
int Key [get]
```

RPC Key

6.234.3.2 Sources

`int Sources [get]`

RPC Sources

6.234.3.3 Targets

`int Targets [get]`

RPC Targets

6.235 NetworkRunner Class Reference

Host Migration related code in order to get a copy of the [Simulation](#) State

Inherits [Behaviour](#), and [Simulation.ICallbacks](#).

Public Types

- enum class [BuildTypes](#)
Enumeration of Fusion.Runtime.dll options.
- enum class [States](#)
Initialization stages of Fusion

Public Member Functions

- void [AddCallbacks](#) (params [INetworkRunnerCallbacks](#)[] callbacks)
Register an [INetworkRunnerCallbacks](#) instance for callbacks from this [NetworkRunner](#).
- void [AddGlobal](#) ([SimulationBehaviour](#) instance)
Add and register a [SimulationBehaviour](#) to this [NetworkRunner](#). Note: It should NOT be a [NetworkBehaviour](#)
- void [AddPlayerAreaOfInterest](#) ([PlayerRef](#) player, [Vector3](#) center, float radius)
Call this every FixedUpdateNetwork to add an area of interest for a player. Should only be called from the Host/Server in Server client mode. Should only be called for the local player in shared mode.
- void [Attach](#) ([NetworkObject](#) networkObject, [PlayerRef?](#) inputAuthority=null, bool allocate=true, bool? masterClientObjectOverride=null)
Attaches a user-created network object to the network.
- void [Attach](#) ([NetworkObject](#)[] networkObjects, [PlayerRef?](#) inputAuthority=null, bool allocate=true, bool? masterClientObjectOverride=null)
Attach and assign to this [NetworkRunner](#) the [NetworkObject](#) provided. Used internally from the default implementation of [INetworkSceneManager](#) to register scene objects.
- void [ClearPlayerAreaOfInterest](#) ([PlayerRef](#) player)
Clears the area of interest for a player. This can only be called from the server/host
- void [Despawn](#) ([NetworkObject](#) networkObject)
Destroys a [NetworkObject](#).
- void [DestroySingleton](#)< T > ()

- Removes a specific [SimulationBehaviour](#) from this [NetworkRunner](#) gameobject, if it exists.*
- void [Disconnect \(PlayerRef player, byte\[\] token=null\)](#)
Disconnects a player from the server.
 - bool [EnsureRunnerScenesActive \(out Scene previousActiveScene\)](#)
Ensures the scene of this runner is active and returns the previous active scene
 - bool [Exists \(NetworkId id\)](#)
Returns if the [Fusion.Simulation](#) contains a [NetworkObject](#) with given id in the current State.
 - bool [Exists \(NetworkObject obj\)](#)
Returns if the [Fusion.Simulation](#) contains a reference to a [NetworkObject](#) in the current State.
 - [NetworkObject FindObject \(NetworkId networkId\)](#)
Get the [NetworkObject](#) instance for this [NetworkRunner](#) from a [NetworkId](#).
 - [SimulationBehaviour\[\] GetAllBehaviours \(Type type\)](#)
Returns an array of all [SimulationBehaviour](#) instances registered with this [NetworkRunner](#).
 - List< T > [GetAllBehaviours< T > \(\)](#)
Get a list with all behaviours of the desired type that are registered on the [NetworkRunner](#).
 - void [GetAllBehaviours< T > \(List< T > result\)](#)
Add on the list all behaviours of the desired type that are registered on the [NetworkRunner](#). Note: The list will not be cleared before adding the results.
 - List< [NetworkObject](#) > [GetAllNetworkObjects \(\)](#)
Retrieves a list of all network objects in the simulation.
 - void [GetAreaOfInterestGizmoData \(List<\(Vector3 center, Vector3 size, int playerCount, int objectCount\)> result\)](#)
Populates the provided list with data about the current Area of Interest (AOI) cells. Each element in the list represents one AOI cell.
 - T? [GetInputForPlayer< T > \(PlayerRef player\)](#)
Returns the [NetworkInput](#) data from player, converted to the indicated [INetworkInput](#).
 - [SimulationBehaviourListScope GetInterfaceListHead \(Type type, int index, out SimulationBehaviour head\)](#)
Get the interface list head.
 - [SimulationBehaviour GetInterfaceListNext \(SimulationBehaviour behaviour\)](#)
Get the next behaviour
 - [SimulationBehaviour GetInterfaceListPrev \(SimulationBehaviour behaviour\)](#)
Get the previous behaviour
 - int [GetInterfaceListsCount \(Type type\)](#)
Get the number of interfaces of the desired type that are registered on the behaviour updater.
 - void [GetMemorySnapshot \(MemoryStatisticsSnapshot.TargetAllocator targetAllocator, ref MemoryStatisticsSnapshot snapshot\)](#)
Gets a memory snapshot of the simulation.
 - List< NetworkId > [GetObjectsInAreaOfInterestForPlayer \(PlayerRef player\)](#)
Retrieves a list of network object IDs that are in the area of interest for the specified player. Server only.
 - PhysicsScene [GetPhysicsScene \(\)](#)
Get the 3D Physics scene being used by this Runner.
 - PhysicsScene2D [GetPhysicsScene2D \(\)](#)
Get the 2D Physics scene being used by this Runner.
 - int? [GetPlayerActorId \(PlayerRef player\)](#)
Gets Player's Actor Number (ID).
 - byte[] [GetPlayerConnectionToken \(PlayerRef player=default\)](#)
*Returns a copy of the Connection Token used by a Player when connecting to this Server. Only available on Server.
It will return null if running on a Client or the Connection token is missing*
 - [ConnectionType GetPlayerConnectionType \(PlayerRef player\)](#)
Return the [ConnectionType](#) with a Remote [PlayerRef](#). Valid only when invoked from a Server ([NetworkRunner.IsServer](#))
 - [NetworkObject GetPlayerObject \(PlayerRef player\)](#)

- Gets the network object associated with a specific player
 - double **GetPlayerRtt** (**PlayerRef** playerRef)

Returns the player round trip time (ping) in seconds
 - string **GetPlayerUserId** (**PlayerRef** player=default)

Gets Player's UserID.
 - **NetworkInput?** **GetRawInputForPlayer** (**PlayerRef** player)

Returns the unconverted unsafe **NetworkInput** for the indicated player.
 - IEnumerable<**NetworkObject**> **GetResumeSnapshotNetworkObjects** ()

Iterate over the old **NetworkObjects** from the Resume Snapshot
 - IEnumerable<(**NetworkObject**, **NetworkObjectHeaderPtr**)> **GetResumeSnapshotNetworkSceneObjects** ()

Iterate over the Scene **NetworkObjects** from the Resume Snapshot while giving the reference of the old Snapshot data associated with that particular Scene Object
 - **RpcTargetStatus** **GetRpcTargetStatus** (**PlayerRef** target)

Return the **RpcTargetStatus** for a specific player.
 - **SceneRef** **GetSceneRef** (**GameObject** gameObj)
 - **SceneRef** **GetSceneRef** (string sceneNameOrPath)
 - T **GetSingleton**< T > ()

Ensures that a specific **SimulationBehaviour** component exists on this **NetworkRunner** gameobject.
 - bool **HasAnyActiveConnections** ()
 - bool **HasSingleton**< T > ()

Returns if a given **SimulationBehaviour** is present in this **NetworkRunner** gameobject.
 - **GameObject** **InstantiateInRunnerScene** (**GameObject** original)

Instantiates an object in the scene of this runner
 - **GameObject** **InstantiateInRunnerScene** (**GameObject** original, Vector3 position, Quaternion rotation)

Instantiates an object in the scene of this runner
 - T **InstantiateInRunnerScene**< T > (T original)

Instantiates an object in the scene of this runner
 - T **InstantiateInRunnerScene**< T > (T original, Vector3 position, Quaternion rotation)

Instantiates an object in the scene of this runner
 - void **InvokeSceneLoadDone** (in **SceneLoadDoneArgs** info)

Invoke **INetworkRunnerCallbacks.OnSceneLoadDone(NetworkRunner)** on all implementations
 - void **InvokeSceneLoadStart** (**SceneRef** sceneRef)

Invoke **INetworkRunnerCallbacks.OnSceneLoadStart(NetworkRunner)** on all implementations
 - bool? **IsInterestedIn** (**NetworkObject** obj, **PlayerRef** player)

Test if a player has Interest in a **NetworkObject**.
 - bool **IsValid** (**PlayerRef** player)

Checks if the provided player is valid in the current simulation.
 - async Task<**StartGameResult**> **JoinSessionLobby** (**SessionLobby** sessionLobby, string lobbyID=null, **AuthenticationValues** authentication=null, FusionAppSettings customAppSettings=null, bool? useDefaultCloudPorts=false, CancellationToken cancellationToken=default, bool useCachedRegions=true)

Join the Peer to a specific Lobby, either a prebuild or a custom one.
 - **NetworkSceneAsyncOp** **LoadScene** (**SceneRef** sceneRef, LoadSceneMode loadSceneMode=LoadSceneMode.Single, LocalPhysicsMode localPhysicsMode=LocalPhysicsMode.None, bool setActiveOnLoad=DefaultSetActiveOnLoad)

Loads a scene
 - **NetworkSceneAsyncOp** **LoadScene** (**SceneRef** sceneRef, LoadSceneParameters parameters, bool setActiveOnLoad=DefaultSetActiveOnLoad)
 - **NetworkSceneAsyncOp** **LoadScene** (string sceneName, LoadSceneMode loadSceneMode=LoadSceneMode.Single, LocalPhysicsMode localPhysicsMode=LocalPhysicsMode.None, bool setActiveOnLoad=DefaultSetActiveOnLoad)

Loads a scene

- **NetworkSceneAsyncOp LoadScene** (string sceneName, LoadSceneParameters parameters, bool setActiveOnLoad=DefaultSetActiveOnLoad)
Loads a scene
- void **MakeDontDestroyOnLoad** (GameObject obj)
Mark an object as DontDestroyOnLoad.
- bool **MoveGameObjectToSameScene** (GameObject gameObj, GameObject other)
Moves a GameObject to the same scene as another GameObject
- bool **MoveGameObjectToScene** (GameObject gameObj, SceneRef sceneRef)
Moves a GameObject to a specific scene
- void **MoveToRunnerScene** (GameObject instance, SceneRef? targetSceneRef=null)
Moves an object to the scene of this runner
- void **MoveToRunnerScene< T >** (T component)
Moves an object to the scene of this runner
- delegate void **ObjectDelegate** (NetworkRunner runner, NetworkObject obj)
Delegate type for object callback
- delegate void **OnBeforeSpawned** (NetworkRunner runner, NetworkObject obj)
Delegate type for on before spawned callback
- async Task< bool > **PushHostMigrationSnapshot** ()
Compute and send a Host Migration Snapshot to the Photon Cloud
- int **RegisterSceneObjects** (SceneRef scene, NetworkObject[] objects, NetworkSceneLoadId loadId=default)
Registers scene objects to the network.
- void **RemoveCallbacks** (params INetworkRunnerCallbacks[] callbacks)
Unregister an INetworkRunnerCallbacks instance for callbacks from this NetworkRunner.
- void **RemoveGlobal** (SimulationBehaviour instance)
Removes a specific SimulationBehaviour from this NetworkObject gameobject, if it exists.
- void **RenderInternal** ()
This method is meant to be called by INetworkRunnerUpdater.
- void **SendReliableDataToPlayer** (PlayerRef player, ReliableKey key, byte[] data)
Sends a reliable data buffer to a target player.
- void **SendReliableDataToServer** (ReliableKey key, byte[] data)
Sends a reliable data buffer to the server.
- void **SendRpc** (SimulationMessage *message)
Sends RPC message. Not meant to be used directly, ILWeaver calls this.
- void **SendRpc** (SimulationMessage *message, out RpcSendResult info)
Sends RPC message. Not meant to be used directly, ILWeaver calls this.
- void **SetAreaOfInterestCellSize** (int size)
Set the area of interest cell size
- void **SetAreaOfInterestGrid** (int x, int y, int z)
Set the area of interest grid dimensions
- void **SetBehaviourReplicateTo** (NetworkBehaviour behaviour, PlayerRef player, bool replicate)
Controls if a specific network behaviours state is replicated to a player or not
- void **SetBehaviourReplicateToAll** (NetworkBehaviour behaviour, bool replicate)
Controls if a specific network behaviours state is replicated to all players or not
- bool **SetIsSimulated** (NetworkObject obj, bool simulate)
Sets the simulation state for this object, if it takes part in the NetworkFixedUpdate, etc.
- void **SetMasterClient** (PlayerRef player)
Promote a player to be the new master client. Only the master client is able to call this method
- void **SetPlayerAlwaysInterested** (PlayerRef player, NetworkObject networkObject, bool alwaysInterested)
Flags this player as always interested in this object. Means it does not have to be in a players area of interest to be replicated. Only the NetworkObject State Authority can set interest.
- void **SetPlayerObject** (PlayerRef player, NetworkObject networkObject)

- **Sets the network object associated with this player**
- void **SetSimulateMultiPeerPhysics** (bool value)

Set the value for simulating physics scenes when using multi-peer. Restores the default physics simulation settings if false, overrides the default if true. (2D: SimulationMode2D.Script | 3D: AutoSimulation = false)
- Task **Shutdown** (bool destroyGameObject=true, **ShutdownReason** shutdownReason=**ShutdownReason.Ok**, bool forceShutdownProcedure=false)

Initiates a Simulation.Dispose.
- void **SinglePlayerContinue** ()

Continues a paused game in single player
- void **SinglePlayerPause** ()

Pauses the game in single player
- void **SinglePlayerPause** (bool paused)

Sets the paused state in a single player
- NetworkObject **Spawn** (GameObject prefab, Vector3? position=null, Quaternion? rotation=null, **PlayerRef?** inputAuthority=null, **OnBeforeSpawned** onBeforeSpawned=null, **NetworkSpawnFlags** flags=default)

Attempts to network instantiate a NetworkObject using a GameObject. The supplied GameObject must have a NetworkObject component.
- NetworkObject **Spawn** (NetworkObject prefab, Vector3? position=null, Quaternion? rotation=null, **PlayerRef?** inputAuthority=null, **OnBeforeSpawned** onBeforeSpawned=null, **NetworkSpawnFlags** flags=default)

Attempts to network instantiate a NetworkObject using a NetworkObject prefab. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use NetworkTransform (or any custom class derived from NetworkTRSP) to replicate the initial transform state.
- NetworkObject **Spawn** (NetworkObjectGuid prefabGuid, Vector3? position=null, Quaternion? rotation=null, **PlayerRef?** inputAuthority=null, **OnBeforeSpawned** onBeforeSpawned=null, **NetworkSpawnFlags** flags=default)

Attempts to network instantiate a NetworkObject using a NetworkObjectGuid Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use NetworkTransform (or any custom class derived from NetworkTRSP) to replicate the initial transform state.
- NetworkObject **Spawn** (NetworkPrefabId typeId, Vector3? position=null, Quaternion? rotation=null, **PlayerRef?** inputAuthority=null, **OnBeforeSpawned** onBeforeSpawned=null, **NetworkSpawnFlags** flags=default)

Attempts to network instantiate a NetworkObject using a NetworkPrefabId Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use NetworkTransform (or any custom class derived from NetworkTRSP) to replicate the initial transform state.
- NetworkObject **Spawn** (NetworkPrefabRef prefabRef, Vector3? position=null, Quaternion? rotation=null, **PlayerRef?** inputAuthority=null, **OnBeforeSpawned** onBeforeSpawned=null, **NetworkSpawnFlags** flags=default)

Attempts to network instantiate a NetworkObject using a NetworkPrefabRef. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use NetworkTransform (or any custom class derived from NetworkTRSP) to replicate the initial transform state.
- T **Spawn< T >** (T prefab, Vector3? position=null, Quaternion? rotation=null, **PlayerRef?** inputAuthority=null, **OnBeforeSpawned** onBeforeSpawned=null, **NetworkSpawnFlags** flags=default)

Attempts to network instantiate a NetworkObject using a Component type that is part of a NetworkObject
- NetworkSpawnOp **SpawnAsync** (GameObject prefab, Vector3? position=null, Quaternion? rotation=null, **PlayerRef?** inputAuthority=null, **OnBeforeSpawned** onBeforeSpawned=null, **NetworkSpawnFlags** flags=default, **NetworkObjectSpawnDelegate** onCompleted=null)

Attempts to network instantiate a NetworkObject using a GameObject. The supplied GameObject must have a NetworkObject component.
- NetworkSpawnOp **SpawnAsync** (NetworkObject prefab, Vector3? position=null, Quaternion? rotation=null, **PlayerRef?** inputAuthority=null, **OnBeforeSpawned** onBeforeSpawned=null, **NetworkSpawnFlags** flags=default, **NetworkObjectSpawnDelegate** onCompleted=null)

Attempts to network instantiate a NetworkObject using a NetworkObject prefab. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use NetworkTransform (or any custom class derived from NetworkTRSP) to replicate the initial transform state.

- **NetworkSpawnOp SpawnAsync** (`NetworkObjectGuid` prefabGuid, `Vector3?` position=null, `Quaternion?` rotation=null, `PlayerRef?` inputAuthority=null, `OnBeforeSpawned` onBeforeSpawned=null, `NetworkSpawnFlags` flags=default, `NetworkObjectSpawnDelegate` onCompleted=null)
- Attempts to network instantiate a `NetworkObject` using a `NetworkObjectGuid`. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use `NetworkTransform` (or any custom class derived from `NetworkTRSP`) to replicate the initial transform state.*
- **NetworkSpawnOp SpawnAsync** (`NetworkPrefabId` typeId, `Vector3?` position=null, `Quaternion?` rotation=null, `PlayerRef?` inputAuthority=null, `OnBeforeSpawned` onBeforeSpawned=null, `NetworkSpawnFlags` flags=default, `NetworkObjectSpawnDelegate` onCompleted=null)
- Attempts to network instantiate a `NetworkObject` using a `NetworkPrefabId`. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use `NetworkTransform` (or any custom class derived from `NetworkTRSP`) to replicate the initial transform state.*
- **NetworkSpawnOp SpawnAsync** (`NetworkPrefabRef` prefabRef, `Vector3?` position=null, `Quaternion?` rotation=null, `PlayerRef?` inputAuthority=null, `OnBeforeSpawned` onBeforeSpawned=null, `NetworkSpawnFlags` flags=default, `NetworkObjectSpawnDelegate` onCompleted=null)
- Attempts to network instantiate a `NetworkObject` using a `NetworkPrefabRef`. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use `NetworkTransform` (or any custom class derived from `NetworkTRSP`) to replicate the initial transform state.*
- **NetworkSpawnOp SpawnAsync< T >** (`T` prefab, `Vector3?` position=null, `Quaternion?` rotation=null, `PlayerRef?` inputAuthority=null, `OnBeforeSpawned` onBeforeSpawned=null, `NetworkSpawnFlags` flags=default, `NetworkObjectSpawnDelegate` onCompleted=null)
- Attempts to network instantiate a `NetworkObject` using a Component type that is part of a `NetworkObject`.*
- **Task< StartGameResult > StartGame** (`StartGameArgs` args)
- Starts the local Fusion Runner and takes care of all major setup necessary*
- **bool TryFindBehaviour** (`NetworkBehaviourId` behaviourId, out `NetworkBehaviour` behaviour)
- Get the `NetworkBehaviour` instance for this `NetworkRunner` from a `NetworkBehaviourId`.*
- **bool TryFindBehaviour< T >** (`NetworkBehaviourId` id, out `T` behaviour)
- Try to find a `NetworkBehaviour` with the provided `NetworkBehaviourId`.*
- **bool TryFindObject** (`NetworkId` objectId, out `NetworkObject` networkObject)
- Get the `NetworkObject` instance for this `NetworkRunner` from a `NetworkId`.*
- **bool TryGetBehaviourStatistics** (`Type` behaviourType, out `BehaviourStatisticsSnapshot` behaviourStatistics↔ Snapshot)
- Tries to get the statistics snapshot for a specified behaviour type.*
- **bool TryGetFusionStatistics** (out `FusionStatisticsManager` statisticsManager)
- Tries to get the `FusionStatisticsManager` from the simulation.*
- **bool TryGetInputForPlayer< T >** (`PlayerRef` player, out `T` input)
- Outputs the `NetworkInput` from player, translated to the indicated `INetworkInput`.*
- **T TryGetNetworkedBehaviourFromNetworkedObjectRef< T >** (`NetworkId` networkId)
- Tries to return the first instance of `T` found on the root of a `NetworkObject`.*
- **NetworkBehaviourId TryGetNetworkedBehaviourId** (`NetworkBehaviour` behaviour)
- Tries to return a `NetworkBehaviourId` for the `NetworkBehaviour` provided.*
- **NetworkId TryGetObjectRefFromNetworkedBehaviour** (`NetworkBehaviour` behaviour)
- Tries to return the behaviour `NetworkId`.*
- **bool TryGetPhysicsInfo** (out `NetworkPhysicsInfo` info)
- Try to get the physics info.*
- **bool TryGetPlayerObject** (`PlayerRef` player, out `NetworkObject` networkObject)
- Try to gets the `NetworkObject` associated with a specific player*
- **bool TryGetSceneInfo** (out `NetworkSceneInfo` sceneInfo)
- Tries to get the `NetworkSceneInfo` of this `NetworkRunner`.*
- **bool TrySetPhysicsInfo** (`NetworkPhysicsInfo` info)
- Try to set the physics info.*
- **NetworkSpawnStatus TrySpawn** (`GameObject` prefab, out `NetworkObject` obj, `Vector3?` position=null, `Quaternion?` rotation=null, `PlayerRef?` inputAuthority=null, `OnBeforeSpawned` onBeforeSpawned=null, `NetworkSpawnFlags` flags=default)

Attempts to network instantiate a [NetworkObject](#) using a [GameObject](#). The supplied [GameObject](#) must have a [NetworkObject](#) component.

- [NetworkSpawnStatus TrySpawn](#) ([NetworkObject](#) prefab, out [NetworkObject](#) obj, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default)

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObject](#) prefab. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

- [NetworkSpawnStatus TrySpawn](#) ([NetworkObjectGuid](#) prefabGuid, out [NetworkObject](#) obj, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default)

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObjectGuid](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

- [NetworkSpawnStatus TrySpawn](#) ([NetworkPrefabId](#) typeId, out [NetworkObject](#) obj, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default)

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabId](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

- [NetworkSpawnStatus TrySpawn](#) ([NetworkPrefabRef](#) prefabRef, out [NetworkObject](#) obj, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default)

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabRef](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

- [NetworkSpawnStatus TrySpawn< T >](#) ([T](#) prefab, out [T](#) obj, [Vector3?](#) position=null, [Quaternion?](#) rotation=null, [PlayerRef?](#) inputAuthority=null, [OnBeforeSpawned](#) onBeforeSpawned=null, [NetworkSpawnFlags](#) flags=default)

Attempts to network instantiate a [NetworkObject](#) using a Component type that is part of a [NetworkObject](#).

- [NetworkSceneAsyncOp UnloadScene](#) ([SceneRef](#) sceneRef)
- [NetworkSceneAsyncOp UnloadScene](#) (string sceneName)

Unloads a scene

- void [UpdateInternal](#) (double dt)

This method is meant to be called by [INetworkRunnerUpdater](#).

Static Public Member Functions

- static [Task< List< RegionInfo > >](#) [GetAvailableRegions](#) (string appId=default, [CancellationToken](#) cancellationToken=default)

Starts an operation to retrieves the list of available regions.

- static [List< NetworkRunner >.Enumerator](#) [GetInstanceEnumerator](#) ()

Get enumerator for the collection of all [NetworkRunners](#). Allows to enumerate alloc-free.

- static [NetworkRunner](#) [GetRunnerForGameObject](#) ([GameObject](#) gameObject)

Get the [NetworkRunner](#) a [GameObject](#) instance belongs to.

- static [NetworkRunner](#) [GetRunnerForScene](#) ([Scene](#) scene)

Get the [NetworkRunner](#) from a specific Scene

Properties

- `IEnumerable< PlayerRef > ActivePlayers [get]`
Returns the collection of `PlayerRef` objects for this `NetworkRunner`'s `Fusion.Simulation`.
- `AuthenticationValues AuthenticationValues [get]`
`AuthenticationValues` used by this Runner to Authenticate the local peer.
- `static BuildTypes BuildType [get]`
Get `Fusion.Runtime.dll` build type.
- `bool CanSpawn [get]`
Signal if the Network Runner can spawn a `NetworkObject`
- `NetworkProjectConfig Config [get]`
Returns the `NetworkProjectConfig` reference.
- `ConnectionType CurrentConnectionType [get]`
Check the current Connection Type with the Remote Server
- `float DeltaTime [get]`
Returns the fixed tick time interval. Derived from the `SimulationRuntimeConfig.TickRate`.
- `GameMode GameMode [get]`
Current Game Mode active on the `Fusion Simulation`
- `static IReadOnlyList< NetworkRunner > Instances [get]`
A list of all `NetworkRunners`.
- `bool IsClient [get]`
Returns if this `Fusion.Simulation` represents a Client connection.
- `bool IsCloudReady [get]`
Signal if the Local Peer is connected to Photon Cloud and is able to Create/Join Room but also receive Lobby Updates
- `bool IsConnectedToServer [get]`
Returns if this Client is currently connected to a Remote Server
- `bool IsFirstTick [get]`
If this is the first tick that executes this update or re-simulation
- `bool IsForward [get]`
If this is not a re-simulation but a new forward tick
- `bool IsLastTick [get]`
If this is the last tick that is being executed this update
- `bool IsPlayer [get]`
Returns true if this runner represents a Client or Host. Dedicated servers have no local player and will return false.
- `bool IsResimulation [get]`
If we are currently executing a client side prediction re-simulation.
- `bool IsResume [get]`
If this instance is a resume (host migration)
- `bool IsRunning [get]`
Returns if this `Fusion.Simulation` is valid and running.
- `bool IsSceneAuthority [get]`
Is this runner responsible for scene management.
- `bool? IsSceneManagerBusy [get]`
Signals if the `INetworkSceneManager` instance assigned to this `NetworkRunner` is busy with any scene loading operation.
- `bool IsServer [get]`
Returns if this `Fusion.Simulation` represents a Server connection.
- `bool IsSharedModeMasterClient [get]`
Signal if the Local Peer is in a Room and is the Room Master Client
- `bool IsShutdown [get]`
If the runner is shutdown

- **bool IsSinglePlayer [get]**
Returns true if this runner was started as single player (Started as [SimulationModes.Host](#) with [SimulationConfig.PlayerCount = 1](#)).
- **bool IsStarting [get]**
If the runner is pending to start
- **HitboxManager LagCompensation [get]**
Returns the global instance of a lag compensation buffer [Fusion.HitboxManager](#).
- **Tick LatestServerTick [get]**
Get the latest confirmed tick of the server we are aware of
- **LobbyInfo LobbyInfo = new LobbyInfo() [get]**
Signal if the local peer is already inside a Lobby
- **float LocalAlpha [get]**
Get the local time alpha value
- **PlayerRef LocalPlayer [get]**
Returns a [PlayerRef](#) for the local simulation. For a dedicated server [PlayerRef.IsRealPlayer](#) will equal false. PlayerRefs are assigned in order from 0 to MaxPlayers-1 and are re-used as players join and leave. The only caveat is that the server player (if one exists), always gets the last index no matter how many clients are connected.
- **float??? LocalRenderTime [get]**
The current time ([current State.Time + Simulation.DeltaTime](#)) for predicted objects (objects in the local time frame). Use as an equivalent to Unity's Time.time. Time is relative to [Tick 0](#) (which represents Time 0f).
- **SimulationModes Mode [get]**
Returns the [SimulationModes](#) flags for The type of network peer the associated [Fusion.Simulation](#) represents.
- **NATType NATType [get]**
Exposes the current NAT Type from the local Peer
- **INetworkObjectProvider ObjectProvider [get]**
Returns the [INetworkObjectProvider](#) instance.
- **NetworkPrefabTable Prefabs [get]**
Reference to the [NetworkPrefabTable](#).
- **bool ProvideInput [get, set]**
Indicates if this [NetworkRunner](#) is collecting [PlayerRef INetworkInput](#).
- **int??? ReliableDataSendRate [get, set]**
- **float RemoteRenderTime [get]**
The current time ([current State.Time + Simulation.DeltaTime](#)) for non-predicted objects (objects in a remote time frame). Use as an equivalent to Unity's Time.time. Time is relative to [Tick 0](#) (which represents Time 0f).
- **INetworkSceneManager SceneManager [get]**
Returns the [INetworkSceneManager](#) instance.
- **SessionInfo SessionInfo = new SessionInfo() [get]**
Stores information about the current running session
- **float SimulationTime [get]**
The time the current State represents (the most recent FixedUpdateNetwork simulation). Use as an equivalent to Unity's Time.fixedTime. Time is relative to [Tick 0](#) (which represents Time 0f).
- **Scene??? SimulationUnityScene [get]**
The main scene of the [NetworkRunner](#) or default if not running.
- **SimulationStages Stage [get]**
Returns the current [SimulationStages](#) stage of this [Fusion.Simulation](#).
- **States State [get]**
The current state of the runner, if it's Starting, Running, Shutdown
- **Tick??? Tick [get]**
The tick associated with the current state of networked objects, or the current simulation tick being processed (when evaluated during [FixedUpdateNetwork](#)).
- **int TickRate [get]**
- **int TicksExecuted [get]**

Returns how many ticks we executed last update.

- [Topologies](#) [Topology](#) [get]

The current topology used

- string [UserId](#) [get]

Photon Client UserID

Events

- [ObjectDelegate](#) [ObjectAcquired](#)

Event for object acquired

6.235.1 Detailed Description

Host Migration related code in order to get a copy of the [Simulation](#) State

All Scene related API and fields

Represents a Server or Client [Simulation](#).

6.235.2 Member Enumeration Documentation

6.235.2.1 BuildTypes

enum [BuildTypes](#) [strong]

Enumeration of Fusion.Runtime.dll options.

Enumerator

Debug	Use the Debug version of the Fusion.Runtime.dll.
Release	Use the Debug version of the Fusion.Runtime.dll.

6.235.2.2 States

enum [States](#) [strong]

Initialization stages of [Fusion](#)

Enumerator

Starting	Runner is about to start
Running	Runner is running
Shutdown	Runner is shutdown

6.235.3 Member Function Documentation

6.235.3.1 AddCallbacks()

```
void AddCallbacks (
    params INetworkRunnerCallbacks[ ] callbacks )
```

Register an [INetworkRunnerCallbacks](#) instance for callbacks from this [NetworkRunner](#).

Parameters

<i>callbacks</i>	Callbacks to register
------------------	-----------------------

6.235.3.2 AddGlobal()

```
void AddGlobal (
    SimulationBehaviour instance )
```

Add and register a [SimulationBehaviour](#) to this [NetworkRunner](#). Note: It should NOT be a [NetworkBehaviour](#)

6.235.3.3 AddPlayerAreaOfInterest()

```
void AddPlayerAreaOfInterest (
    PlayerRef player,
    Vector3 center,
    float radius )
```

Call this every FixedUpdateNetwork to add an area of interest for a player. Should only be called from the Host/↔ Server in Server client mode. Should only be called for the local player in shared mode.

6.235.3.4 Attach() [1/2]

```
void Attach (
    NetworkObject networkObject,
    PlayerRef? inputAuthority = null,
    bool allocate = true,
    bool? masterClientObjectOverride = null )
```

Attaches a user-created network object to the network.

Parameters

<i>networkObject</i>	The network object to attach. Must not be null and must have a valid NetworkTypeid.
<i>inputAuthority</i>	Optional PlayerRef . If assigned, it will be the default input authority for this object.
<i>allocate</i>	Optional boolean. If true, the object will be allocated in memory and attached to the scene object. Default is true.
<i>masterClientObjectOverride</i>	Optional boolean. If provided, it will override the master client object setting. Default is null.

Exceptions

ArgumentNullException	Thrown when the provided network object is null.
ArgumentException	Thrown when the provided network object has an invalid NetworkTypeid.

6.235.3.5 Attach() [2/2]

```
void Attach (
    NetworkObject[ ] networkObjects,
    PlayerRef? inputAuthority = null,
    bool allocate = true,
    bool? masterClientObjectOverride = null )
```

Attach and assign to this [NetworkRunner](#) the [NetworkObject](#) provided. Used internally from the default implementation of [INetworkSceneManager](#) to register scene objects.

6.235.3.6 ClearPlayerAreaOfInterest()

```
void ClearPlayerAreaOfInterest (
    PlayerRef player )
```

Clears the area of interest for a player. This can only be called from the server/host

6.235.3.7 Despawn()

```
void Despawn (
    NetworkObject networkObject )
```

Destroys a [NetworkObject](#).

Parameters

<i>networkObject</i>	The NetworkObject to be destroyed.
----------------------	--

This method checks if the local simulation has state authority over the [NetworkObject](#). If it does, it checks if the [NetworkObject](#) exists and has state authority. If these conditions are met, it destroys the [NetworkObject](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the NetworkObject does not belong to this runner.
----------------------------------	---

6.235.3.8 DestroySingleton< T >()

```
void DestroySingleton< T > ( )
```

Removes a specific [SimulationBehaviour](#) from this [NetworkRunner](#) gameobject, if it exists.

Type Constraints

T : SimulationBehaviour

6.235.3.9 Disconnect()

```
void Disconnect (
    PlayerRef player,
    byte[] token = null )
```

Disconnects a player from the server.

Parameters

<i>player</i>	The player to disconnect. Must be a valid PlayerRef .
<i>token</i>	Optional byte array. If provided, it will be used as the disconnection token.

This method can only be called from the server. If called from a client, an error message will be logged.

6.235.3.10 EnsureRunnerScenesActive()

```
bool EnsureRunnerSceneIsActive (
    out Scene previousActiveScene )
```

Ensures the scene of this runner is active and returns the previous active scene

Parameters

<i>previousActiveScene</i>	Previous active scene
----------------------------	-----------------------

Returns

True if the scene was changed, false otherwise

6.235.3.11 Exists() [1/2]

```
bool Exists (
    NetworkId id )
```

Returns if the [Fusion.Simulation](#) contains a [NetworkObject](#) with given *id* in the current State.

6.235.3.12 Exists() [2/2]

```
bool Exists (
    NetworkObject obj )
```

Returns if the [Fusion.Simulation](#) contains a reference to a [NetworkObject](#) in the current State.

6.235.3.13 FindObject()

```
NetworkObject FindObject (
    NetworkId networkId )
```

Get the [NetworkObject](#) instance for this [NetworkRunner](#) from a [NetworkId](#).

Parameters

<i>networkId</i>	NetworkID to look forward
------------------	---------------------------

Returns

null if object cannot be found.

6.235.3.14 GetAllBehaviours()

```
SimulationBehaviour [ ] GetAllBehaviours (
    Type type )
```

Returns an array of all [SimulationBehaviour](#) instances registered with this [NetworkRunner](#).

Parameters

<code>type</code>	The type of the behaviours to be returned.
-------------------	--

Returns

An array of [SimulationBehaviour](#) instances of the specified type.

6.235.3.15 GetAllBehaviours< T >() [1/2]

```
List<T> GetAllBehaviours< T > ( )
```

Get a list with all behaviours of the desired type that are registered on the [NetworkRunner](#).

Template Parameters

<code>T</code>	SimulationBehaviour type
----------------	--

Returns

The result list

Type Constraints

`T : SimulationBehaviour`

6.235.3.16 GetAllBehaviours< T >() [2/2]

```
void GetAllBehaviours< T > (
    List< T > result )
```

Add on the list all behaviours of the desired type that are registered on the [NetworkRunner](#). Note: The list will not be cleared before adding the results.

Parameters

<code>result</code>	The list to add the behaviours
---------------------	--------------------------------

Template Parameters

<i>T</i>	SimulationBehaviour type
----------	--------------------------

Type Constraints

T : **SimulationBehaviour**

6.235.3.17 GetAllNetworkObjects()

```
List<NetworkObject> GetAllNetworkObjects ( )
```

Retrieves a list of all network objects in the simulation.

Returns

A list of **NetworkObject** instances.

6.235.3.18 GetAreaOfInterestGizmoData()

```
void GetAreaOfInterestGizmoData ( 
    List<(Vector3 center, Vector3 size, int playerCount, int objectCount)> result )
```

Populates the provided list with data about the current Area of Interest (AOI) cells. Each element in the list represents one AOI cell.

Parameters

<i>result</i>	The list to be populated with AOI cell data. Each tuple in the list contains the center of the AOI cell, its size, the count of players in the cell, and the count of objects in the cell.
---------------	--

6.235.3.19 GetAvailableRegions()

```
static Task<List<RegionInfo>> GetAvailableRegions ( 
    string appId = default,
    CancellationToken cancellationToken = default ) [static]
```

Starts an operation to retrieves the list of available regions.

Parameters

<i>appId</i>	Optional App ID. If not provided, the method will use the global App ID from the PhotonAppSettings.
<i>cancellationToken</i>	Optional CancellationToken parameter that can be used to cancel the operation.

Returns

Returns a list with information about all the available regions.

6.235.3.20 GetInputForPlayer< T >()

```
T? GetInputForPlayer< T > (
    PlayerRef player )
```

Returns the [NetworkInput](#) data from player, converted to the indicated [INetworkInput](#).

Type Constraints

T : unmanaged

T : INetworkInput

6.235.3.21 GetInstancesEnumerator()

```
static List<NetworkRunner>.Enumerator GetInstancesEnumerator ( ) [static]
```

Get enumerator for the collection of all [NetworkRunners](#). Allows to enumerate alloc-free.

6.235.3.22 GetInterfaceListHead()

```
SimulationBehaviourListScope GetInterfaceListHead (
    Type type,
    int index,
    out SimulationBehaviour head )
```

Get the interface list head.

Parameters

<i>type</i>	The interface type
<i>index</i>	The desired index on the list of behaviourList
<i>head</i>	The head reference

Returns

A disposable [SimulationBehaviourListScope](#) to be used on an using scope

6.235.3.23 GetInterfaceListNext()

```
SimulationBehaviour GetInterfaceListNext (
    SimulationBehaviour behaviour )
```

Get the next behaviour

Parameters

<i>behaviour</i>	The reference behaviour to get the next one
------------------	---

Returns

Gives the next behaviour

6.235.3.24 GetInterfaceListPrev()

```
SimulationBehaviour GetInterfaceListPrev (
    SimulationBehaviour behaviour )
```

Get the previous behaviour

Parameters

<i>behaviour</i>	The reference behaviour to get the previous one
------------------	---

Returns

Gives the previous behaviour

6.235.3.25 GetInterfaceListsCount()

```
int GetInterfaceListsCount (
    Type type )
```

Get the number of interfaces of the desired type that are registered on the behaviour updater.

Parameters

<i>type</i>	The interface type
-------------	--------------------

Returns

The number of interfaces

6.235.3.26 GetMemorySnapshot()

```
void GetMemorySnapshot (
    MemoryStatisticsSnapshot.TargetAllocator targetAllocator,
    ref MemoryStatisticsSnapshot snapshot )
```

Gets a memory snapshot of the simulation.

Parameters

<i>targetAllocator</i>	The target allocator for the memory statistics. Must be a valid value from the MemoryStatisticsSnapshot.TargetAllocator enum.
<i>snapshot</i>	A reference to the MemoryStatisticsSnapshot struct that will store the memory snapshot.

6.235.3.27 GetObjectsInAreaOfInterestForPlayer()

```
List<NetworkId> GetObjectsInAreaOfInterestForPlayer (
    PlayerRef player )
```

Retrieves a list of network object IDs that are in the area of interest for the specified player. Server only.

Parameters

<i>player</i>	The player for whom the area of interest is being queried.
---------------	--

Returns

A list of network object IDs in the area of interest for the player.

6.235.3.28 GetPhysicsScene()

```
PhysicsScene GetPhysicsScene ( )
```

Get the 3D Physics scene being used by this Runner.

6.235.3.29 GetPhysicsScene2D()

```
PhysicsScene2D GetPhysicsScene2D ( )
```

Get the 2D Physics scene being used by this Runner.

6.235.3.30 GetPlayerActorId()

```
int? GetPlayerActorId (
    PlayerRef player )
```

Gets Player's Actor Number (ID).

If used in Shared Mode, every client can get this information. If used in Client Server Mode, only the Server is able to get this information.

Parameters

<i>player</i>	PlayerRef to get the Actor Number (ID)
---------------	--

Returns

Actor Number associated with the [PlayerRef](#), otherwise null.

6.235.3.31 GetPlayerConnectionToken()

```
byte [] GetPlayerConnectionToken (
    PlayerRef player = default )
```

Returns a copy of the Connection Token used by a Player when connecting to this Server. Only available on Server. It will return null if running on a Client or the Connection token is missing

Parameters

<i>player</i>	PlayerRef to check for a Connection Token
---------------	---

Returns

Copy of the Connection Token

6.235.3.32 GetPlayerConnectionType()

```
ConnectionType GetPlayerConnectionType (
    PlayerRef player )
```

Return the [ConnectionType](#) with a Remote [PlayerRef](#). Valid only when invoked from a Server ([NetworkRunner.IsServer](#))

Parameters

<i>player</i>	Remote Player to check the ConnectionType
---------------	---

Returns

[ConnectionType](#) with a [PlayerRef](#)

6.235.3.33 GetPlayerObject()

```
NetworkObject GetPlayerObject (
    PlayerRef player )
```

Gets the network object associated with a specific player

Parameters

<i>player</i>	PlayerRef to get the network object
---------------	---

Returns

Network object if one is associated with the player

6.235.3.34 GetPlayerRtt()

```
double GetPlayerRtt (
    PlayerRef playerRef )
```

Returns the player round trip time (ping) in seconds

Parameters

<i>playerRef</i>	The player you want the round trip time for
------------------	---

6.235.3.35 GetPlayerUserId()

```
string GetPlayerUserId (
    PlayerRef player = default )
```

Gets Player's UserID.

If used in Shared Mode, every client can get this information. If used in Client Server Mode, only the Server is able to get this information.

Parameters

<i>player</i>	PlayerRef to get the UserID. If no PlayerRef is passed, the UserID of the local client is returned instead.
---------------	---

Returns

UserID if valid player found, otherwise null.

6.235.3.36 GetRawInputForPlayer()

```
NetworkInput? GetRawInputForPlayer (
    PlayerRef player )
```

Returns the unconverted unsafe NetworkInput for the indicated player.

6.235.3.37 GetResumeSnapshotNetworkObjects()

```
IEnumerable<NetworkObject> GetResumeSnapshotNetworkObjects ( )
```

Iterate over the old NetworkObjects from the Resume Snapshot

Returns

Iterable list of NetworkObject

6.235.3.38 GetResumeSnapshotNetworkSceneObjects()

```
IEnumerable<(NetworkObject, NetworkObjectHeaderPtr)> GetResumeSnapshotNetworkSceneObjects ( )
```

Iterate over the Scene NetworkObjects from the Resume Snapshot while giving the reference of the old Snapshot data associated with that particular Scene Object

Returns

Iterable list of Scene NetworkObject and Scene Object Header

6.235.3.39 GetRpcTargetStatus()

```
RpcTargetStatus GetRpcTargetStatus (
    PlayerRef target )
```

Return the [RpcTargetStatus](#) for a specific player.

6.235.3.40 GetRunnerForObject()

```
static NetworkRunner GetRunnerForObject (
    GameObject gameObject ) [static]
```

Get the [NetworkRunner](#) a GameObject instance belongs to.

Parameters

<i>gameObject</i>	GameObject to check for a NetworkRunner
-------------------	---

Returns

[NetworkRunner](#) reference, or null if not found

6.235.3.41 GetRunnerForScene()

```
static NetworkRunner GetRunnerForScene (
    Scene scene ) [static]
```

Get the [NetworkRunner](#) from a specific Scene

Parameters

<i>scene</i>	Scene to check for a NetworkRunner
--------------	--

Returns

[NetworkRunner](#) reference, or null if not found

6.235.3.42 GetSingleton< T >()

```
T GetSingleton< T > ( )
```

Ensures that a specific [SimulationBehaviour](#) component exists on this [NetworkRunner](#) gameobject.

Type Constraints

T : [SimulationBehaviour](#)

6.235.3.43 HasSingleton< T >()

```
bool HasSingleton< T > ( )
```

Returns if a given [SimulationBehaviour](#) is present in this [NetworkRunner](#) gameobject.

Returns

Returns true if the [SimulationBehaviour](#) was found

Type Constraints

T : [SimulationBehaviour](#)

6.235.3.44 InstantiateInRunnerScene() [1/2]

```
GameObject InstantiateInRunnerScene (
    GameObject original )
```

Instantiates an object in the scene of this runner

6.235.3.45 InstantiateInRunnerScene() [2/2]

```
GameObject InstantiateInRunnerScene (
    GameObject original,
    Vector3 position,
    Quaternion rotation )
```

Instantiates an object in the scene of this runner

6.235.3.46 InstantiateInRunnerScene< T >() [1/2]

```
T InstantiateInRunnerScene< T > (
    T original )
```

Instantiates an object in the scene of this runner

Type Constraints

T : [Component](#)

6.235.3.47 InstantiateInRunnerScene< T >() [2/2]

```
T InstantiateInRunnerScene< T > (
    T original,
    Vector3 position,
    Quaternion rotation )
```

Instantiates an object in the scene of this runner

Type Constraints

T: *Component*

6.235.3.48 InvokeSceneLoadDone()

```
void InvokeSceneLoadDone (
    in SceneLoadDoneArgs info )
```

Invoke [INetworkRunnerCallbacks.OnSceneLoadDone\(NetworkRunner\)](#) on all implementations

6.235.3.49 InvokeSceneLoadStart()

```
void InvokeSceneLoadStart (
    SceneRef sceneRef )
```

Invoke [INetworkRunnerCallbacks.OnSceneLoadStart\(NetworkRunner\)](#) on all implementations

6.235.3.50 IsInterestedIn()

```
bool? IsInterestedIn (
    NetworkObject obj,
    PlayerRef player )
```

Test if a player has Interest in a [NetworkObject](#).

Returns

Returns null if interest cannot be determined (clients without State Authority are not aware of other client's Object Interest)

6.235.3.51 IsPlayerValid()

```
bool IsPlayerValid (
    PlayerRef player )
```

Checks if the provided player is valid in the current simulation.

Parameters

<i>player</i>	The player reference to be validated.
---------------	---------------------------------------

Returns

Returns true if the player is valid, false otherwise.

6.235.3.52 JoinSessionLobby()

```
async Task<StartGameResult> JoinSessionLobby (
    SessionLobby sessionLobby,
    string lobbyID = null,
    AuthenticationValues authentication = null,
    FusionAppSettings customAppSettings = null,
    bool? useDefaultCloudPorts = false,
    CancellationToken cancellationToken = default,
    bool useCachedRegions = true )
```

Join the Peer to a specific Lobby, either a prebuild or a custom one.

More about matchmaking: <https://doc.photonengine.com/en-us/fusion/current/manual/matchmaking>

Parameters

<i>sessionLobby</i>	Lobby Type to Join
<i>lobbyID</i>	Lobby ID
<i>authentication</i>	Authentication Values used to authenticate this peer
<i>customAppSettings</i>	Custom Photon Application Settings
<i>useDefaultCloudPorts</i>	Signal if the LoadBalancingClient should use the Default or Alternative Ports
<i>cancellationToken</i>	Optional Cancellation Token
<i>useCachedRegions</i>	Signal if the cached regions ping should be used to speed up connection

Returns

Async Task to Join a Session Lobby. Can be used to wait for the process to be finished.

6.235.3.53 LoadScene() [1/3]

```
NetworkSceneAsyncOp LoadScene (
    SceneRef sceneRef,
    LoadSceneMode loadSceneMode = LoadSceneMode.Single,
    LocalPhysicsMode localPhysicsMode = LocalPhysicsMode.None,
    bool setActiveOnLoad = DefaultSetActiveOnLoad )
```

Loads a scene

Parameters

<i>sceneRef</i>	Reference to the scene to load
<i>loadSceneMode</i>	Scene load mode
<i>localPhysicsMode</i>	Scene physics mode
<i>setActiveOnLoad</i>	Should the scene be set as active when loaded

Returns

Scene Load operation

6.235.3.54 LoadScene() [2/3]

```
NetworkSceneAsyncOp LoadScene (
    string sceneName,
    LoadSceneMode loadSceneMode = LoadSceneMode.Single,
    LocalPhysicsMode localPhysicsMode = LocalPhysicsMode.None,
    bool setActiveOnLoad = DefaultSetActiveOnLoad )
```

Loads a scene

Parameters

<i>sceneName</i>	Name of the scene to load
<i>loadSceneMode</i>	Scene load mode
<i>localPhysicsMode</i>	Scene physics mode
<i>setActiveOnLoad</i>	Should the scene be set as active when loaded

Returns

Scene Load operation

6.235.3.55 LoadScene() [3/3]

```
NetworkSceneAsyncOp LoadScene (
    string sceneName,
    LoadSceneParameters parameters,
    bool setActiveOnLoad = DefaultSetActiveOnLoad )
```

Loads a scene

Parameters

<i>sceneName</i>	Name of the scene to load
<i>parameters</i>	Parameters to use when loading the scene
<i>setActiveOnLoad</i>	Should the scene be set as active when loaded

Returns

Scene Load operation

6.235.3.56 MakeDontDestroyOnLoad()

```
void MakeDontDestroyOnLoad (
    GameObject obj )
```

Mark an object as DontDestroyOnLoad.

Parameters

<i>obj</i>	Object to mark
------------	----------------

6.235.3.57 MoveGameObjectToSameScene()

```
bool MoveGameObjectToSameScene (
    GameObject gameObj,
    GameObject other )
```

Moves a GameObject to the same scene as another GameObject

Parameters

<i>gameObj</i>	Game Object to move
<i>other</i>	Game Object to move to the same scene as

Returns

True if the object was moved, false otherwise

6.235.3.58 MoveGameObjectToScene()

```
bool MoveGameObjectToScene (
    GameObject gameObj,
    SceneRef sceneRef )
```

Moves a GameObject to a specific scene

Parameters

<i>gameObj</i>	Game Object to move
<i>sceneRef</i>	Scene to move the object to

Returns

True if the object was moved, false otherwise

6.235.3.59 MoveToRunnerScene()

```
void MoveToRunnerScene (
    GameObject instance,
    SceneRef? targetSceneRef = null )
```

Moves an object to the scene of this runner

Parameters

<i>instance</i>	Object to move
<i>targetSceneRef</i>	Target scene to move the object to

6.235.3.60 MoveToRunnerScene< T >()

```
void MoveToRunnerScene< T > (
    T component )
```

Moves an object to the scene of this runner

Template Parameters

<i>T</i>	
----------	--

Parameters

<i>component</i>	Component of object to move
------------------	-----------------------------

Type Constraints

T: *Component*

6.235.3.61 ObjectDelegate()

```
delegate void ObjectDelegate (
    NetworkRunner runner,
    NetworkObject obj )
```

Delegate type for object callback

6.235.3.62 OnBeforeSpawned()

```
delegate void OnBeforeSpawned (
    NetworkRunner runner,
    NetworkObject obj )
```

Delegate type for on before spawned callback

6.235.3.63 PushHostMigrationSnapshot()

```
async Task<bool> PushHostMigrationSnapshot ( )
```

Compute and send a Host Migration Snapshot to the Photon Cloud

Returns

Task with the result of the operation. True if it was successful, false otherwise.

6.235.3.64 RegisterSceneObjects()

```
int RegisterSceneObjects (
    SceneRef scene,
    NetworkObject[ ] objects,
    NetworkSceneLoadId loadId = default )
```

Registers scene objects to the network.

Parameters

<i>scene</i>	The scene reference. Must be valid.
<i>objects</i>	Array of NetworkObject instances to be registered. Must not be null.
<i>loadId</i>	Optional NetworkSceneLoadId . Default value is used if not provided.

Returns

The number of objects registered.

Exceptions

ArgumentException	Thrown when the provided scene is not valid.
ArgumentNullException	Thrown when the provided objects array is null.

6.235.3.65 RemoveCallbacks()

```
void RemoveCallbacks (
    params INetworkRunnerCallbacks[ ] callbacks )
```

Unregister an [INetworkRunnerCallbacks](#) instance for callbacks from this [NetworkRunner](#).

Parameters

<i>callbacks</i>	Callbacks to unregister
------------------	-------------------------

6.235.3.66 RemoveGlobal()

```
void RemoveGlobal (
    SimulationBehaviour instance )
```

Removes a specific [SimulationBehaviour](#) from this [NetworkObject](#) gameobject, if it exists.

6.235.3.67 RenderInternal()

```
void RenderInternal ( )
```

This method is meant to be called by [INetworkRunnerUpdater](#).

6.235.3.68 SendReliableDataToPlayer()

```
void SendReliableDataToPlayer (
    PlayerRef player,
    ReliableKey key,
    byte[ ] data )
```

Sends a reliable data buffer to a target player.

Parameters

<i>player</i>	The player who should receive the buffer.
<i>key</i>	The key associated with the reliable data.
<i>data</i>	The data buffer to be sent.

6.235.3.69 SendReliableDataToServer()

```
void SendReliableDataToServer (
    ReliableKey key,
    byte[ ] data )
```

Sends a reliable data buffer to the server.

Parameters

<i>key</i>	The key associated with the reliable data.
<i>data</i>	The data buffer to be sent.

If the runner is a client, the data is sent to the server (connection index 0) with the player's index. If the runner is a server, the data is sent via the simulation callbacks.

6.235.3.70 SendRpc() [1/2]

```
void SendRpc (
    SimulationMessage * message )
```

Sends RPC message. Not meant to be used directly, ILWeaver calls this.

Parameters

<i>message</i>	SimulationMessage to send
----------------	---------------------------

6.235.3.71 SendRpc() [2/2]

```
void SendRpc (
    SimulationMessage * message,
    out RpcSendResult info )
```

Sends RPC message. Not meant to be used directly, ILWeaver calls this.

Parameters

<i>message</i>	SimulationMessage to send
<i>info</i>	RpcSendResult

6.235.3.72 SetAreaOfInterestCellSize()

```
void SetAreaOfInterestCellSize (
    int size )
```

Set the area of interest cell size

Parameters

<i>size</i>	Size of the cell
-------------	------------------

Exceptions

<i>Exception</i>	Can't change cell size in shared mode
------------------	---------------------------------------

6.235.3.73 SetAreaOfInterestGrid()

```
void SetAreaOfInterestGrid (
    int x,
    int y,
    int z )
```

Set the area of interest grid dimensions

Parameters

<i>x</i>	X dimension
<i>y</i>	Y dimension
<i>z</i>	Z dimension

Exceptions

<i>Exception</i>	Can't change grid size in shared mode
------------------	---------------------------------------

6.235.3.74 SetBehaviourReplicateTo()

```
void SetBehaviourReplicateTo (
    NetworkBehaviour behaviour,
    PlayerRef player,
    bool replicate )
```

Controls if a specific network behaviours state is replicated to a player or not

Parameters

<i>behaviour</i>	The behaviour to change replication status for
<i>player</i>	The player to change replication status for
<i>replicate</i>	true = replicate, false = don't replicate

6.235.3.75 SetBehaviourReplicateToAll()

```
void SetBehaviourReplicateToAll (
    NetworkBehaviour behaviour,
    bool replicate )
```

Controls if a specific network behaviours state is replicated to all players or not

Parameters

<i>behaviour</i>	The behaviour to change replication status for
<i>replicate</i>	true = replicate, false = don't replicate

6.235.3.76 SetIsSimulated()

```
bool SetIsSimulated (
    NetworkObject obj,
    bool simulate )
```

Sets the simulation state for this object, if it takes part in the NetworkFixedUpdate, etc.

Parameters

<i>obj</i>	the object to change state for
<i>simulate</i>	true if it should be simulated, false if otherwise

Returns

true if the state of the object changed, false otherwise

6.235.3.77 SetMasterClient()

```
void SetMasterClient (
    PlayerRef player )
```

Promote a player to be the new master client. Only the master client is able to call this method

Parameters

<i>player</i>	The player to be promoted to master client
---------------	--

6.235.3.78 SetPlayerAlwaysInterested()

```
void SetPlayerAlwaysInterested (
    PlayerRef player,
    NetworkObject networkObject,
    bool alwaysInterested )
```

Flags this player as always interested in this object. Means it does not have to be in a players area of interest to be replicated. Only the [NetworkObject State Authority](#) can set interest.

Parameters

<i>player</i>	The player
<i>networkObject</i>	The object
<i>alwaysInterested</i>	If he's always interested, or not.

6.235.3.79 SetPlayerObject()

```
void SetPlayerObject (
    PlayerRef player,
    NetworkObject networkObject )
```

Sets the network object associated with this player

Parameters

<i>player</i>	PlayerRef to set the network object
<i>networkObject</i>	Network object to associate with the player

6.235.3.80 SetSimulateMultiPeerPhysics()

```
void SetSimulateMultiPeerPhysics (
    bool value )
```

Set the value for simulating physics scenes when using multi-peer. Restores the default physics simulation settings if false, overrides the default if true. (2D: SimulationMode2D.Script | 3D: AutoSimulation = false)

Parameters

<i>value</i>	The value to set. True to simulate physics scenes, false otherwise.
--------------	---

6.235.3.81 Shutdown()

```
Task Shutdown (
    bool destroyGameObject = true,
    ShutdownReason shutdownReason = ShutdownReason.Ok,
    bool forceShutdownProcedure = false )
```

Initiates a Simulation.Dispose.

6.235.3.82 SinglePlayerContinue()

```
void SinglePlayerContinue ( )
```

Continues a paused game in single player

6.235.3.83 SinglePlayerPause() [1/2]

```
void SinglePlayerPause ( )
```

Pauses the game in single player

6.235.3.84 SinglePlayerPause() [2/2]

```
void SinglePlayerPause (
    bool paused )
```

Sets the paused state in a single player

6.235.3.85 Spawn() [1/5]

```
NetworkObject Spawn (
    GameObject prefab,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a `NetworkObject` using a `GameObject`. The supplied `GameObject` must have a `NetworkObject` component.

Parameters

<i>prefab</i>	A GameObject with a NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Returns

NetworkObject reference, or null if it was not able to spawn the object

6.235.3.86 Spawn() [2/5]

```
NetworkObject Spawn (
    NetworkObject prefab,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a NetworkObject using a NetworkObject prefab. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use NetworkTransform (or any custom class derived from NetworkTRSP) to replicate the initial transform state.

Parameters

<i>prefab</i>	Prefab used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Returns

NetworkObject reference, or null if it was not able to spawn the object

6.235.3.87 Spawn() [3/5]

```
NetworkObject Spawn (
    NetworkObjectGuid prefabGuid,
```

```
Vector3? position = null,
Quaternion? rotation = null,
PlayerRef? inputAuthority = null,
OnBeforeSpawned onBeforeSpawned = null,
NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObjectGuid](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefabGuid</i>	Object Guid used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Returns

[NetworkObject](#) reference, or null if it was not able to spawn the object

6.235.3.88 Spawn() [4/5]

```
NetworkObject Spawn (
    NetworkPrefabId typeId,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabId](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>typeId</i>	Prefab ID used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Returns

[NetworkObject](#) reference, or null if it was not able to spawn the object

6.235.3.89 Spawn() [5/5]

```
NetworkObject Spawn (
    NetworkPrefabRef prefabRef,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabRef](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefabRef</i>	Prefab Ref used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

Returns

[NetworkObject](#) reference, or null if it was not able to spawn the object

6.235.3.90 Spawn< T >()

```
T Spawn< T > (
    T prefab,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a Component type that is part of a [NetworkObject](#)

Template Parameters

<i>T</i>	Must be a Type derived from SimulationBehaviour
----------	---

Parameters

<i>prefab</i>	SimulationBehaviour used to spawn the NetworkObject
---------------	---

Returns

`T` reference, or null if it was not able to spawn the object

Parameters

<code>position</code>	Spawn Position
<code>rotation</code>	Spawn Rotation
<code>inputAuthority</code>	Player Input Authority
<code>onBeforeSpawned</code>	OnBeforeSpawned reference
<code>flags</code>	Spawn flags

Type Constraints

`T : SimulationBehaviour`

6.235.3.91 SpawnAsync() [1/5]

```
NetworkSpawnOp SpawnAsync (
    GameObject prefab,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default,
    NetworkObjectSpawnDelegate onCompleted = null )
```

Attempts to network instantiate a [NetworkObject](#) using a `GameObject`. The supplied `GameObject` must have a [NetworkObject](#) component.

Parameters

<code>prefab</code>	A <code>GameObject</code> with a NetworkObject
<code>position</code>	Spawn Position
<code>rotation</code>	Spawn Rotation
<code>inputAuthority</code>	Player Input Authority
<code>onBeforeSpawned</code>	OnBeforeSpawned reference
<code>flags</code>	Spawn flags

,

Parameters

<code>onCompleted</code>	A callback to fire once the spawn is done.
--------------------------	--

6.235.3.92 SpawnAsync() [2/5]

```
NetworkSpawnOp SpawnAsync (
    NetworkObject prefab,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default,
    NetworkObjectSpawnDelegate onCompleted = null )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObject](#) prefab. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefab</i>	Prefab used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

,

Parameters

<i>onCompleted</i>	A callback to fire once the spawn is done.
--------------------	--

6.235.3.93 SpawnAsync() [3/5]

```
NetworkSpawnOp SpawnAsync (
    NetworkObjectGuid prefabGuid,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default,
    NetworkObjectSpawnDelegate onCompleted = null )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObjectGuid](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefabGuid</i>	Object Guid used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation

Parameters

<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

,

Parameters

<i>onCompleted</i>	A callback to fire once the spawn is done.
--------------------	--

6.235.3.94 SpawnAsync() [4/5]

```
NetworkSpawnOp SpawnAsync (
    NetworkPrefabId typeId,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default,
    NetworkObjectSpawnDelegate onCompleted = null )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabId](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>typeId</i>	Prefab ID used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

,

Parameters

<i>onCompleted</i>	A callback to fire once the spawn is done.
--------------------	--

6.235.3.95 SpawnAsync() [5/5]

```
NetworkSpawnOp SpawnAsync (
```

```
NetworkPrefabRef prefabRef,
Vector3? position = null,
Quaternion? rotation = null,
PlayerRef? inputAuthority = null,
OnBeforeSpawned onBeforeSpawned = null,
NetworkSpawnFlags flags = default,
NetworkObjectSpawnDelegate onCompleted = null )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabRef](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefabRef</i>	Prefab Ref used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags

,

Parameters

<i>onCompleted</i>	A callback to fire once the spawn is done.
--------------------	--

6.235.3.96 [SpawnAsync< T >\(\)](#)

```
NetworkSpawnOp SpawnAsync< T > (
T prefab,
Vector3? position = null,
Quaternion? rotation = null,
PlayerRef? inputAuthority = null,
OnBeforeSpawned onBeforeSpawned = null,
NetworkSpawnFlags flags = default,
NetworkObjectSpawnDelegate onCompleted = null )
```

Attempts to network instantiate a [NetworkObject](#) using a Component type that is part of a [NetworkObject](#)

Template Parameters

<i>T</i>	Must be a Type derived from SimulationBehaviour
----------	---

Parameters

<i>prefab</i>	SimulationBehaviour used to spawn the NetworkObject
---------------	---

Returns

`T` reference, or null if it was not able to spawn the object

Parameters

<code>onCompleted</code>	A callback to fire once the spawn is done.
<code>position</code>	Spawn Position
<code>rotation</code>	Spawn Rotation
<code>inputAuthority</code>	Player Input Authority
<code>onBeforeSpawned</code>	OnBeforeSpawned reference
<code>flags</code>	Spawn flags

Type Constraints

`T : SimulationBehaviour`

6.235.3.97 StartGame()

```
Task<StartGameResult> StartGame (
    StartGameArgs args )
```

Starts the local [Fusion](#) Runner and takes care of all major setup necessary

More about matchmaking: <https://doc.photonengine.com/en-us/fusion/current/manual/matchmaking>

Parameters

<code>args</code>	Custom arguments used to setup the Fusion Simulation
-------------------	--

Returns

Task that can be awaited to chain actions

6.235.3.98 TryFindBehaviour()

```
bool TryFindBehaviour (
    NetworkBehaviourId behaviourId,
    out NetworkBehaviour behaviour )
```

Get the [NetworkBehaviour](#) instance for this [NetworkRunner](#) from a [NetworkBehaviourId](#).

Parameters

<i>behaviour</i> \leftarrow	<code>NetworkBehaviourId</code> to look forward
<i>id</i>	

Returns

True if object was found.

6.235.3.99 TryFindBehaviour< T >()

```
bool TryFindBehaviour< T > (
    NetworkBehaviourId id,
    out T behaviour )
```

Try to find a `NetworkBehaviour` with the provided `NetworkBehaviourId`.

Parameters

<i>id</i>	The <code>NetworkBehaviourId</code> to search for
<i>behaviour</i>	The behaviour found

Template Parameters

<i>T</i>	A <code>NetworkBehaviour</code> type
----------	--------------------------------------

Returns

Returns true if the behaviour was found and it is alive. False otherwise

Type Constraints

T : `NetworkBehaviour`

6.235.3.100 TryFindObject()

```
bool TryFindObject (
    NetworkId objectId,
    out NetworkObject networkObject )
```

Get the `NetworkObject` instance for this `NetworkRunner` from a `NetworkId`.

Parameters

<i>objectId</i>	Object NetworkID to look forward
<i>networkObject</i>	NetworkObject reference, if found

Returns

True if object was found.

6.235.3.101 TryGetBehaviourStatistics()

```
bool TryGetBehaviourStatistics (
    Type behaviourType,
    out BehaviourStatisticsSnapshot behaviourStatisticsSnapshot )
```

Tries to get the statistics snapshot for a specified behaviour type.

Parameters

<i>behaviourType</i>	The type of the behaviour for which to get the statistics snapshot.
<i>behaviourStatisticsSnapshot</i>	When this method returns, contains the statistics snapshot for the specified behaviour type, if found; otherwise, the default value.

Returns

true if the statistics snapshot for the specified behaviour type is found; otherwise, false.

6.235.3.102 TryGetFusionStatistics()

```
bool TryGetFusionStatistics (
    out FusionStatisticsManager statisticsManager )
```

Tries to get the FusionStatisticsManager from the simulation.

Parameters

<i>statisticsManager</i>	The FusionStatisticsManager returned by the method
--------------------------	--

Returns

True if the FusionStatisticsManager is successfully retrieved, otherwise false

6.235.3.103 TryGetInputForPlayer< T >()

```
bool TryGetInputForPlayer< T > (
    PlayerRef player,
    out T input )
```

Outputs the [NetworkInput](#) from player, translated to the indicated [INetworkInput](#).

Type Constraints

T : unmanaged

T : INetworkInput

6.235.3.104 TryGetNetworkedBehaviourFromNetworkedObjectRef< T >()

```
T TryGetNetworkedBehaviourFromNetworkedObjectRef< T > (
    NetworkId networkId )
```

Tries to return the first instance of T found on the root of a [NetworkObject](#).

Template Parameters

T	The type of the component to search for
----------	---

Parameters

networkId	NetworkId of the NetworkObject to search for
------------------	--

Returns

Returns the found component. Null if the [NetworkObject](#) cannot be found, or if T cannot be found on the GameObject.

Type Constraints

T : NetworkBehaviour

6.235.3.105 TryGetNetworkedBehaviourId()

```
NetworkBehaviourId TryGetNetworkedBehaviourId (
    NetworkBehaviour behaviour )
```

Tries to return a [NetworkBehaviourId](#) for the [NetworkBehaviour](#) provided.

Parameters

<i>behaviour</i>	<code>NetworkBehaviour</code> to get the <code>NetworkBehaviourId</code> from
------------------	---

Returns

Returns a `NetworkBehaviourId` to the provided behaviour. Returns default if the behaviour is not alive or the `NetworkObject` that has this behaviour is not valid.

6.235.3.106 TryGetObjectRefFromNetworkedBehaviour()

```
NetworkId TryGetObjectRefFromNetworkedBehaviour (
    NetworkBehaviour behaviour )
```

Tries to return the behaviour `NetworkId`.

Parameters

<i>behaviour</i>	<code>NetworkBehaviour</code> to get the <code>NetworkId</code> from
------------------	--

Returns

Returns the `NetworkId` of the provided behaviour. Returns default if the behaviour is not alive or the `NetworkObject` that has this behaviour is not valid.

6.235.3.107 TryGetPhysicsInfo()

```
bool TryGetPhysicsInfo (
    out NetworkPhysicsInfo info )
```

Try to get the physics info.

Parameters

<i>info</i>	Network physics info
-------------	----------------------

Returns

True if the physics info exists, otherwise false.

6.235.3.108 TryGetPlayerObject()

```
bool TryGetPlayerObject (
    PlayerRef player,
    out NetworkObject networkObject )
```

Try to gets the [NetworkObject](#) associated with a specific player

Parameters

<i>player</i>	PlayerRef to get the network object
<i>networkObject</i>	Network object if one is associated with the player

Returns

Signals if it was able to get a [NetworkObject](#) for the player provided

6.235.3.109 TryGetSceneInfo()

```
bool TryGetSceneInfo (
    out NetworkSceneInfo sceneInfo )
```

Tries to get the [NetworkSceneInfo](#) of this [NetworkRunner](#).

Parameters

<i>sceneInfo</i>	The result NetworkSceneInfo
------------------	---

Returns

Returns true if it was able to get the scene info

6.235.3.110 TrySetPhysicsInfo()

```
bool TrySetPhysicsInfo (
    NetworkPhysicsInfo info )
```

Try to set the physics info.

Parameters

<i>info</i>	Network physics info
-------------	----------------------

Returns

True if the physics info was set, otherwise false.

Exceptions

<i>InvalidOperationException</i>	Thrown if the runner does not have the scene authority.
----------------------------------	---

6.235.3.111 TrySpawn() [1/5]

```
NetworkSpawnStatus TrySpawn (
    GameObject prefab,
    out NetworkObject obj,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [GameObject](#). The supplied [GameObject](#) must have a [NetworkObject](#) component.

Parameters

<i>prefab</i>	A GameObject with a NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags
<i>obj</i>	Spawned NetworkObject reference

Returns

[NetworkSpawnStatus](#) reference, or null if it was not able to spawn the object

6.235.3.112 TrySpawn() [2/5]

```
NetworkSpawnStatus TrySpawn (
    NetworkObject prefab,
    out NetworkObject obj,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObject](#) prefab. Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefab</i>	Prefab used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags
<i>obj</i>	Spawned NetworkObject reference

Returns

[NetworkSpawnStatus](#) reference, or null if it was not able to spawn the object

6.235.3.113 TrySpawn() [3/5]

```
NetworkSpawnStatus TrySpawn (
    NetworkObjectGuid prefabGuid,
    out NetworkObject obj,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkObjectGuid](#) Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefabGuid</i>	Object Guid used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags
<i>obj</i>	Spawned NetworkObject reference

Returns

[NetworkSpawnStatus](#) reference, or null if it was not able to spawn the object

6.235.3.114 TrySpawn() [4/5]

```
NetworkSpawnStatus TrySpawn (
    NetworkPrefabId typeId,
    out NetworkObject obj,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabId](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>typeId</i>	Prefab ID used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags
<i>obj</i>	Spawned NetworkObject reference

Returns

[NetworkSpawnStatus](#) reference, or null if it was not able to spawn the object

6.235.3.115 TrySpawn() [5/5]

```
NetworkSpawnStatus TrySpawn (
    NetworkPrefabRef prefabRef,
    out NetworkObject obj,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a [NetworkObject](#) using a [NetworkPrefabRef](#). Note: position and rotation values are only used locally for the instantiation of the object, and are not inherently networked. Use [NetworkTransform](#) (or any custom class derived from [NetworkTRSP](#)) to replicate the initial transform state.

Parameters

<i>prefabRef</i>	Prefab Ref used to spawn the NetworkObject
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	OnBeforeSpawned reference
<i>flags</i>	Spawn flags
<i>obj</i>	Spawned NetworkObject reference

Returns

`NetworkSpawnStatus` reference, or null if it was not able to spawn the object

6.235.3.116 TrySpawn< T >()

```
NetworkSpawnStatus TrySpawn< T > (
    T prefab,
    out T obj,
    Vector3? position = null,
    Quaternion? rotation = null,
    PlayerRef? inputAuthority = null,
    OnBeforeSpawned onBeforeSpawned = null,
    NetworkSpawnFlags flags = default )
```

Attempts to network instantiate a `NetworkObject` using a Component type that is part of a `NetworkObject`

Template Parameters

<i>T</i>	Must be a Type derived from <code>SimulationBehaviour</code>
----------	--

Parameters

<i>prefab</i>	<code>SimulationBehaviour</code> used to spawn the <code>NetworkObject</code>
---------------	---

Returns

`T` reference, or null if it was not able to spawn the object

Parameters

<i>obj</i>	Spawned <code>NetworkObject</code> reference
<i>position</i>	Spawn Position
<i>rotation</i>	Spawn Rotation
<i>inputAuthority</i>	Player Input Authority
<i>onBeforeSpawned</i>	<code>OnBeforeSpawned</code> reference
<i>flags</i>	Spawn flags

Type Constraints

T : `SimulationBehaviour`

6.235.3.117 UnloadScene()

```
NetworkSceneAsyncOp UnloadScene (
    string sceneName )
```

Unloads a scene

Parameters

<code>sceneName</code>	Name of the scene to unload
------------------------	-----------------------------

Returns

Scene Unload operation

6.235.3.118 UpdateInternal()

```
void UpdateInternal (
    double dt )
```

This method is meant to be called by [INetworkRunnerUpdater](#).

6.235.4 Property Documentation**6.235.4.1 ActivePlayers**

```
IEnumerable<PlayerRef> ActivePlayers [get]
```

Returns the collection of [PlayerRef](#) objects for this [NetworkRunner](#)'s [Fusion.Simulation](#).

6.235.4.2 AuthenticationValues

```
AuthenticationValues AuthenticationValues [get]
```

[AuthenticationValues](#) used by this Runner to Authenticate the local peer.

6.235.4.3 BuildType

```
BuildTypes BuildType [static], [get]
```

Get [Fusion.Runtime.dll](#) build type.

6.235.4.4 CanSpawn

```
bool CanSpawn [get]
```

Signal if the Network Runner can spawn a [NetworkObject](#)

6.235.4.5 Config

```
NetworkProjectConfig Config [get]
```

Returns the [NetworkProjectConfig](#) reference.

6.235.4.6 CurrentConnectionType

```
ConnectionType CurrentConnectionType [get]
```

Check the current Connection Type with the Remote Server

6.235.4.7 DeltaTime

```
float DeltaTime [get]
```

Returns the fixed tick time interval. Derived from the [SimulationRuntimeConfig.TickRate](#).

6.235.4.8 GameMode

```
GameMode GameMode [get]
```

Current Game Mode active on the [Fusion Simulation](#)

6.235.4.9 Instances

```
IReadOnlyList<NetworkRunner> Instances [static], [get]
```

A list of all [NetworkRunners](#).

6.235.4.10 IsClient

```
bool IsClient [get]
```

Returns if this [Fusion.Simulation](#) represents a Client connection.

6.235.4.11 IsCloudReady

```
bool IsCloudReady [get]
```

Signal if the Local Peer is connected to Photon Cloud and is able to Create/Join Room but also receive Lobby Updates

6.235.4.12 IsConnectedToServer

```
bool IsConnectedToServer [get]
```

Returns if this Client is currently connected to a Remote Server

6.235.4.13 IsFirstTick

```
bool IsFirstTick [get]
```

If this is the first tick that executes this update or re-simulation

6.235.4.14 IsForward

```
bool IsForward [get]
```

If this is not a re-simulation but a new forward tick

6.235.4.15 IsLastTick

```
bool IsLastTick [get]
```

If this is the last tick that is being executed this update

6.235.4.16 IsPlayer

```
bool IsPlayer [get]
```

Returns true if this runner represents a Client or Host. Dedicated servers have no local player and will return false.

6.235.4.17 IsResimulation

```
bool IsResimulation [get]
```

If we are currently executing a client side prediction re-simulation.

6.235.4.18 IsResume

```
bool IsResume [get]
```

If this instance is a resume (host migration)

6.235.4.19 IsRunning

```
bool IsRunning [get]
```

Returns if this [Fusion.Simulation](#) is valid and running.

6.235.4.20 IsSceneAuthority

```
bool IsSceneAuthority [get]
```

Is this runner responsible for scene management.

6.235.4.21 IsSceneManagerBusy

```
bool? IsSceneManagerBusy [get]
```

Signals if the [INetworkSceneManager](#) instance assigned to this [NetworkRunner](#) is busy with any scene loading operation.

6.235.4.22 IsServer

```
bool IsServer [get]
```

Returns if this [Fusion.Simulation](#) represents a Server connection.

6.235.4.23 IsSharedModeMasterClient

```
bool IsSharedModeMasterClient [get]
```

Signal if the Local Peer is in a Room and is the Room Master Client

6.235.4.24 IsShutdown

```
bool IsShutdown [get]
```

If the runner is shutdown

6.235.4.25 IsSinglePlayer

```
bool IsSinglePlayer [get]
```

Returns true if this runner was started as single player (Started as [SimulationModes.Host](#) with [SimulationConfig.PlayerCount = 1](#)).

6.235.4.26 IsStarting

```
bool IsStarting [get]
```

If the runner is pending to start

6.235.4.27 LagCompensation

```
HitboxManager LagCompensation [get]
```

Returns the global instance of a lag compensation buffer [Fusion.HitboxManager](#).

6.235.4.28 LatestServerTick

```
Tick LatestServerTick [get]
```

Get the latest confirmed tick of the server we are aware of

6.235.4.29 LobbyInfo

```
LobbyInfo LobbyInfo = new LobbyInfo() [get]
```

Signal if the local peer is already inside a Lobby

6.235.4.30 LocalAlpha

```
float LocalAlpha [get]
```

Get the local time alpha value

6.235.4.31 LocalPlayer

```
PlayerRef LocalPlayer [get]
```

Returns a [PlayerRef](#) for the local simulation. For a dedicated server [PlayerRef.IsRealPlayer](#) will equal false. PlayerRefs are assigned in order from 0 to MaxPlayers-1 and are re-used as players join and leave. The only caveat is that the server player (if one exists), always gets the last index no matter how many clients are connected.

6.235.4.32 LocalRenderTime

```
float??? LocalRenderTime [get]
```

The current time (current State.Time + [Simulation.DeltaTime](#)) for predicted objects (objects in the local time frame). Use as an equivalent to Unity's Time.time. Time is relative to [Tick](#) 0 (which represents Time 0f).

6.235.4.33 Mode

```
SimulationModes Mode [get]
```

Returns the [SimulationModes](#) flags for The type of network peer the associated [Fusion.Simulation](#) represents.

6.235.4.34 NATType

`NATType NATType [get]`

Exposesthe current NAT Type from the local Peer

6.235.4.35 ObjectProvider

`INetworkObjectProvider ObjectProvider [get]`

Returns the [INetworkObjectProvider](#) instance.

6.235.4.36 Prefabs

`NetworkPrefabTable Prefabs [get]`

Reference to the [NetworkPrefabTable](#).

6.235.4.37 ProvideInput

`bool ProvideInput [get], [set]`

Indicates if this [NetworkRunner](#) is collecting [PlayerRef INetworkInput](#).

6.235.4.38 RemoteRenderTime

`float RemoteRenderTime [get]`

The current time (current State.Time + [Simulation.DeltaTime](#)) for non-predicted objects (objects in a remote time frame). Use as an equivalent to Unity's Time.time. Time is relative to [Tick 0](#) (which represents Time 0f).

6.235.4.39 SceneManager

`INetworkSceneManager SceneManager [get]`

Returns the [INetworkSceneManager](#) instance.

6.235.4.40 SessionInfo

```
SessionInfo SessionInfo = new SessionInfo() [get]
```

Stores information about the current running session

6.235.4.41 SimulationTime

```
float SimulationTime [get]
```

The time the current State represents (the most recent FixedUpdateNetwork simulation). Use as an equivalent to Unity's Time.fixedTime. Time is relative to [Tick](#) 0 (which represents Time 0f).

6.235.4.42 SimulationUnityScene

```
Scene??? SimulationUnityScene [get]
```

The main scene of the [NetworkRunner](#) or default if not running.

6.235.4.43 Stage

```
SimulationStages Stage [get]
```

Returns the current [SimulationStages](#) stage of this [Fusion.Simulation](#).

6.235.4.44 State

```
States State [get]
```

The current state of the runner, if it's Starting, Running, Shutdown

6.235.4.45 Tick

```
Tick??? Tick [get]
```

The tick associated with the current state of networked objects, or the current simulation tick being processed (when evaluated during FixedUpdateNetwork).

6.235.4.46 TickRate

```
int TickRate [get]
```

6.235.4.47 TicksExecuted

```
int TicksExecuted [get]
```

Returns how many ticks we executed last update.

6.235.4.48 Topology

```
Topologies Topology [get]
```

The current topology used

6.235.4.49 UserId

```
string UserId [get]
```

Photon Client UserID

Returns null if Peer is not connected to Photon Cloud

6.235.5 Event Documentation

6.235.5.1 ObjectAcquired

```
ObjectDelegate ObjectAcquired
```

Event for object acquired

6.236 NetworkRunnerCallbackArgs Class Reference

Stores data types used on the [INetworkRunnerCallbacks](#) interface

Classes

- class [ConnectRequest](#)
Data holder of a Connection Request from a remote client

6.236.1 Detailed Description

Stores data types used on the [INetworkRunnerCallbacks](#) interface

6.237 NetworkRunnerCallbackArgs.ConnectRequest Class Reference

Data holder of a Connection Request from a remote client

Public Member Functions

- void [Accept \(\)](#)
Accepts the Request
- void [Refuse \(\)](#)
Refuses the Request
- void [Waiting \(\)](#)
Refuses the Request

Properties

- [NetAddress RemoteAddress](#) [get, set]
Address of the remote client

6.237.1 Detailed Description

Data holder of a Connection Request from a remote client

6.237.2 Member Function Documentation

6.237.2.1 Accept()

```
void Accept ( )
```

Accepts the Request

6.237.2.2 Refuse()

```
void Refuse ( )
```

Refuses the Request

6.237.2.3 Waiting()

```
void Waiting ( )
```

Refuses the Request

6.237.3 Property Documentation

6.237.3.1 RemoteAddress

`NetAddress` `RemoteAddress` [get], [set]

Address of the remote client

6.238 NetworkRunnerUpdaterDefault Class Reference

Default implementation of [INetworkRunnerUpdater](#) that uses the Unity PlayerLoop.

Inherits [INetworkRunnerUpdater](#).

Classes

- struct `NetworkRunnerRender`
Used to invoke `NetworkRunner.RenderInternal` in the PlayerLoop.
- struct `NetworkRunnerUpdate`
Used to invoke `NetworkRunner.UpdateInternal(double)` in the PlayerLoop.

Static Public Member Functions

- static bool `RegisterInPlayerLoop` (`NetworkRunnerUpdaterDefaultInvokeSettings` `updateSettings`, `NetworkRunnerUpdaterDefaultInvokeSettings` `renderSettings`)
Registers in the PlayerLoop.
- static bool `UnregisterFromPlayerLoop` ()
Unregisters from the PlayerLoop.

Public Attributes

- `NetworkRunnerUpdaterDefaultInvokeSettings RenderSettings`
Default settings for the `NetworkRunner` Render Loop.
- `NetworkRunnerUpdaterDefaultInvokeSettings UpdateSettings`
Default settings for the `NetworkRunner` Update Loop.

Additional Inherited Members

6.238.1 Detailed Description

Default implementation of `INetworkRunnerUpdater` that uses the Unity PlayerLoop.

6.238.2 Member Function Documentation

6.238.2.1 RegisterInPlayerLoop()

```
static bool RegisterInPlayerLoop (
    NetworkRunnerUpdaterDefaultInvokeSettings updateSettings,
    NetworkRunnerUpdaterDefaultInvokeSettings renderSettings ) [static]
```

Registers in the PlayerLoop.

Parameters

<code>updateSettings</code>	Update settings.
<code>renderSettings</code>	Render settings.

Returns

True if registered, false if already registered with the same settings.

6.238.2.2 UnregisterFromPlayerLoop()

```
static bool UnregisterFromPlayerLoop ( ) [static]
```

Unregisters from the PlayerLoop.

Returns

True if unregistered, false if not registered.

6.238.3 Member Data Documentation

6.238.3.1 RenderSettings

```
NetworkRunnerUpdaterDefaultInvokeSettings RenderSettings
```

Initial value:

```
= new NetworkRunnerUpdaterDefaultInvokeSettings {
    ReferencePlayerLoopSystem = typeof(Update.ScriptRunBehaviourUpdate),
    AddMode                  = UnityPlayerLoopSystemAddMode.After
}
```

Default settings for the [NetworkRunner](#) Render Loop.

6.238.3.2 UpdateSettings

```
NetworkRunnerUpdaterDefaultInvokeSettings UpdateSettings
```

Initial value:

```
= new NetworkRunnerUpdaterDefaultInvokeSettings {
    ReferencePlayerLoopSystem = typeof(Update.ScriptRunBehaviourUpdate),
    AddMode                  = UnityPlayerLoopSystemAddMode.Before
}
```

Default settings for the [NetworkRunner](#) Update Loop.

6.239 NetworkRunnerUpdaterDefault.NetworkRunnerRender Struct Reference

Used to invoke [NetworkRunner.RenderInternal](#) in the PlayerLoop.

6.239.1 Detailed Description

Used to invoke [NetworkRunner.RenderInternal](#) in the PlayerLoop.

6.240 NetworkRunnerUpdaterDefault.NetworkRunnerUpdate Struct Reference

Used to invoke [NetworkRunner.UpdateInternal\(double\)](#) in the PlayerLoop.

6.240.1 Detailed Description

Used to invoke [NetworkRunner.UpdateInternal\(double\)](#) in the PlayerLoop.

6.241 NetworkRunnerUpdaterDefaultInvokeSettings Struct Reference

Settings for the [NetworkRunnerUpdaterDefault](#).

Inherits [IEquatable< NetworkRunnerUpdaterDefaultInvokeSettings >](#).

Public Member Functions

- `bool Equals (NetworkRunnerUpdaterDefaultInvokeSettings other)`
Checks if the settings are equal.
- `override bool Equals (object obj)`
Checks if the settings are equal.
- `override int GetHashCode ()`
Gets the hash code of the settings.
- `override string ToString ()`
Returns a string representation of the settings.

Static Public Member Functions

- `static bool operator!= (NetworkRunnerUpdaterDefaultInvokeSettings left, NetworkRunnerUpdaterDefaultInvokeSettings right)`
Checks if the settings are not equal.
- `static bool operator== (NetworkRunnerUpdaterDefaultInvokeSettings left, NetworkRunnerUpdaterDefaultInvokeSettings right)`
Checks if the settings are equal.

Public Attributes

- `UnityPlayerLoopSystemAddMode AddMode`
Add mode for the PlayerLoopSystem.
- `Type ReferencePlayerLoopSystem`
Reference to the PlayerLoopSystem to add the [NetworkRunner](#) to.

6.241.1 Detailed Description

Settings for the [NetworkRunnerUpdaterDefault](#).

6.241.2 Member Function Documentation

6.241.2.1 Equals() [1/2]

```
bool Equals (
    NetworkRunnerUpdaterDefaultInvokeSettings other )
```

Checks if the settings are equal.

Parameters

<i>other</i>	Settings to check for equality.
--------------	---------------------------------

Returns

True if equal, false otherwise.

6.241.2.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Checks if the settings are equal.

Parameters

<i>obj</i>	Settings to check for equality.
------------	---------------------------------

Returns

True if equal, false otherwise.

6.241.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Gets the hash code of the settings.

Returns

Hash code.

6.241.2.4 operator"!=()

```
static bool operator!= (
    NetworkRunnerUpdaterDefaultInvokeSettings left,
    NetworkRunnerUpdaterDefaultInvokeSettings right ) [static]
```

Checks if the settings are not equal.

Parameters

<i>left</i>	Settings to check for equality.
<i>right</i>	Settings to check for equality.

Returns

True if not equal, false otherwise.

6.241.2.5 operator==()

```
static bool operator== (
    NetworkRunnerUpdaterDefaultInvokeSettings left,
    NetworkRunnerUpdaterDefaultInvokeSettings right ) [static]
```

Checks if the settings are equal.

Parameters

<i>left</i>	Settings to check for equality.
<i>right</i>	Settings to check for equality.

Returns

True if equal, false otherwise.

6.241.2.6 ToString()

```
override string ToString ( )
```

Returns a string representation of the settings.

Returns

String representation.

6.241.3 Member Data Documentation

6.241.3.1 AddMode

`UnityPlayerLoopSystemAddMode AddMode`

Add mode for the PlayerLoopSystem.

6.241.3.2 ReferencePlayerLoopSystem

Type `ReferencePlayerLoopSystem`

Reference to the PlayerLoopSystem to add the [NetworkRunner](#) to.

6.242 NetworkSceneAsyncOp Struct Reference

A wrapper for async scene operations.

Inherits `IEnumerator`.

Classes

- struct [Awaiter](#)

Awaiter for NetworkSceneAsyncOp

Public Member Functions

- void [AddOnCompleted](#) (`Action< NetworkSceneAsyncOp > action`)
Adds a callback to be called when the operation is completed
- [Awaiter GetAwaiter](#) ()
Gets the awaiter for the operation
- bool `IEnumerator.MoveNext` ()
- void `IEnumerator.Reset` ()

Static Public Member Functions

- static [NetworkSceneAsyncOp FromAsyncOperation](#) (`SceneRef sceneRef, UnityEngine.AsyncOperation asyncOp`)
Creates a NetworkSceneAsyncOp from a UnityEngine.AsyncOperation
- static [NetworkSceneAsyncOp FromCompleted](#) (`SceneRef sceneRef`)
Creates a completed NetworkSceneAsyncOp
- static [NetworkSceneAsyncOp FromCoroutine](#) (`SceneRef sceneRef, ICoroutine coroutine`)
Creates a NetworkSceneAsyncOp from a ICoroutine
- static [NetworkSceneAsyncOp FromError](#) (`SceneRef sceneRef, Exception error`)
Creates a NetworkSceneAsyncOp from a Exception
- static [NetworkSceneAsyncOp FromTask](#) (`SceneRef sceneRef, Task task`)
Creates a NetworkSceneAsyncOp from a Task

Public Attributes

- readonly [SceneRef SceneRef](#)
The scene reference of the operation

Properties

- object IEnumerator. **Current** [get]
- Exception? **Error** [get]
Attached error to the operation
- bool **IsDone** [get]
Signals if the operation is done
- bool **IsValid** [get]
Signals if the operation is valid

6.242.1 Detailed Description

A wrapper for async scene operations.

6.242.2 Member Function Documentation

6.242.2.1 AddOnCompleted()

```
void AddOnCompleted (
    Action< NetworkSceneAsyncOp > action )
```

Adds a callback to be called when the operation is completed

Parameters

<i>action</i>	The callback to be called
---------------	---------------------------

6.242.2.2 FromAsyncOperation()

```
static NetworkSceneAsyncOp FromAsyncOperation (
    SceneRef sceneRef,
    UnityEngine.AsyncOperation asyncOp ) [static]
```

Creates a [NetworkSceneAsyncOp](#) from a [UnityEngine.AsyncOperation](#)

Parameters

<i>sceneRef</i>	Scene reference
<i>asyncOp</i>	Async operation reference

Generated by Doxygen

Returns

Returns a [NetworkSceneAsyncOp](#) instance

Exceptions

<code>ArgumentNullException</code>	Thrown if <i>asyncOp</i> is null
------------------------------------	----------------------------------

6.242.2.3 FromCompleted()

```
static NetworkSceneAsyncOp FromCompleted (
    SceneRef sceneRef ) [static]
```

Creates a completed [NetworkSceneAsyncOp](#)

Parameters

<code>sceneRef</code>	Scene reference
-----------------------	-----------------

Returns

Returns a [NetworkSceneAsyncOp](#) instance

6.242.2.4 FromCoroutine()

```
static NetworkSceneAsyncOp FromCoroutine (
    SceneRef sceneRef,
    ICoroutine coroutine ) [static]
```

Creates a [NetworkSceneAsyncOp](#) from a [ICoroutine](#)

Parameters

<code>sceneRef</code>	Scene reference
<code>coroutine</code>	Coroutine reference

Returns

Returns a [NetworkSceneAsyncOp](#) instance

Exceptions

<code>ArgumentNullException</code>	Thrown if <i>coroutine</i> is null
------------------------------------	------------------------------------

6.242.2.5 FromError()

```
static NetworkSceneAsyncOp FromError (
    SceneRef sceneRef,
    Exception error ) [static]
```

Creates a [NetworkSceneAsyncOp](#) from a [Exception](#)

Parameters

<i>sceneRef</i>	Scene reference
<i>error</i>	Exception reference

Returns

Returns a [NetworkSceneAsyncOp](#) instance

Exceptions

<i>ArgumentNullException</i>	Thrown if <i>error</i> is null
------------------------------	--------------------------------

6.242.2.6 FromTask()

```
static NetworkSceneAsyncOp FromTask (
    SceneRef sceneRef,
    Task task ) [static]
```

Creates a [NetworkSceneAsyncOp](#) from a [Task](#)

Parameters

<i>sceneRef</i>	Scene reference
<i>task</i>	Task reference

Returns

Returns a [NetworkSceneAsyncOp](#) instance

Exceptions

<i>ArgumentNullException</i>	Thrown if <i>task</i> is null
------------------------------	-------------------------------

6.242.2.7 GetAwaiter()

```
Awaiter GetAwaiter ()
```

Gets the awaiter for the operation

6.242.3 Member Data Documentation

6.242.3.1 SceneRef

```
readonly SceneRef SceneRef
```

The scene reference of the operation

6.242.4 Property Documentation

6.242.4.1 Error

```
Exception? Error [get]
```

Attached error to the operation

6.242.4.2 IsDone

```
bool IsDone [get]
```

Signals if the operation is done

6.242.4.3 IsValid

```
bool IsValid [get]
```

Signals if the operation is valid

6.243 NetworkSceneAsyncOp.Awaiter Struct Reference

Awaiter for NetworkSceneAsyncOp

Inherits INotifyCompletion.

Public Member Functions

- `Awaiter (in NetworkSceneAsyncOp op)`
Creates a new Awaiter instance
- `void GetResult ()`
Gets the result of the operation
- `void OnCompleted (Action continuation)`
Adds a callback to be called when the operation is completed

Public Attributes

- `NetworkSceneAsyncOp _op`

Properties

- `bool IsCompleted [get]`
Signals if the operation is completed

6.243.1 Detailed Description

Awaiter for NetworkSceneAsyncOp

6.243.2 Constructor & Destructor Documentation

6.243.2.1 Awaiter()

```
Awaiter (
    in NetworkSceneAsyncOp op )
```

Creates a new Awaiter instance

Parameters

<code>op</code>	The operation to await
-----------------	------------------------

6.243.3 Member Function Documentation

6.243.3.1 GetResult()

```
void GetResult ( )
```

Gets the result of the operation

6.243.3.2 OnCompleted()

```
void OnCompleted (
    Action continuation )
```

Adds a callback to be called when the operation is completed

Parameters

<i>continuation</i>	The callback to be called
---------------------	---------------------------

6.243.4 Property Documentation

6.243.4.1 IsCompleted

```
bool IsCompleted [get]
```

Signals if the operation is completed

6.244 NetworkSceneInfo Struct Reference

Can store up to 8 active scenes and allows for duplicates. Each write increases [Version](#) which can be used to generate unique scene objects ids for when a scene is supposed to be reloaded.

Inherits [INetworkStruct](#), and [IEquatable< NetworkSceneInfo >](#).

Public Member Functions

- int [AddSceneRef](#) ([SceneRef](#) sceneRef, [LoadSceneMode](#) loadSceneMode=[LoadSceneMode.Single](#), [LocalPhysicsMode](#) localPhysicsMode=[LocalPhysicsMode.None](#), bool activeOnLoad=false)
Adds a scene to the list
- bool [Equals](#) ([NetworkSceneInfo](#) other)
Compares two [NetworkSceneInfo](#) for equality
- override bool [Equals](#) (object obj)
Compares two [NetworkSceneInfo](#) for equality
- override int [GetHashCode](#) ()
Get the hash code of the [NetworkSceneInfo](#)
- int [IndexOf](#) (([SceneRef](#) SceneRef, [NetworkLoadSceneParameters](#) SceneParams) scene)
• int [IndexOf](#) ([SceneRef](#) sceneRef, [NetworkLoadSceneParameters](#) sceneParams)
Gets the index of the given scene
- bool [RemoveSceneRef](#) ([SceneRef](#) sceneRef)
Removes a scene from the list
- override string [ToString](#) ()
String representation of the [NetworkSceneInfo](#)

Static Public Member Functions

- static implicit operator [NetworkSceneInfo](#) ([SceneRef](#) sceneRef)
Implicit conversion to [NetworkSceneInfo](#)

Static Public Attributes

- const int [MaxScenes](#) = 8
Max number of scenes that can be stored
- const int [SIZE](#) = 52
The size of the struct in bytes
- const int [WORD_COUNT](#) = 13
The size of the struct in words

Properties

- int [SceneCount](#) [get]
Total Scene Count
- [FixedSize](#)< [NetworkLoadSceneParameters](#) > [SceneParams](#) [get]
The scenes load parameters list
- [FixedSize](#)< [SceneRef](#) > [Scenes](#) [get]
The scenes list
- int [Version](#) [get]
Version number

6.244.1 Detailed Description

Can store up to 8 active scenes and allows for duplicates. Each write increases [Version](#) which can be used to generate unique scene objects ids for when a scene is supposed to be reloaded.

6.244.2 Member Function Documentation

6.244.2.1 AddSceneRef()

```
int AddSceneRef (
    SceneRef sceneRef,
    LoadSceneMode loadSceneMode = LoadSceneMode.Single,
    LocalPhysicsMode localPhysicsMode = LocalPhysicsMode.None,
    bool activeOnLoad = false )
```

Adds a scene to the list

Parameters

<i>sceneRef</i>	Scene to add
<i>loadSceneMode</i>	Load scene mode
<i>localPhysicsMode</i>	Local physics mode
<i>activeOnLoad</i>	Signals if the scene should be active on load

Returns

Returns the index of the scene or -1 if the scene could not be added

6.244.2.2 Equals() [1/2]

```
bool Equals (
    NetworkSceneInfo other )
```

Compares two [NetworkSceneInfo](#) for equality

Parameters

<i>other</i>	The other NetworkSceneInfo
--------------	--

Returns

Returns true if the two [NetworkSceneInfo](#) are equal

6.244.2.3 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Compares two [NetworkSceneInfo](#) for equality

Parameters

<i>obj</i>	The other NetworkSceneInfo
------------	--

Returns

Returns true if the two [NetworkSceneInfo](#) are equal

6.244.2.4 GetHashCode()

```
override int GetHashCode ( )
```

Get the hash code of the [NetworkSceneInfo](#)

Returns

Hash code of the [NetworkSceneInfo](#)

6.244.2.5 IndexOf()

```
int IndexOf (
    SceneRef sceneRef,
    NetworkLoadSceneParameters sceneParams )
```

Gets the index of the given scene

Parameters

<i>sceneRef</i>	SceneRef to look for
<i>sceneParams</i>	Scene parameters to look for

Returns

Returns the index of the scene or -1 if not found

6.244.2.6 operator NetworkSceneInfo()

```
static implicit operator NetworkSceneInfo (
    SceneRef sceneRef ) [static]
```

Implicit conversion to [NetworkSceneInfo](#)

Parameters

<code>sceneRef</code>	SceneRef to convert
-----------------------	---------------------

Returns

Returns a [NetworkSceneInfo](#) instance

6.244.2.7 RemoveSceneRef()

```
bool RemoveSceneRef (
    SceneRef sceneRef )
```

Removes a scene from the list

Parameters

<code>sceneRef</code>	Scene to remove
-----------------------	-----------------

Returns

Returns true if the scene was removed

6.244.2.8 ToString()

```
override string ToString ( )
```

String representation of the [NetworkSceneInfo](#)

6.244.3 Member Data Documentation**6.244.3.1 MaxScenes**

```
const int MaxScenes = 8 [static]
```

Max number of scenes that can be stored

6.244.3.2 SIZE

```
const int SIZE = 52 [static]
```

The size of the struct in bytes

6.244.3.3 WORD_COUNT

```
const int WORD_COUNT = 13 [static]
```

The size of the struct in words

6.244.4 Property Documentation

6.244.4.1 SceneCount

```
int SceneCount [get]
```

Total Scene Count

6.244.4.2 SceneParams

```
FixedArray<NetworkLoadSceneParameters> SceneParams [get]
```

The scenes load parameters list

6.244.4.3 Scenes

```
FixedArray<SceneRef> Scenes [get]
```

The scenes list

6.244.4.4 Version

```
int Version [get]
```

Version number

6.245 NetworkSceneLoadId Struct Reference

A unique identifier for a scene load operation.

Inherits IEquatable< NetworkSceneLoadId >.

Public Member Functions

- bool Equals (NetworkSceneLoadId other)
Compares two NetworkSceneLoadId for equality
- override bool Equals (object obj)
Compares two NetworkSceneLoadId for equality
- override int GetHashCode ()
Returns the hash code of the NetworkSceneLoadId
- NetworkSceneLoadId (byte value)
Creates a new NetworkSceneLoadId with the given value

Public Attributes

- readonly byte Value
The value of the id

6.245.1 Detailed Description

A unique identifier for a scene load operation.

6.245.2 Constructor & Destructor Documentation

6.245.2.1 NetworkSceneLoadId()

```
NetworkSceneLoadId (
    byte value )
```

Creates a new NetworkSceneLoadId with the given value

Parameters

value	The value of the id
-------	---------------------

6.245.3 Member Function Documentation

6.245.3.1 Equals() [1/2]

```
bool Equals (  
    NetworkSceneLoadId other )
```

Compares two [NetworkSceneLoadId](#) for equality

Parameters

<i>other</i>	The other NetworkSceneLoadId
--------------	--

Returns

Returns true if the two [NetworkSceneLoadId](#) are equal

6.245.3.2 Equals() [2/2]

```
override bool Equals (  
    object obj )
```

Compares two [NetworkSceneLoadId](#) for equality

Parameters

<i>obj</i>	The other object to check
------------	---------------------------

Returns

Returns true if the two [NetworkSceneLoadId](#) are equal

6.245.3.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code of the [NetworkSceneLoadId](#)

6.245.4 Member Data Documentation

6.245.4.1 Value

```
readonly byte Value
```

The value of the id

6.246 NetworkSceneObjectId Struct Reference

A unique identifier for a scene object.

Inherits IEquatable< NetworkSceneObjectId >.

Public Member Functions

- bool [Equals \(NetworkSceneObjectId other\)](#)
Check if two NetworkSceneObjectId are equal.
- override bool [Equals \(object obj\)](#)
Check if two NetworkSceneObjectId are equal.
- override int [GetHashCode \(\)](#)
Get the hash code of the NetworkSceneObjectId.
- override string [ToString \(\)](#)
String representation of the NetworkSceneObjectId.

Public Attributes

- int [ObjectId](#)
Index of the object in the scene or any other form of unique identifier.
- [SceneRef Scene](#)
Identifies the scene in which the object is located.
- int [SceneLoadId](#)
Unique identifier of a specific scene load. Needs to be used when loading multiple scenes with the same SceneRef or reloading a scene. For example, NetworkSceneInfo increments its internal LoadId every time a new scene is added.

Properties

- bool [IsValid \[get\]](#)
Signal if the NetworkSceneObjectId is valid.

6.246.1 Detailed Description

A unique identifier for a scene object.

6.246.2 Member Function Documentation

6.246.2.1 Equals() [1/2]

```
bool Equals (
    NetworkSceneObjectId other )
```

Check if two NetworkSceneObjectId are equal.

Parameters

<i>other</i>	Another NetworkSceneObjectId to check for equality
--------------	--

Returns

Returns true if the two [NetworkSceneObjectId](#) are equal

6.246.2.2 Equals() [2/2]

```
override bool Equals (
    object obj )
```

Check if two [NetworkSceneObjectId](#) are equal.

Parameters

<i>obj</i>	Another NetworkSceneObjectId to check for equality
------------	--

Returns

Returns true if the two [NetworkSceneObjectId](#) are equal

6.246.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Get the hash code of the [NetworkSceneObjectId](#).

6.246.2.4 ToString()

```
override string ToString ( )
```

String representation of the [NetworkSceneObjectId](#).

6.246.3 Member Data Documentation

6.246.3.1 ObjectId

```
int ObjectId
```

Index of the object in the scene or any other form of unique identifier.

6.246.3.2 Scene

```
SceneRef Scene
```

Identifies the scene in which the object is located.

6.246.3.3 SceneLoadId

```
int SceneLoadId
```

Unique identifier of a specific scene load. Needs to be used when loading multiple scenes with the same [SceneRef](#) or reloading a scene. For example, [NetworkSceneInfo](#) increments its internal LoadId every time a new scene is added.

6.246.4 Property Documentation

6.246.4.1 IsValid

```
bool IsValid [get]
```

Signal if the [NetworkSceneObjectId](#) is valid.

6.247 NetworkSerializeMethodAttribute Class Reference

Network Serialize Method Attribute

Inherits Attribute.

Properties

- int [MaxSize](#) [get, set]

If set, this changes expected Wrap method signature to int Name(NetworkRunner, T, byte) and Unwrap to int Name(NetworkRunner, byte*, ref T). In both cases, the result is the number of bytes written/read and can not be greater than what's declared here.*

6.247.1 Detailed Description

Network Serialize Method Attribute

6.247.2 Property Documentation

6.247.2.1 MaxSize

```
int MaxSize [get], [set]
```

If set, this changes expected Wrap method signature to int Name(NetworkRunner, T, byte*) and Unwrap to int Name(NetworkRunner, byte*, ref T). In both cases, the result is the number of bytes written/read and can not be greater than what's declared here.

6.248 NetworkSimulationConfiguration Class Reference

Configuration for network conditions simulation (induced latency and loss).

Public Member Functions

- [NetworkSimulationConfiguration Clone \(\)](#)
Creates a copy of this NetworkSimulationConfiguration.
- [NetConfigSimulation Create \(\)](#)
Creates a new NetConfigSimulation based on the current configuration.

Public Attributes

- double [AdditionalJitter](#) = 0.05
After the delay value from the DelayShape oscillator is determined, random 0 to this value of additional seconds be added to the packet latency.
- double [AdditionalLoss](#) = 0
After the LossChanceShape oscillation loss chance is calculated, an additional random value of 0 to this (normalized) percentage of loss chance is added.
- double [DelayMax](#) = 0.15
The highest packet delay value returned from the DelayShape oscillator.
- double [DelayMin](#) = 0.15
The lowest packet delay value returned from the DelayShape oscillator.
- double [DelayPeriod](#) = 0
The period of the DelayShape oscillator (the rate at which delay oscillates in seconds).
- NetConfigSimulationOscillator.WaveShape [DelayShape](#) = NetConfigSimulationOscillator.WaveShape.Noise
The pattern used to oscillate between DelayMin and DelayMax values.
- double [DelayThreshold](#) = 0
The DelayShape oscillates between 0 and 1. Values below this threshold are reduced to zero, resulting in a value equal to DelayMin.

- bool `Enabled`
If adverse network conditions are being simulated.
- double `LossChanceMax` = 0.05
The highest loss chance value the `LossChanceShape` oscillator will produce. 0 = 0% chance of being lost. 1 = 100% chance of being lost.
- double `LossChanceMin` = 0.05
The lowest loss chance value the `LossChanceShape` oscillator will produce. 0 = 0% chance of being lost. 1 = 100% chance of being lost.
- double `LossChancePeriod` = 0
The period of the `LossChanceShape` oscillator (the rate at which delay oscillates between `LossChanceMin` and `LossChanceMax`).
- NetConfigSimulationOscillator.WaveShape `LossChanceShape` = NetConfigSimulationOscillator.WaveShape.Noise
The pattern used to oscillate between `LossChanceMin` and `LossChanceMax` values.
- double `LossChanceThreshold` = 0
The `LossChanceShape` wave oscillates between 0 and 1. Values below this threshold are reduced to zero, resulting in a value equal to `LossChanceMin`.

6.248.1 Detailed Description

Configuration for network conditions simulation (induced latency and loss).

6.248.2 Member Function Documentation

6.248.2.1 `Clone()`

```
NetworkSimulationConfiguration Clone ( )
```

Creates a copy of this `NetworkSimulationConfiguration`.

6.248.2.2 `Create()`

```
NetConfigSimulation Create ( )
```

Creates a new `NetConfigSimulation` based on the current configuration.

Returns

A new `NetConfigSimulation` based on the current configuration.

6.248.3 Member Data Documentation

6.248.3.1 AdditionalJitter

```
double AdditionalJitter = 0.05
```

After the delay value from the [DelayShape](#) oscillator is determined, random 0 to this value of additional seconds be added to the packet latency.

6.248.3.2 AdditionalLoss

```
double AdditionalLoss = 0
```

After the [LossChanceShape](#) oscillation loss chance is calculated, an additional random value of 0 to this (normalized) percentage of loss chance is added.

6.248.3.3 DelayMax

```
double DelayMax = 0.15
```

The highest packet delay value returned from the [DelayShape](#) oscillator.

6.248.3.4 DelayMin

```
double DelayMin = 0.15
```

The lowest packet delay value returned from the [DelayShape](#) oscillator.

6.248.3.5 DelayPeriod

```
double DelayPeriod = 0
```

The period of the [DelayShape](#) oscillator (the rate at which delay oscillates in seconds).

6.248.3.6 DelayShape

```
NetConfigSimulationOscillator.WaveShape DelayShape = NetConfigSimulationOscillator.Wave←  
Shape.Noise
```

The pattern used to oscillate between [DelayMin](#) and [DelayMax](#) values.

6.248.3.7 DelayThreshold

```
double DelayThreshold = 0
```

The [DelayShape](#) oscillates between 0 and 1. Values below this threshold are reduced to zero, resulting in a value equal to [DelayMin](#).

6.248.3.8 Enabled

```
bool Enabled
```

If adverse network conditions are being simulated.

6.248.3.9 LossChanceMax

```
double LossChanceMax = 0.05
```

The highest loss chance value the [LossChanceShape](#) oscillator will produce. 0 = 0% chance of being lost. 1 = 100% chance of being lost.

6.248.3.10 LossChanceMin

```
double LossChanceMin = 0.05
```

The lowest loss chance value the [LossChanceShape](#) oscillator will produce. 0 = 0% chance of being lost. 1 = 100% chance of being lost.

6.248.3.11 LossChancePeriod

```
double LossChancePeriod = 0
```

The period of the [LossChanceShape](#) oscillator (the rate at which delay oscillates between [LossChanceMin](#) and [LossChanceMax](#)).

6.248.3.12 LossChanceShape

```
NetConfigSimulationOscillator.WaveShape LossChanceShape = NetConfigSimulationOscillator.WaveShape.Noise
```

The pattern used to oscillate between [LossChanceMin](#) and [LossChanceMax](#) values.

6.248.3.13 LossChanceThreshold

```
double LossChanceThreshold = 0
```

The [LossChanceShape](#) wave oscillates between 0 and 1. Values below this threshold are reduced to zero, resulting in a value equal to [LossChanceMin](#).

6.249 NetworkSpawnOp Struct Reference

Spawn Operation

Classes

- struct [Awaiter](#)
Awaiter for NetworkSpawnOp

Public Member Functions

- [Awaiter GetAwaiter \(\)](#)
Get this Spawn Operation Awaiter

Public Attributes

- readonly [NetworkRunner Runner](#)
Network Runner Reference

Properties

- bool [IsFailed \[get\]](#)
Returns true if the object has failed to spawn.
- bool [IsQueued \[get\]](#)
Returns true if the object is still queued for spawning.
- bool [IsSpawned \[get\]](#)
Returns true if the object has been spawned.
- [NetworkObject Object \[get\]](#)
Get the spawned Network Object
- [NetworkSpawnStatus Status \[get\]](#)
Get the Spawn Operation Status

6.249.1 Detailed Description

Spawn Operation

6.249.2 Member Function Documentation

6.249.2.1 GetAwaiter()

```
Awaiter GetAwaiter ()
```

Get this Spawn Operation [Awaiter](#)

6.249.3 Member Data Documentation

6.249.3.1 Runner

```
readonly NetworkRunner Runner
```

Network Runner Reference

6.249.4 Property Documentation

6.249.4.1 IsFailed

```
bool IsFailed [get]
```

Returns true if the object has failed to spawn.

6.249.4.2 IsQueued

```
bool IsQueued [get]
```

Returns true if the object is still queued for spawning.

6.249.4.3 IsSpawned

```
bool IsSpawned [get]
```

Returns true if the object has been spawned.

6.249.4.4 Object

`NetworkObject Object [get]`

Get the spawned Network Object

6.249.4.5 Status

`NetworkSpawnStatus Status [get]`

Get the Spawn Operation Status

6.250 NetworkSpawnOp.Awaiter Struct Reference

Awaiter for `NetworkSpawnOp`

Inherits `INotifyCompletion`.

Public Member Functions

- `Awaiter` (in `NetworkSpawnOp op`)
Awaiter Constructor
- `NetworkObject GetResult ()`
Get the result of the Spawn Operation
- `void OnCompleted (Action continuation)`
Awaiter OnCompleted Callback

Public Attributes

- `NetworkSpawnOp _op`

Properties

- `bool IsCompleted [get]`
Returns true if the Spawn Operation is completed

6.250.1 Detailed Description

Awaiter for `NetworkSpawnOp`

6.250.2 Constructor & Destructor Documentation

6.250.2.1 Awaiter()

`Awaiter (`
 `in NetworkSpawnOp op)`

`Awaiter` Constructor

Parameters

<i>op</i>	Spawn Operation
-----------	-----------------

6.250.3 Member Function Documentation

6.250.3.1 GetResult()

`NetworkObject GetResult ()`

Get the result of the Spawn Operation

Returns

Spawned Network Object

Exceptions

<code>NetworkObjectSpawnException</code>	Thrown if the Spawn Operation failed
--	--------------------------------------

6.250.3.2 OnCompleted()

`void OnCompleted (`
 Action continuation `)`

`Awaiter` OnCompleted Callback

Parameters

<i>continuation</i>	Continuation Action
---------------------	---------------------

Exceptions

<code>NotSupportedException</code>	Thrown if the Spawn Operation is not supported
------------------------------------	--

6.250.4 Property Documentation

6.250.4.1 IsCompleted

```
bool IsCompleted [get]
```

Returns true if the Spawn Operation is completed

6.251 NetworkString< TSize > Class Template Reference

Fixed-size UTF32 string. All operations are alloc-free, except for converting to System.String.

Inherits INetworkString, INetworkStruct, IEquatable< NetworkString< TSize >>, and IEnumerable< char >.

Public Member Functions

- void [Assign](#) (string value)
Assign a new value to this NetworkString.
- int [Compare](#) (NetworkString< TSize > s)
Compares this instance with a specified NetworkString.
- int [Compare](#) (ref NetworkString< TSize > s)
Compares this instance with a specified NetworkString.
- int [Compare](#) (string s)
Compares this instance with a specified string.
- int [Compare< TOtherSize >](#) (NetworkString< TOtherSize > other)
Compares this instance with a specified NetworkString of a different size.
- int [Compare< TOtherSize >](#) (ref NetworkString< TOtherSize > other)
Compares this instance with a specified NetworkString of a different size.
- bool [Contains](#) (char c)
Determines whether a specified character is in this instance.
- bool [Contains](#) (string str)
Determines whether a specified string is in this instance.
- bool [Contains](#) (uint codePoint)
Determines whether a specified Unicode code point is in this instance.
- bool [Contains< TOtherSize >](#) (NetworkString< TOtherSize > str)
Determines whether a specified NetworkString is in this instance.
- bool [Contains< TOtherSize >](#) (ref NetworkString< TOtherSize > str)
Determines whether a specified NetworkString is in this instance.
- bool [EndsWith](#) (string s)
Checks if the current NetworkString ends with a specified string.
- bool [EndsWith< TOtherSize >](#) (ref NetworkString< TOtherSize > other)
Checks if the current NetworkString ends with a specified NetworkString of a different size.
- bool [Equals](#) (NetworkString< TSize > other)
Determines whether the current NetworkString is equal to a specified NetworkString.
- override bool [Equals](#) (object obj)
Determines whether the current NetworkString is equal to a specified object.
- bool [Equals](#) (ref NetworkString< TSize > other)
Determines whether the current NetworkString is equal to a specified NetworkString.
- bool [Equals](#) (string s)
Determines whether the current NetworkString is equal to a specified string.
- bool [Equals< TOtherSize >](#) (NetworkString< TOtherSize > other)
Determines whether the current NetworkString is equal to a specified NetworkString of a different size.

- **Determines whether the current `NetworkString` is equal to a specified `NetworkString` of a different size.**
- bool `Equals< TOtherSize >` (ref `NetworkString< TOtherSize >` other)

Determines whether the current `NetworkString` is equal to a specified `NetworkString` of a different size.
- bool `Get` (ref string cache)

Checks if cache is equivalent and if not converts to UTF16 and stores the result in cache .
- int `GetCharCount ()`

Calculates the length of the equivalent UTF16 string.
- `UTF32Tools.CharEnumerator GetEnumerator ()`

Returns an enumerator that iterates through the `NetworkString`.
- `IEnumerator< char > IEnumerable< char >. GetEnumerator ()`
- `IEnumerator IEnumerable. GetEnumerator ()`
- override int `GetHashCode ()`

Returns the hash code for this `NetworkString`.
- int `IndexOf (char c, int startIndex, int count)`

Returns the index of the first occurrence of a specified character in this instance.
- int `IndexOf (char c, int startIndex=0)`

Returns the index of the first occurrence of a specified character in this instance.
- int `IndexOf (string str, int startIndex, int count)`

Returns the index of the first occurrence of a specified string in this instance.
- int `IndexOf (string str, int startIndex=0)`

Returns the index of the first occurrence of a specified string in this instance.
- int `IndexOf (uint codePoint, int startIndex, int count)`

Returns the index of the first occurrence of a specified Unicode code point in this instance.
- int `IndexOf (uint codePoint, int startIndex=0)`

Returns the index of the first occurrence of a specified Unicode code point in this instance.
- int `IndexOf< TOtherSize > (NetworkString< TOtherSize > str, int startIndex, int count)`

Returns the index of the first occurrence of a specified `NetworkString` in this instance.
- int `IndexOf< TOtherSize > (NetworkString< TOtherSize > str, int startIndex=0)`

Returns the index of the first occurrence of a specified `NetworkString` in this instance.
- int `IndexOf< TOtherSize > (ref NetworkString< TOtherSize > str, int startIndex, int count)`

Returns the index of the first occurrence of a specified `NetworkString` in this instance.
- int `IndexOf< TOtherSize > (ref NetworkString< TOtherSize > str, int startIndex=0)`

Returns the index of the first occurrence of a specified `NetworkString` in this instance.
- **`NetworkString` (string value)**

Creates a new instance of `NetworkString< Size >` with the given value.
- bool `Set (string value)`

Converts value to UTF32 string and stores it internally.
- bool `StartsWith (string s)`

Checks if the current `NetworkString` starts with a specified string.
- bool `StartsWith< TOtherSize > (ref NetworkString< TOtherSize > other)`

Checks if the current `NetworkString` starts with a specified `NetworkString` of a different size.
- `NetworkString< TSize > Substring (int startIndex)`

Returns a substring from this instance. The substring starts at a specified character position.
- `NetworkString< TSize > Substring (int startIndex, int length)`

Returns a substring from this instance. The substring starts at a specified character position and has a specified length.
- `NetworkString< TSize > ToLower ()`

Converts all the characters in this `NetworkString` to lowercase.
- override string `ToString ()`

Converts the value of this `NetworkString` to its equivalent string representation.
- `NetworkString< TSize > ToUpper ()`

Converts all the characters in this `NetworkString` to uppercase.

Static Public Member Functions

- static int `GetCapacity< TSize > ()`
Gets the capacity of a [NetworkString](#) of a specified size.
- static implicit operator `NetworkString< TSize > (string str)`
Defines an implicit conversion of a string to a [NetworkString](#).
- static operator `string (NetworkString< TSize > str)`
Defines an explicit conversion of a [NetworkString](#) to a string.
- static bool `operator!= (NetworkString< TSize > a, NetworkString< TSize > b)`
Defines an inequality operator for [NetworkString](#).
- static bool `operator!= (NetworkString< TSize > a, string b)`
Defines an inequality operator for a [NetworkString](#) and a string.
- static bool `operator!= (string a, NetworkString< TSize > b)`
Defines an inequality operator for a string and a [NetworkString](#).
- static bool `operator== (NetworkString< TSize > a, NetworkString< TSize > b)`
Defines an equality operator for [NetworkString](#).
- static bool `operator== (NetworkString< TSize > a, string b)`
Defines an equality operator for a [NetworkString](#) and a string.
- static bool `operator== (string a, NetworkString< TSize > b)`
Defines an equality operator for a string and a [NetworkString](#).

Properties

- int `Capacity [get]`
Maximum UTF32 string length.
- int `Length [get]`
Number of UTF32 scalars. It is equal or less than [GetCharCount](#) or the length of [Value](#), because those use UTF16 encoding, which needs two characters to encode some values.
- ref uint `this[int index] [get]`
Returns UTF32 scalar at index position. To iterate over characters, use [GetEnumerator](#).
- string `Value [get, set]`
Converts to/from regular UTF16 string. Setter is alloc-free. Use [Get](#) to get possibly alloc-free conversion.

6.251.1 Detailed Description

Fixed-size UTF32 string. All operations are alloc-free, except for converting to System.String.

Provides static methods for [NetworkString](#) operations.

Template Parameters

<code>TSize</code>	
--------------------	--

Type Constraints

- `TSize : unmanaged`**
- `TSize : IFixedStorage`**

6.251.2 Constructor & Destructor Documentation

6.251.2.1 NetworkString()

```
NetworkString (
    string value )
```

Creates a new instance of [NetworkString<Size>](#) with the given value.

Parameters

<code>value</code>	String value.
--------------------	---------------

6.251.3 Member Function Documentation

6.251.3.1 Assign()

```
void Assign (
    string value )
```

Assign a new value to this [NetworkString](#).

Parameters

<code>value</code>	String value.
--------------------	---------------

6.251.3.2 Compare() [1/3]

```
int Compare (
    NetworkString< TSize > s )
```

Compares this instance with a specified [NetworkString](#).

Parameters

<code>s</code>	The NetworkString to compare.
----------------	---

Returns

A 32-bit signed integer that indicates the comparison result.

6.251.3.3 Compare() [2/3]

```
int Compare (
    ref NetworkString< TSize > s )
```

Compares this instance with a specified [NetworkString](#).

Parameters

s	The NetworkString to compare.
---	---

Returns

A 32-bit signed integer that indicates the comparison result.

6.251.3.4 Compare() [3/3]

```
int Compare (
    string s )
```

Compares this instance with a specified string.

Parameters

s	The string to compare.
---	------------------------

Returns

A 32-bit signed integer that indicates the comparison result.

6.251.3.5 Compare< TOtherSize >() [1/2]

```
int Compare< TOtherSize > (
    NetworkString< TOtherSize > other )
```

Compares this instance with a specified [NetworkString](#) of a different size.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>other</i>	The NetworkString to compare.
--------------	---

Returns

A 32-bit signed integer that indicates the comparison result.

Type Constraints

TOtherSize : *unmanaged*
TOtherSize : *IFixedStorage*
TOtherSize : *Compare*
TOtherSize : *ref*
TOtherSize : *other*

6.251.3.6 Compare< TOtherSize >() [2/2]

```
int Compare< TOtherSize > (
    ref NetworkString< TOtherSize > other )
```

Compares this instance with a specified [NetworkString](#) of a different size.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>other</i>	The NetworkString to compare.
--------------	---

Returns

A 32-bit signed integer that indicates the comparison result.

Type Constraints

TOtherSize : *unmanaged*
TOtherSize : *IFixedStorage*

6.251.3.7 Contains() [1/3]

```
bool Contains (
    char c )
```

Determines whether a specified character is in this instance.

Parameters

<i>c</i>	The Unicode character to seek.
----------	--------------------------------

Returns

true if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

6.251.3.8 Contains() [2/3]

```
bool Contains (
    string str )
```

Determines whether a specified string is in this instance.

Parameters

<i>str</i>	The string to seek.
------------	---------------------

Returns

true if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

6.251.3.9 Contains() [3/3]

```
bool Contains (
    uint codePoint )
```

Determines whether a specified Unicode code point is in this instance.

Parameters

<i>codePoint</i>	The Unicode code point to seek.
------------------	---------------------------------

Returns

true if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

6.251.3.10 Contains< TOtherSize >() [1/2]

```
bool Contains< TOtherSize > (
    NetworkString< TOtherSize > str )
```

Determines whether a specified [NetworkString](#) is in this instance.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>str</i>	The NetworkString to seek.
------------	--

Returns

true if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

Type Constraints

TOtherSize : *unmanaged*
TOtherSize : *IFixedStorage*
TOtherSize : *IndexOf*
TOtherSize : *ref*
TOtherSize : *str*

6.251.3.11 Contains< TOtherSize >() [2/2]

```
bool Contains< TOtherSize > (
    ref NetworkString< TOtherSize > str )
```

Determines whether a specified [NetworkString](#) is in this instance.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>str</i>	The NetworkString to seek.
------------	--

Returns

true if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

Type Constraints

TOtherSize : *unmanaged*
TOtherSize : *IFixedStorage*
TOtherSize : *IndexOf*
TOtherSize : *ref*
TOtherSize : *str*

6.251.3.12 EndsWith()

```
bool EndsWith (
    string s )
```

Checks if the current [NetworkString](#) ends with a specified string.

Parameters

<i>s</i>	The string to check.
----------	----------------------

Returns

true if the current [NetworkString](#) ends with the specified string; otherwise, false.

Exceptions

<i>ArgumentNullException</i>	Thrown when the string is null.
------------------------------	---------------------------------

6.251.3.13 EndsWith< TOtherSize >()

```
bool EndsWith< TOtherSize > (
    ref NetworkString< TOtherSize > other )
```

Checks if the current [NetworkString](#) ends with a specified [NetworkString](#) of a different size.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>other</i>	The NetworkString to check.
--------------	---

Returns

true if the current [NetworkString](#) ends with the specified [NetworkString](#); otherwise, false.

Type Constraints

TOtherSize : *unmanaged*
TOtherSize : *IFixedStorage*

6.251.3.14 Equals() [1/4]

```
bool Equals (   
    NetworkString< TSize > other )
```

Determines whether the current [NetworkString](#) is equal to a specified [NetworkString](#).

Parameters

<i>other</i>	The NetworkString to compare with the current NetworkString .
--------------	---

Returns

true if the specified [NetworkString](#) is equal to the current [NetworkString](#); otherwise, false.

6.251.3.15 Equals() [2/4]

```
override bool Equals (   
    object obj )
```

Determines whether the current [NetworkString](#) is equal to a specified object.

Parameters

<i>obj</i>	The object to compare with the current NetworkString .
------------	--

Returns

true if the specified object is equal to the current [NetworkString](#); otherwise, false.

6.251.3.16 Equals() [3/4]

```
bool Equals (
    ref NetworkString< TSize > other )
```

Determines whether the current [NetworkString](#) is equal to a specified [NetworkString](#).

Parameters

<i>other</i>	The NetworkString to compare with the current NetworkString .
--------------	---

Returns

true if the specified [NetworkString](#) is equal to the current [NetworkString](#); otherwise, false.

6.251.3.17 Equals() [4/4]

```
bool Equals (
    string s )
```

Determines whether the current [NetworkString](#) is equal to a specified string.

Parameters

<i>s</i>	The string to compare with the current NetworkString .
----------	--

Returns

true if the specified string is equal to the current [NetworkString](#); otherwise, false.

6.251.3.18 Equals< TOtherSize >() [1/2]

```
bool Equals< TOtherSize > (
    NetworkString< TOtherSize > other )
```

Determines whether the current [NetworkString](#) is equal to a specified [NetworkString](#) of a different size.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>other</i>	The NetworkString to compare with the current NetworkString .
--------------	---

Returns

true if the specified [NetworkString](#) is equal to the current [NetworkString](#); otherwise, false.

Type Constraints

TOtherSize : *unmanaged*
TOtherSize : *IFixedStorage*
TOtherSize : *Compare*
TOtherSize : *ref*
TOtherSize : *other*

6.251.3.19 Equals< TOtherSize >() [2/2]

```
bool Equals< TOtherSize > (
    ref NetworkString< TOtherSize > other )
```

Determines whether the current [NetworkString](#) is equal to a specified [NetworkString](#) of a different size.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>other</i>	The NetworkString to compare with the current NetworkString .
--------------	---

Returns

true if the specified [NetworkString](#) is equal to the current [NetworkString](#); otherwise, false.

Type Constraints

TOtherSize : *unmanaged*
TOtherSize : *IFixedStorage*
TOtherSize : *Compare*
TOtherSize : *ref*
TOtherSize : *other*

6.251.3.20 Get()

```
bool Get (
    ref string cache )
```

Checks if *cache* is equivalent and if not converts to UTF16 and stores the result in *cache*.

Parameters

<code>cache</code>	The string to convert.
--------------------	------------------------

Returns

False if no conversion was performed, true otherwise.

6.251.3.21 GetCapacity< TSize >()

```
static int GetCapacity< TSize > ( ) [static]
```

Gets the capacity of a [NetworkString](#) of a specified size.

Template Parameters

<code>TSize</code>	The size of the NetworkString .
--------------------	---

Returns

The capacity of a [NetworkString](#) of the specified size.

Type Constraints

`TSize : unmanaged`

`TSize : IFixedStorage`

6.251.3.22 GetCharCount()

```
int GetCharCount ( )
```

Calculates the length of the equivalent UTF16 string.

Returns

The length of the equivalent UTF16 string.

6.251.3.23 GetEnumerator()

```
UTF32Tools.CharEnumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the [NetworkString](#).

Returns

A `UTF32Tools.CharEnumerator` for the [NetworkString](#).

6.251.3.24 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for this [NetworkString](#).

Returns

A 32-bit signed integer hash code.

6.251.3.25 IndexOf() [1/6]

```
int IndexOf (
    char c,
    int startIndex,
    int count )
```

Returns the index of the first occurrence of a specified character in this instance.

Parameters

<i>c</i>	The Unicode character to seek.
<i>startIndex</i>	The search starting position.
<i>count</i>	The number of character positions to examine.

Returns

The zero-based index position of value if that character is found, or -1 if it is not.

6.251.3.26 IndexOf() [2/6]

```
int IndexOf (
    char c,
    int startIndex = 0 )
```

Returns the index of the first occurrence of a specified character in this instance.

Parameters

<i>c</i>	The Unicode character to seek.
<i>startIndex</i>	The search starting position.

Returns

The zero-based index position of value if that character is found, or -1 if it is not.

6.251.3.27 IndexOf() [3/6]

```
int IndexOf (
    string str,
    int startIndex,
    int count )
```

Returns the index of the first occurrence of a specified string in this instance.

Parameters

<i>str</i>	The string to seek.
<i>startIndex</i>	The search starting position.
<i>count</i>	The number of character positions to examine.

Returns

The zero-based index position of value if that string is found, or -1 if it is not.

Exceptions

<i>ArgumentNullException</i>	Thrown when the string is null.
<i>ArgumentOutOfRangeException</i>	Thrown when the start index is less than zero or greater than the safe length of the string, or when the count is less than zero or the sum of the start index and count is greater than the safe length of the string.

6.251.3.28 IndexOf() [4/6]

```
int IndexOf (
    string str,
    int startIndex = 0 )
```

Returns the index of the first occurrence of a specified string in this instance.

Parameters

<i>str</i>	The string to seek.
<i>startIndex</i>	The search starting position.

Returns

The zero-based index position of value if that string is found, or -1 if it is not.

6.251.3.29 IndexOf() [5/6]

```
int IndexOf (
    uint codePoint,
    int startIndex,
    int count )
```

Returns the index of the first occurrence of a specified Unicode code point in this instance.

Parameters

<i>codePoint</i>	The Unicode code point to seek.
<i>startIndex</i>	The search starting position.
<i>count</i>	The number of character positions to examine.

Returns

The zero-based index position of value if that Unicode code point is found, or -1 if it is not.

Exceptions

<i>ArgumentOutOfRangeException</i>	Thrown when the start index is less than zero or greater than the safe length of the string, or when the count is less than zero or the sum of the start index and count is greater than the safe length of the string.
------------------------------------	---

6.251.3.30 IndexOf() [6/6]

```
int IndexOf (
    uint codePoint,
    int startIndex = 0 )
```

Returns the index of the first occurrence of a specified Unicode code point in this instance.

Parameters

<i>codePoint</i>	The Unicode code point to seek.
<i>startIndex</i>	The search starting position.

Returns

The zero-based index position of value if that Unicode code point is found, or -1 if it is not.

6.251.3.31 IndexOf< TOtherSize >() [1/4]

```
int IndexOf< TOtherSize > (
    NetworkString< TOtherSize > str,
    int startIndex,
    int count )
```

Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>str</i>	The NetworkString to seek.
<i>startIndex</i>	The search starting position.
<i>count</i>	The number of character positions to examine.

Returns

The zero-based index position of value if that [NetworkString](#) is found, or -1 if it is not.

Type Constraints

TOtherSize : *unmanaged*
TOtherSize : *IFixedStorage*
TOtherSize : *IndexOf*
TOtherSize : *ref*
TOtherSize : *str*
TOtherSize : *startIndex*
TOtherSize : *count*

6.251.3.32 IndexOf< TOtherSize >() [2/4]

```
int IndexOf< TOtherSize > (
    NetworkString< TOtherSize > str,
    int startIndex = 0 )
```

Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>str</i>	The NetworkString to seek.
<i>startIndex</i>	The search starting position.

Returns

The zero-based index position of value if that [NetworkString](#) is found, or -1 if it is not.

Type Constraints

TOtherSize : *unmanaged*
TOtherSize : *IFixedStorage*
TOtherSize : *IndexOf*
TOtherSize : *ref*
TOtherSize : *str*
TOtherSize : *startIndex*
TOtherSize : *SafeLength*
TOtherSize : *startIndex*

6.251.3.33 `IndexOf< TOtherSize >()` [3/4]

```
int IndexOf< TOtherSize > (
    ref NetworkString< TOtherSize > str,
    int startIndex,
    int count )
```

Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>str</i>	The NetworkString to seek.
<i>startIndex</i>	The search starting position.
<i>count</i>	The number of character positions to examine.

Returns

The zero-based index position of value if that [NetworkString](#) is found, or -1 if it is not.

Exceptions

<i>TOtherSize</i> : unmanaged	Thrown when the start index is less than zero or greater than the safe length of the string, or when the count is less than zero or the sum of the start index and count is greater than the safe length of the string.
---	---

Type Constraints

***TOtherSize* : [unmanaged](#)**
***TOtherSize* : [IFixedStorage](#)**

6.251.3.34 IndexOf< TOtherSize >() [4/4]

```
int IndexOf< TOtherSize > (
    ref NetworkString< TOtherSize > str,
    int startIndex = 0 )
```

Returns the index of the first occurrence of a specified [NetworkString](#) in this instance.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>str</i>	The NetworkString to seek.
<i>startIndex</i>	The search starting position.

Returns

The zero-based index position of value if that [NetworkString](#) is found, or -1 if it is not.

Type Constraints

***TOtherSize* : [unmanaged](#)**
***TOtherSize* : [IFixedStorage](#)**
***TOtherSize* : [IndexOf](#)**
***TOtherSize* : [ref](#)**
***TOtherSize* : [str](#)**
***TOtherSize* : [startIndex](#)**
***TOtherSize* : [SafeLength](#)**
***TOtherSize* : [startIndex](#)**

6.251.3.35 operator NetworkString< TSize >()

```
static implicit operator NetworkString< TSize > (
    string str ) [static]
```

Defines an implicit conversion of a string to a [NetworkString](#).

Parameters

<i>str</i>	The string to convert.
------------	------------------------

Returns

A new instance of [NetworkString](#) with the same value as the string.

6.251.3.36 operator string()

```
static operator string (
    NetworkString< TSize > str ) [explicit], [static]
```

Defines an explicit conversion of a [NetworkString](#) to a string.

Parameters

<i>str</i>	The NetworkString to convert.
------------	---

Returns

The string value of the [NetworkString](#).

6.251.3.37 operator"!=() [1/3]

```
static bool operator!= (
    NetworkString< TSize > a,
    NetworkString< TSize > b ) [static]
```

Defines an inequality operator for [NetworkString](#).

Parameters

<i>a</i>	The first NetworkString to compare.
<i>b</i>	The second NetworkString to compare.

Returns

true if the NetworkStrings are not equal; otherwise, false.

6.251.3.38 operator"!=() [2/3]

```
static bool operator!= (
    NetworkString< TSize > a,
    string b )  [static]
```

Defines an inequality operator for a [NetworkString](#) and a string.

Parameters

<i>a</i>	The NetworkString to compare.
<i>b</i>	The string to compare.

Returns

true if the [NetworkString](#) and the string are not equal; otherwise, false.

6.251.3.39 operator"!=() [3/3]

```
static bool operator!= (
    string a,
    NetworkString< TSize > b )  [static]
```

Defines an inequality operator for a string and a [NetworkString](#).

Parameters

<i>a</i>	The string to compare.
<i>b</i>	The NetworkString to compare.

Returns

true if the string and the [NetworkString](#) are not equal; otherwise, false.

6.251.3.40 operator==() [1/3]

```
static bool operator== (
    NetworkString< TSize > a,
    NetworkString< TSize > b )  [static]
```

Defines an equality operator for [NetworkString](#).

Parameters

<i>a</i>	The first NetworkString to compare.
<i>b</i>	The second NetworkString to compare.

Returns

true if the NetworkStrings are equal; otherwise, false.

6.251.3.41 operator==() [2/3]

```
static bool operator== (
    NetworkString< TSize > a,
    string b ) [static]
```

Defines an equality operator for a [NetworkString](#) and a string.

Parameters

<i>a</i>	The NetworkString to compare.
<i>b</i>	The string to compare.

Returns

true if the [NetworkString](#) and the string are equal; otherwise, false.

6.251.3.42 operator==() [3/3]

```
static bool operator== (
    string a,
    NetworkString< TSize > b ) [static]
```

Defines an equality operator for a string and a [NetworkString](#).

Parameters

<i>a</i>	The string to compare.
<i>b</i>	The NetworkString to compare.

Returns

true if the string and the [NetworkString](#) are equal; otherwise, false.

6.251.3.43 Set()

```
bool Set (
    string value )
```

Converts *value* to UTF32 string and stores it internally.

Parameters

<i>value</i>	The string to set.
--------------	--------------------

Returns

False if *value* was too long to fit and had to be trimmed.

6.251.3.44 StartsWith()

```
bool StartsWith (
    string s )
```

Checks if the current [NetworkString](#) starts with a specified string.

Parameters

<i>s</i>	The string to check.
----------	----------------------

Returns

true if the current [NetworkString](#) starts with the specified string; otherwise, false.

Exceptions

ArgumentNullException	Thrown when the string is null.
---------------------------------------	---------------------------------

6.251.3.45 StartsWith< TOtherSize >()

```
bool StartsWith< TOtherSize > (
    ref NetworkString< TOtherSize > other )
```

Checks if the current [NetworkString](#) starts with a specified [NetworkString](#) of a different size.

Template Parameters

<i>TOtherSize</i>	The size of the other NetworkString .
-------------------	---

Parameters

<i>other</i>	The NetworkString to check.
--------------	---

Returns

true if the current [NetworkString](#) starts with the specified [NetworkString](#); otherwise, false.

Type Constraints

TOtherSize : *unmanaged*
TOtherSize : *IFixedStorage*

6.251.3.46 Substring() [1/2]

```
NetworkString<TSize> Substring (  
    int startIndex )
```

Returns a substring from this instance. The substring starts at a specified character position.

Parameters

<i>startIndex</i>	The zero-based starting character position of a substring in this instance.
-------------------	---

Returns

A new [NetworkString](#) that is equivalent to the substring that begins at *startIndex* in this instance, or [NetworkString.Empty](#) if *startIndex* is equal to the length of this instance.

Exceptions

<i>ArgumentOutOfRangeException</i>	<i>startIndex</i> is less than zero or greater than the length of this instance.
------------------------------------	--

6.251.3.47 Substring() [2/2]

```
NetworkString<TSize> Substring (  
    int startIndex,  
    int length )
```

Returns a substring from this instance. The substring starts at a specified character position and has a specified length.

Parameters

<i>startIndex</i>	The zero-based starting character position of a substring in this instance.
<i>length</i>	The number of characters in the substring.

Returns

A new [NetworkString](#) that is equivalent to the substring of length *length* that begins at *startIndex* in this instance, or `NetworkString::Empty` if *startIndex* is equal to the length of this instance and *length* is zero.

Exceptions

<i>ArgumentOutOfRangeException</i>	<i>startIndex</i> plus <i>length</i> indicates a position not within this instance, or <i>startIndex</i> or <i>length</i> is less than zero.
------------------------------------	--

6.251.3.48 ToLower()

```
NetworkString<TSize> ToLower ( )
```

Converts all the characters in this [NetworkString](#) to lowercase.

Returns

A new [NetworkString](#) in which all characters in this [NetworkString](#) are converted to lowercase.

6.251.3.49 ToString()

```
override string ToString ( )
```

Converts the value of this [NetworkString](#) to its equivalent string representation.

Returns

A string representation of the value of this [NetworkString](#).

6.251.3.50 ToUpper()

```
NetworkString<TSize> ToUpper ( )
```

Converts all the characters in this [NetworkString](#) to uppercase.

Returns

A new [NetworkString](#) in which all characters in this [NetworkString](#) are converted to uppercase.

6.251.4 Property Documentation

6.251.4.1 Capacity

```
int Capacity [get]
```

Maximum UTF32 string length.

6.251.4.2 Length

```
int Length [get]
```

Number of UTF32 scalars. It is equal or less than [GetCharCount](#) or the length of [Value](#), because those use UTF16 encoding, which needs two characters to encode some values.

6.251.4.3 this[int index]

```
ref uint this[int index] [get]
```

Returns UTF32 scalar at *index* position. To iterate over characters, use [GetEnumerator](#).

Parameters

<i>index</i>	Index to get.
--------------	---------------

Returns

UTF32 scalar at *index* position.

6.251.4.4 Value

```
string Value [get], [set]
```

Converts to/from regular UTF16 string. Setter is alloc-free. Use [Get](#) to get possibly alloc-free conversion.

6.252 NetworkStructUtils Class Reference

Utility methods for [INetworkStruct](#)

Static Public Member Functions

- static int [GetWordCount](#) (Type type)
- static int [GetWordCount< T >](#) ()
Get INetworkStruct Word Count

6.252.1 Detailed Description

Utility methods for [INetworkStruct](#)

6.252.2 Member Function Documentation

6.252.2.1 GetWordCount< T >()

static int [GetWordCount< T >](#) () [static]

Get [INetworkStruct](#) Word Count

Template Parameters

<i>T</i>	INetworkStruct type reference
----------	---

Returns

Number of Words necessary for this specific [INetworkStruct](#)

Type Constraints

T : *unmanaged*

T : [INetworkStruct](#)

6.253 NetworkStructWeavedAttribute Class Reference

Describes the total number of WORDs a [INetworkStruct](#) uses.

Inherits Attribute.

Public Member Functions

- [NetworkStructWeavedAttribute](#) (int wordCount)
NetworkStructWeavedAttribute Constructor

Properties

- int [WordCount](#) [get]
INetworkStruct Word Count

6.253.1 Detailed Description

Describes the total number of WORDs a [INetworkStruct](#) uses.

6.253.2 Constructor & Destructor Documentation

6.253.2.1 NetworkStructWeavedAttribute()

```
NetworkStructWeavedAttribute (
    int wordCount )
```

[NetworkStructWeavedAttribute](#) Constructor

Parameters

<code>wordCount</code>	INetworkStruct word count
------------------------	---

6.253.3 Property Documentation

6.253.3.1 WordCount

```
int WordCount [get]
```

[INetworkStruct](#) Word Count

6.254 NetworkTransform Class Reference

Add to any [NetworkObject](#) Transform, or its associated child Transforms to automatically synchronize TRSP (Position/Rotation/Scale/Parent).

Inherits [NetworkTRSP](#), [INetworkTRSPTeleport](#), [IBeforeAllTicks](#), [IAfterAllTicks](#), and [IBeforeCopyPreviousState](#).

Public Member Functions

- override void [Render \(\)](#)
Post simulation frame rendering callback. Runs after all simulations have finished. Use in place of Unity's Update when [Fusion](#) is handling Physics.
- override void [SetAreaOfInterestOverride \(NetworkObject obj\)](#)
Manually set the [NetworkObject](#) used as the AreaOfInterestOverride.
- override void [Spawned \(\)](#)
Post spawn callback.
- void [Teleport \(Vector3? position=null, Quaternion? rotation=null\)](#)
Set the transform position and rotation to the indicated values, and network the Teleport event. This will suspend interpolation between the previous tick state and the current tick state in [Render\(\)](#), on this peer and all remote peers.

Public Attributes

- bool [DisableSharedModeInterpolation = false](#)
Disable interpolation on State Authority in Shared Mode. You should disable interpolation if your controller code moves an object inside of [Update\(\)](#) rather than [FixedUpdateNetwork\(\)](#).
- bool [SyncParent = false](#)
Enables synchronization of transform.parent. NOTE: Parent GameObjects must have a [NetworkBehaviour](#) derived component to be a valid parent, parent must belong to a different [NetworkObject](#) than this Object.
- bool [SyncScale = false](#)
Enables synchronization of LocalScale.

Properties

- bool [AutoUpdateAreaOfInterestOverride \[get, set\]](#)
Determines if parent changes should automatically call [SetAreaOfInterestOverride\(NetworkObject\)](#), and assign the parent [NetworkObject](#) as the override. Default is true, as you typically will want player interest in this object to reflect player interest in the nested parent object. For example, if a player is carrying an nested Object, players should only see that carried Object if they see the player. Additionally, AOI works in world space, and [NetworkTransform](#) operates in local space, so any AOI position values of nested Objects will ALWAYS be invalid, so nested Objects should always have their AOI Override set to a non-nested Object.

Additional Inherited Members

6.254.1 Detailed Description

Add to any [NetworkObject](#) Transform, or its associated child Transforms to automatically synchronize TRSP (Position/Rotation/Scale/Parent).

6.254.2 Member Function Documentation

6.254.2.1 SetAreaOfInterestOverride()

```
override void SetAreaOfInterestOverride (
    NetworkObject obj )  [virtual]
```

Manually set the [NetworkObject](#) used as the AreaOfInterestOverride.

Parameters

<i>obj</i>	NetworkObject to use as the AreaOfInterestOverride.
------------	---

Reimplemented from [NetworkTRSP](#).

6.254.2.2 Teleport()

```
void Teleport (
    Vector3? position = null,
    Quaternion? rotation = null )
```

Set the transform position and rotation to the indicated values, and network the Teleport event. This will suspend interpolation between the previous tick state and the current tick state in [Render\(\)](#), on this peer and all remote peers.

Implements [INetworkTRSPTeleport](#).

6.254.3 Member Data Documentation

6.254.3.1 DisableSharedModeInterpolation

```
bool DisableSharedModeInterpolation = false
```

Disable interpolation on State Authority in Shared Mode. You should disable interpolation if your controller code moves an object inside of [Update\(\)](#) rather than [FixedUpdateNetwork\(\)](#).

6.254.3.2 SyncParent

```
bool SyncParent = false
```

Enables synchronization of `transform.parent`. NOTE: Parent GameObjects must have a [NetworkBehaviour](#) derived component to be a valid parent, parent must belong to a different [NetworkObject](#) than this Object.

6.254.3.3 SyncScale

```
bool SyncScale = false
```

Enables synchronization of `LocalScale`.

6.254.4 Property Documentation

6.254.4.1 AutoUpdateAreaOfInterestOverride

```
bool AutoUpdateAreaOfInterestOverride [get], [set]
```

Determines if parent changes should automatically call [SetAreaOfInterestOverride\(NetworkObject\)](#), and assign the parent [NetworkObject](#) as the override. Default is true, as you typically will want player interest in this object to reflect player interest in the nested parent object. For example, if a player is carrying an nested Object, players should only see that carried Object if they see the player. Additionally, AOI works in world space, and [NetworkTransform](#) operates in local space, so any AOI position values of nested Objects will ALWAYS be invalid, so nested Objects should always have their AOI Override set to a non-nested Object.

6.255 NetworkTRSP Class Reference

Base class for spatial (Position/Rotation/Scale/Parent) synchronization component, such as [NetworkTransform](#). Provides the base logic for render interpolation, parenting synchronization, and teleport, that can be used in components derived from this class.

Inherits [NetworkBehaviour](#).

Inherited by [NetworkTransform](#).

Public Member Functions

- virtual void [SetAreaOfInterestOverride \(NetworkObject obj\)](#)
Manually set the [NetworkObject](#) used as the AreaOfInterestOverride.

Static Protected Member Functions

- static void [Render \(NetworkTRSP behaviour, Transform transform, bool syncScale, bool syncParent, bool local, ref Tick initial\)](#)
Default Render handling for [NetworkTRSP](#) derived classes.
- static void [ResolveAOIOVERRIDE \(NetworkTRSP behaviour, Transform parent\)](#)
Recursively attempts to find nested parent [NetworkObject](#), and if found assigns that [NetworkObject](#) as the AreaOfInterestOverride.
- static void [SetParentTransform \(NetworkTRSP behaviour, Transform transform, NetworkBehaviourId parentId\)](#)
Default handling for setting a [NetworkTRSP](#)'s parent using a [NetworkBehaviourId](#) value.
- static void [Teleport \(NetworkTRSP behaviour, Transform transform, Vector3? position=null, Quaternion? rotation=null\)](#)
The default Teleport implementation for [NetworkTRSP](#) derived classes.

Properties

- **NetworkTRSPData Data** [get]
The networked data of this NetworkTRSP.
- **bool IsMainTRSP** [get]
The main NetworkTRSP is at the root of the NetworkObject and it will be used for area of interest operations and parenting of the NetworkObject.
- **ref NetworkTRSPData State** [get]
A reference to the networked data of this NetworkTRSP.

Additional Inherited Members

6.255.1 Detailed Description

Base class for spatial (Position/Rotation/Scale/Parent) synchronization component, such as [NetworkTransform](#). Provides the base logic for render interpolation, parenting synchronization, and teleport, that can be used in components derived from this class.

6.255.2 Member Function Documentation

6.255.2.1 Render()

```
static void Render (
    NetworkTRSP behaviour,
    Transform transform,
    bool syncScale,
    bool syncParent,
    bool local,
    ref Tick initial ) [static], [protected]
```

Default Render handling for [NetworkTRSP](#) derived classes.

6.255.2.2 ResolveAOIOVERRIDE()

```
static void ResolveAOIOVERRIDE (
    NetworkTRSP behaviour,
    Transform parent ) [static], [protected]
```

Recursively attempts to find nested parent [NetworkObject](#), and if found assigns that [NetworkObject](#) as the AreaOfInterestOverride.

Parameters

behaviour	Only pass a NetworkTRSP derived class that is on the same Transform as its associated NetworkObject , as AreaOfInterestOverride is only applicable when IsMainTRSP is true.
------------------	---

Parameters

<i>parent</i>	The direct parent of the
---------------	--------------------------

6.255.2.3 SetAreaOfInterestOverride()

```
virtual void SetAreaOfInterestOverride (
    NetworkObject obj ) [virtual]
```

Manually set the [NetworkObject](#) used as the AreaOfInterestOverride.

Parameters

<i>obj</i>	NetworkObject to use as the AreaOfInterestOverride.
------------	---

Reimplemented in [NetworkTransform](#).

6.255.2.4 SetParentTransform()

```
static void SetParentTransform (
    NetworkTRSP behaviour,
    Transform transform,
    NetworkBehaviourId parentId ) [static], [protected]
```

Default handling for setting a [NetworkTRSP](#)'s parent using a [NetworkBehaviourId](#) value.

6.255.2.5 Teleport()

```
static void Teleport (
    NetworkTRSP behaviour,
    Transform transform,
    Vector3? position = null,
    Quaternion? rotation = null ) [static], [protected]
```

The default Teleport implementation for [NetworkTRSP](#) derived classes.

6.255.3 Property Documentation

6.255.3.1 Data

```
NetworkTRSPData Data [get]
```

The networked data of this [NetworkTRSP](#).

6.255.3.2 IsMainTRSP

```
bool IsMainTRSP [get]
```

The main [NetworkTRSP](#) is at the root of the [NetworkObject](#) and it will be used for area of interest operations and parenting of the [NetworkObject](#).

6.255.3.3 State

```
ref NetworkTRSPData State [get], [protected]
```

A reference to the networked data of this [NetworkTRSP](#).

6.256 NetworkTRSPData Struct Reference

Data structure storing spatial (Position/Rotation/Scale/Parent) synchronization data for spatial synchronization components, [NetworkTRSP](#) and its subclass [NetworkTransform](#).

Inherits [INetworkStruct](#).

Public Attributes

- [NetworkId AreaOfInterestOverride](#)
Id of a behaviour used as the reference point for this component during area of interest operations. The behaviour should be a [NetworkTRSP](#) derived class, that is on the same Transform as its associated [NetworkObject](#)
- [NetworkBehaviour Parent](#)
Id of a [NetworkBehaviour](#) on the parent of the component's transform.
- [Vector3 Position](#)
Position relevant for the spatial synchronization component (can be used to either store a local position or a world position, depending on the component)
- [Quaternion Rotation](#)
Rotation relevant for the spatial synchronization component (can be used to either store a local rotation or a world rotation, depending on the component)
- [Vector3Compressed Scale](#)
Scale relevant for the spatial synchronization component
- int [TeleportKey](#)
Key used to differentiate between several teleports

Static Public Attributes

- const int **POSITION_OFFSET** = 2
Offset to point at the position values on the data buffer
- const int **SIZE** = **WORDS** * **Allocator.REPLICATE_WORD_SIZE**
The actual size for the networked properties in bytes
- const int **WORDS** = 14
Networked properties word count for the base [NetworkTRSPData](#)

Properties

- static [NetworkBehaviourId NonNetworkedParent](#) [get]
Special [NetworkBehaviourId](#) value, used as a flag to tell the parent is a non-networked object

6.256.1 Detailed Description

Data structure storing spatial (Position/Rotation/Scale/Parent) synchronization data for spatial synchronization components, [NetworkTRSP](#) and its subclass [NetworkTransform](#).

6.256.2 Member Data Documentation

6.256.2.1 AreaOfInterestOverride

[NetworkId](#) **AreaOfInterestOverride**

Id of a behaviour used as the reference point for this component during area of interest operations. The behaviour should be a [NetworkTRSP](#) derived class, that is on the same Transform as its associated [NetworkObject](#)

6.256.2.2 Parent

[NetworkBehaviourId](#) **Parent**

Id of a [NetworkBehaviour](#) on the parent of the component's transform.

6.256.2.3 Position

[Vector3](#) **Position**

Position relevant for the spatial synchronization component (can be used to either store a local position or a world position, depending on the component)

6.256.2.4 POSITION_OFFSET

```
const int POSITION_OFFSET = 2 [static]
```

Offset to point at the position values on the data buffer

6.256.2.5 Rotation

Quaternion Rotation

Rotation relevant for the spatial synchronization component (can be used to either store a local rotation or a world rotation, depending on the component)

6.256.2.6 Scale

[Vector3Compressed](#) Scale

Scale relevant for the spatial synchronization component

6.256.2.7 SIZE

```
const int SIZE = WORDS * Allocator.REPLICATE_WORD_SIZE [static]
```

The actual size for the networked properties in bytes

6.256.2.8 TeleportKey

int TeleportKey

Key used to differentiate between several teleports

6.256.2.9 WORDS

```
const int WORDS = 14 [static]
```

Networked properties word count for the base [NetworkTRSPData](#)

6.256.3 Property Documentation

6.256.3.1 NonNetworkedParent

`NetworkBehaviourId` NonNetworkedParent [static], [get]

Special `NetworkBehaviourId` value, used as a flag to tell the parent is a non-networked object

6.257 NormalizedRectAttribute Class Reference

Enables a special inspector drawer for Unity Rect type, specially designed for editing RectTransforms using normalized values.

Inherits `PropertyAttribute`.

Public Member Functions

- `NormalizedRectAttribute` (bool invertY=true, float aspectRatio=0)

Constructor for `NormalizedRectAttribute`. InvertY inverts Y handling, for RectTransforms which treat lowerRight as origin, rather than upper left.

Public Attributes

- float `AspectRatio`
Set the Aspect Ratio
- bool `InvertY`
Signal if Y should be inverted

6.257.1 Detailed Description

Enables a special inspector drawer for Unity Rect type, specially designed for editing RectTransforms using normalized values.

6.257.2 Constructor & Destructor Documentation

6.257.2.1 NormalizedRectAttribute()

```
NormalizedRectAttribute (
    bool invertY = true,
    float aspectRatio = 0 )
```

Constructor for `NormalizedRectAttribute`. InvertY inverts Y handling, for RectTransforms which treat lowerRight as origin, rather than upper left.

Parameters

<i>invertY</i>	Invert Y handling
<i>aspectRatio</i>	Expressed as Width/Height, this defines the ratio of the box shown in the inspector. Value of 0 indicates game window resolution will be used.

6.257.3 Member Data Documentation**6.257.3.1 AspectRatio**

```
float AspectRatio
```

Set the Aspect Ratio

6.257.3.2 InvertY

```
bool InvertY
```

Signal if Y should be inverted

6.258 OnChangedRenderAttribute Class Reference

OnChangedRender Attribute

Inherits Attribute.

Public Member Functions

- [OnChangedRenderAttribute](#) (string methodName)
Initializes a new instance of the [OnChangedRenderAttribute](#) class.

Properties

- string [MethodName](#) [get]
Gets the name of the method to be called when the property changes.

6.258.1 Detailed Description

OnChangedRender Attribute

This attribute is used to specify a method that should be called when the property changes.

6.258.2 Constructor & Destructor Documentation

6.258.2.1 OnChangedRenderAttribute()

```
OnChangedRenderAttribute (
    string methodName )
```

Initializes a new instance of the [OnChangedRenderAttribute](#) class.

Parameters

<code>methodName</code>	The name of the method to be called when the property changes.
-------------------------	--

Exceptions

<code>ArgumentNullException</code>	Thrown when <code>methodName</code> is null or empty.
------------------------------------	---

6.258.3 Property Documentation

6.258.3.1 MethodName

```
string MethodName [get]
```

Gets the name of the method to be called when the property changes.

6.259 PlayerRef Struct Reference

Represents a [Fusion](#) player.

Inherits [INetworkStruct](#), and [IEquatable< PlayerRef >](#).

Public Member Functions

- override bool [Equals](#) (object obj)
Determines whether the specified object is equal to the current object.
- bool [Equals](#) ([PlayerRef](#) other)
Determines whether the specified [PlayerRef](#) is equal to the current [PlayerRef](#).
- override int [GetHashCode](#) ()
Serves as the default hash function.
- override string [ToString](#) ()
Returns a string that represents the current object.

Static Public Member Functions

- static `PlayerRef FromEncoded` (int encoded)

Creates a new `PlayerRef` from the given encoded value.
- static `PlayerRef FromIndex` (int index)

Creates a new `PlayerRef` from the given index.
- static bool `operator!=` (`PlayerRef` a, `PlayerRef` b)

Determines whether two `PlayerRef` instances are not equal.
- static bool `operator==` (`PlayerRef` a, `PlayerRef` b)

Determines whether two `PlayerRef` instances are equal.
- static unsafe `PlayerRef Read` (NetBitBuffer *buffer)

Reads a `PlayerRef` from the provided NetBitBuffer.
- static unsafe void `Write` (NetBitBuffer *buffer, `PlayerRef` playerRef)

Writes the `PlayerRef` to the provided NetBitBuffer.
- static unsafe void `Write< T >` (T *buffer, `PlayerRef` playerRef)

Writes the `PlayerRef` to the provided buffer.

Public Attributes

- int `_index`

Static Public Attributes

- const int `MASTER_CLIENT_RAW` = -1

A constant representing the raw index value for the master client.
- const int `SIZE` = 4

The size of the `PlayerRef` structure in bytes.

Properties

- int `AsIndex` [get]

Returns the `PlayerRef` int as an integer Id value.
- static IEqualityComparer< `PlayerRef` > `Comparer` = new IndexEqualityComparer() [get]

Gets an equality comparer that can be used to compare two `PlayerRef` instances.
- bool `IsMasterClient` [get]

Returns true if this `PlayerRef` indicates the MasterClient rather than a specific Player by Index. This is a special flag value which has the encoded index value of -2 (internal raw backing value of -1). This is not a valid `PlayerRef` value in itself, and no Runner will ever be assigned this value as its LocalPlayer. It is used by properties like Object.State Authority to indicate that the MasterClient has authority (which ever player that currently is), rather than a specific Player.
- bool `IsNone` [get]

Returns true if the index value equals -1 (internal raw value of 0), indicating no player.
- bool `IsRealPlayer` [get]

If this player ref is a valid unique player index
- static `PlayerRef MasterClient` [get]

Special master client player ref value of -1
- static `PlayerRef None` [get]

None player
- int `PlayerId` [get]

Returns the `PlayerRef` as an integer Id value.
- int `RawEncoded` [get]

Returns the index backing value without modification. Unlike `AsIndex` which returns the backing value - 1.

6.259.1 Detailed Description

Represents a [Fusion](#) player.

The [PlayerRef](#), in contrast to the player index, is 1-based. The reason is that `default(PlayerRef)` will return a "null/invalid" player ref struct for convenience. There are automatic cast operators that can cast an int into a [PlayerRef](#).

```
default(PlayerRef), internally a 0, means NOBODY  
PlayerRef, internally 1, is the same as player index 0  
PlayerRef, internally 2, is the same as player index 1
```

6.259.2 Member Function Documentation

6.259.2.1 Equals() [1/2]

```
override bool Equals (  
    object obj )
```

Determines whether the specified object is equal to the current object.

Parameters

<i>obj</i>	The object to compare with the current object.
------------	--

Returns

true if the specified object is equal to the current object; otherwise, false.

6.259.2.2 Equals() [2/2]

```
bool Equals (  
    PlayerRef other )
```

Determines whether the specified [PlayerRef](#) is equal to the current [PlayerRef](#).

Parameters

<i>other</i>	The PlayerRef to compare with the current PlayerRef .
--------------	---

Returns

true if the specified [PlayerRef](#) is equal to the current [PlayerRef](#); otherwise, false.

6.259.2.3 FromEncoded()

```
static PlayerRef FromEncoded (
    int encoded ) [static]
```

Creates a new [PlayerRef](#) from the given encoded value.

Parameters

<i>encoded</i>	The encoded value to create the PlayerRef from.
----------------	---

Returns

A new [PlayerRef](#) that represents the encoded value.

6.259.2.4 FromIndex()

```
static PlayerRef FromIndex (
    int index ) [static]
```

Creates a new [PlayerRef](#) from the given index.

Parameters

<i>index</i>	The index to create the PlayerRef from.
--------------	---

Returns

A new [PlayerRef](#) that represents the index.

6.259.2.5 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

6.259.2.6 operator"!=()

```
static bool operator!= (
    PlayerRef a,
    PlayerRef b ) [static]
```

Determines whether two [PlayerRef](#) instances are not equal.

Parameters

<i>a</i>	The first PlayerRef to compare.
<i>b</i>	The second PlayerRef to compare.

Returns

true if the PlayerRefs are not equal; otherwise, false.

6.259.2.7 operator==()

```
static bool operator== (
    PlayerRef a,
    PlayerRef b ) [static]
```

Determines whether two [PlayerRef](#) instances are equal.

Parameters

<i>a</i>	The first PlayerRef to compare.
<i>b</i>	The second PlayerRef to compare.

Returns

true if the PlayerRefs are equal; otherwise, false.

6.259.2.8 Read()

```
static unsafe PlayerRef Read (
    NetBitBuffer * buffer ) [static]
```

Reads a [PlayerRef](#) from the provided NetBitBuffer.

Parameters

<i>buffer</i>	The buffer to read from.
---------------	--------------------------

Returns

The [PlayerRef](#) read from the buffer.

6.259.2.9 `ToString()`

```
override string ToString ( )
```

Returns a string that represents the current object.

6.259.2.10 `Write()`

```
static unsafe void Write (
    NetBitBuffer * buffer,
    PlayerRef playerRef ) [static]
```

Writes the [PlayerRef](#) to the provided NetBitBuffer.

Parameters

<i>buffer</i>	The buffer to write to.
<i>playerRef</i>	The PlayerRef to write.

6.259.2.11 `Write< T >()`

```
static unsafe void Write< T > (
    T * buffer,
    PlayerRef playerRef ) [static]
```

Writes the [PlayerRef](#) to the provided buffer.

Template Parameters

<i>T</i>	The type of the buffer. Must be unmanaged and implement INetBitWriteStream .
----------	--

Parameters

<i>buffer</i>	The buffer to write to.
<i>playerRef</i>	The PlayerRef to write.

Type Constraints

T : *unmanaged*

T : *INetBitWriteStream*

6.259.3 Member Data Documentation

6.259.3.1 MASTER_CLIENT_RAW

```
const int MASTER_CLIENT_RAW = -1 [static]
```

A constant representing the raw index value for the master client.

6.259.3.2 SIZE

```
const int SIZE = 4 [static]
```

The size of the [PlayerRef](#) structure in bytes.

6.259.4 Property Documentation

6.259.4.1 AsIndex

```
int AsIndex [get]
```

Returns the [PlayerRef](#) int as an integer Id value.

-1=None -2=MasterClient >=0=PlayerId

6.259.4.2 Comparer

```
IEqualityComparer<PlayerRef> Comparer = new IndexEqualityComparer() [static], [get]
```

Gets an equality comparer that can be used to compare two [PlayerRef](#) instances.

6.259.4.3 IsMasterClient

```
bool IsMasterClient [get]
```

Returns true if this [PlayerRef](#) indicates the MasterClient rather than a specific Player by Index. This is a special flag value which has the encoded index value of -2 (internal raw backing value of -1). This is not a valid [PlayerRef](#) value in itself, and no Runner will ever be assigned this value as its LocalPlayer. It is used by properties like Object.State Authority to indicate that the MasterClient has authority (which ever player that currently is), rather than a specific Player.

6.259.4.4 IsNone

```
bool IsNone [get]
```

Returns true if the index value equals -1 (internal raw value of 0), indicating no player.

6.259.4.5 IsRealPlayer

```
bool IsRealPlayer [get]
```

If this player ref is a valid unique player index

6.259.4.6 MasterClient

```
PlayerRef MasterClient [static], [get]
```

Special master client player ref value of -1

6.259.4.7 None

```
PlayerRef None [static], [get]
```

None player

6.259.4.8 PlayerId

```
int PlayerId [get]
```

Returns the [PlayerRef](#) as an integer Id value.

-1=None -2=MasterClient

6.259.4.9 RawEncoded

```
int RawEncoded [get]
```

Returns the index backing value without modification. Unlike [AsIndex](#) which returns the backing value - 1.

0=None -1=MasterClient >0=PlayerId

6.260 PreserveInPluginAttribute Class Reference

Preserve In Plugin Attribute

Inherits Attribute.

Public Member Functions

- [PreserveInPluginAttribute \(\)](#)
PreserveInPluginAttribute Constructor

6.260.1 Detailed Description

Preserve In Plugin Attribute

6.260.2 Constructor & Destructor Documentation

6.260.2.1 PreserveInPluginAttribute()

`PreserveInPluginAttribute ()`

`PreserveInPluginAttribute` Constructor

6.261 IMessage Interface Reference

Represents a [Protocol](#) Message

6.261.1 Detailed Description

Represents a [Protocol](#) Message

Used to tag the Messages in ICommunicator.

6.262 Versioning Class Reference

[Versioning](#) Information

Static Public Attributes

- static readonly Version **InvalidVersion** = new Version(0, 0, 0)

Properties

- static Version [GetCurrentVersion](#) [get]
Get the current version
- static string [GetProductVersion](#) [get]
Get the current product version

6.262.1 Detailed Description

[Versioning](#) Information

6.262.2 Property Documentation

6.262.2.1 GetCurrentVersion

Version [GetCurrentVersion](#) [static], [get]

Get the current version

6.262.2.2 GetProductVersion

string [GetProductVersion](#) [static], [get]

Get the current product version

6.263 Ptr Struct Reference

[Ptr](#)

Inherits [IEquatable< Ptr >](#), and [INetworkStruct](#).

Classes

- class [EqualityComparer](#)
Ptr Equality Comparer

Public Member Functions

- override bool `Equals` (object `obj`)
Check `Ptr` equality
- bool `Equals` (`Ptr` `other`)
Check `Ptr` equality
- override int `GetHashCode` ()
`Ptr` Hash Code, same as `Address`
- override string `ToString` ()
`Ptr` to String

Static Public Member Functions

- static implicit operator bool (`Ptr` `a`)
Implicit Bool Operator Check if `Address` is not 0
- static bool `operator!=` (`Ptr` `a`, `Ptr` `b`)
Implicit `Ptr` Not Equals Operator
- static `Ptr operator+` (`Ptr` `p`, int `v`)
Implicit `Ptr` Sum Operator
- static `Ptr operator-` (`Ptr` `p`, int `v`)
Implicit `Ptr` Subtraction Operator
- static bool `operator==` (`Ptr` `a`, `Ptr` `b`)
Implicit `Ptr` Equals Operator

Public Attributes

- int `Address`
`Ptr` Address

Static Public Attributes

- const int `SIZE` = 4
`Ptr` Size

Properties

- static `Ptr Null` [get]
Null `Ptr`

6.263.1 Detailed Description

`Ptr`

6.263.2 Member Function Documentation

6.263.2.1 Equals() [1/2]

```
override bool Equals (
    object obj )
```

Check `Ptr` equality

Parameters

<i>obj</i>	Any object reference
------------	----------------------

Returns

True if obj is a [Ptr](#) and points to the same Address

6.263.2.2 Equals() [2/2]

```
bool Equals (
    Ptr other )
```

Check [Ptr](#) equality

Parameters

<i>other</i>	Ptr Ref
--------------	-------------------------

Returns

True if points to the same Address

6.263.2.3 GetHashCode()

```
override int GetHashCode ( )
```

[Ptr](#) Hash Code, same as [Address](#)

Returns

[Address](#)

6.263.2.4 operator bool()

```
static implicit operator bool (
    Ptr a ) [static]
```

Implicit Bool Operator Check if [Address](#) is not 0

Parameters

a	Ptr to check
---	--------------

Returns

True if [Address](#) is not 0

6.263.2.5 operator"!=()

```
static bool operator!= (
    Ptr a,
    Ptr b ) [static]
```

Implicit [Ptr](#) Not Equals Operator

Parameters

a	Ptr A
b	Ptr B

Returns

True if [Address](#) is not the same

6.263.2.6 operator+()

```
static Ptr operator+ (
    Ptr p,
    int v ) [static]
```

Implicit [Ptr](#) Sum Operator

Parameters

p	Ptr to add to
v	Value to add

Returns

[Ptr](#) with [Address](#) increased by v

6.263.2.7 operator-()

```
static Ptr operator- (
    Ptr p,
    int v ) [static]
```

Implicit **Ptr** Subtraction Operator

Parameters

<i>p</i>	Ptr to subtract from
<i>v</i>	Value to subtract

Returns

Ptr with **Address** decreased by *v*

6.263.2.8 operator==()

```
static bool operator== (
    Ptr a,
    Ptr b ) [static]
```

Implicit **Ptr** Equals Operator

Parameters

<i>a</i>	Ptr A
<i>b</i>	Ptr B

Returns

True if **Address** is the same

6.263.2.9 ToString()

```
override string ToString ( )
```

Ptr to String

Returns

Address in Hexadecimal format

6.263.3 Member Data Documentation

6.263.3.1 Address

int Address

Ptr Address

6.263.3.2 SIZE

const int SIZE = 4 [static]

Ptr Size

6.263.4 Property Documentation

6.263.4.1 Null

Ptr Null [static], [get]

Null Ptr

6.264 Ptr.EqualityComparer Class Reference

Ptr Equality Comparer

Inherits IEqualityComparer< Ptr >.

Public Member Functions

- bool Equals (Ptr x, Ptr y)
Ptr Equality Comparer
- int GetHashCode (Ptr obj)
Get Hash Code

6.264.1 Detailed Description

Ptr Equality Comparer

6.264.2 Member Function Documentation

6.264.2.1 Equals()

```
bool Equals (
```

`Ptr x,`

`Ptr y)`

`Ptr Equality Comparer`

Parameters

x	Ptr X
y	Ptr Y

Returns

True if point to the same Address

6.264.2.2 GetHashCode()

```
int GetHashCode (
    Ptr obj )
```

Get Hash Code

Parameters

obj	Ptr
-----	-----

Returns

Ptr Address

6.265 QuaternionCompressed Struct Reference

Represents a compressed Quaternion value for network transmission.

Inherits [INetworkStruct](#), and [IEquatable< QuaternionCompressed >](#).

Public Member Functions

- override bool [Equals](#) (object obj)
Checks if the provided object is a `QuaternionCompressed` instance and if it's equal to the current `QuaternionCompressed` instance.
- bool [Equals](#) (`QuaternionCompressed` other)
Checks if the current `QuaternionCompressed` instance is equal to the other `QuaternionCompressed` instance.
- override int [GetHashCode](#) ()
Returns the hash code for the current `QuaternionCompressed` instance.

Static Public Member Functions

- static implicit operator `Quaternion` (`QuaternionCompressed` q)
Implicit conversion from `QuaternionCompressed` to `Quaternion`.
- static implicit operator `QuaternionCompressed` (`Quaternion` v)
Implicit conversion from `Quaternion` to `QuaternionCompressed`.
- static bool [operator!=](#) (`QuaternionCompressed` left, `QuaternionCompressed` right)
Inequality operator for `QuaternionCompressed` struct.
- static bool [operator==](#) (`QuaternionCompressed` left, `QuaternionCompressed` right)
Equality operator for `QuaternionCompressed` struct.

Public Attributes

- int `wEncoded`
Encoded value of the w component.
- int `xEncoded`
Encoded value of the x component.
- int `yEncoded`
Encoded value of the y component.
- int `zEncoded`
Encoded value of the z component.

Properties

- float `W` [get, set]
Gets or sets the w component.
- float `X` [get, set]
Gets or sets the x component.
- float `Y` [get, set]
Gets or sets the y component.
- float `Z` [get, set]
Gets or sets the z component.

6.265.1 Detailed Description

Represents a compressed Quaternion value for network transmission.

6.265.2 Member Function Documentation

6.265.2.1 Equals() [1/2]

```
override bool Equals (
    object obj )
```

Checks if the provided object is a `QuaternionCompressed` instance and if it's equal to the current `QuaternionCompressed` instance.

Parameters

<code>obj</code>	The object to compare with the current <code>QuaternionCompressed</code> instance.
------------------	--

Returns

True if the provided object is a `QuaternionCompressed` instance and it's equal to the current `QuaternionCompressed` instance, otherwise false.

6.265.2.2 Equals() [2/2]

```
bool Equals (
    QuaternionCompressed other )
```

Checks if the current `QuaternionCompressed` instance is equal to the other `QuaternionCompressed` instance.

Parameters

<code>other</code>	The other <code>QuaternionCompressed</code> instance to compare with the current <code>QuaternionCompressed</code> instance.
--------------------	--

Returns

True if the values of both `QuaternionCompressed` instances are equal, otherwise false.

6.265.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current `QuaternionCompressed` instance.

Returns

A hash code for the current `QuaternionCompressed` instance.

6.265.2.4 operator Quaternion()

```
static implicit operator Quaternion (
    QuaternionCompressed q ) [static]
```

Implicit conversion from `QuaternionCompressed` to `Quaternion`.

Parameters

<code>q</code>	The <code>QuaternionCompressed</code> instance to convert.
----------------	--

Returns

The decompressed `Quaternion` value of the `QuaternionCompressed` instance.

6.265.2.5 operator QuaternionCompressed()

```
static implicit operator QuaternionCompressed (
    Quaternion v ) [static]
```

Implicit conversion from Quaternion to [QuaternionCompressed](#).

Parameters

<i>v</i>	The Quaternion value to convert.
----------	----------------------------------

Returns

A new [QuaternionCompressed](#) instance with the compressed value of the Quaternion.

6.265.2.6 operator"!=()

```
static bool operator!= (
    QuaternionCompressed left,
    QuaternionCompressed right ) [static]
```

Inequality operator for [QuaternionCompressed](#) struct.

Parameters

<i>left</i>	First QuaternionCompressed instance.
<i>right</i>	Second QuaternionCompressed instance.

Returns

True if the value of the first [QuaternionCompressed](#) instance is not equal to the value of the second [QuaternionCompressed](#) instance, otherwise false.

6.265.2.7 operator==()

```
static bool operator== (
    QuaternionCompressed left,
    QuaternionCompressed right ) [static]
```

Equality operator for [QuaternionCompressed](#) struct.

Parameters

<i>left</i>	First QuaternionCompressed instance.
<i>right</i>	Second QuaternionCompressed instance.

Returns

True if the value of the first `QuaternionCompressed` instance is equal to the value of the second `QuaternionCompressed` instance, otherwise false.

6.265.3 Member Data Documentation

6.265.3.1 wEncoded

```
int wEncoded
```

Encoded value of the w component.

6.265.3.2 xEncoded

```
int xEncoded
```

Encoded value of the x component.

6.265.3.3 yEncoded

```
int yEncoded
```

Encoded value of the y component.

6.265.3.4 zEncoded

```
int zEncoded
```

Encoded value of the z component.

6.265.4 Property Documentation

6.265.4.1 W

```
float W [get], [set]
```

Gets or sets the w component.

6.265.4.2 X

```
float X [get], [set]
```

Gets or sets the x component.

6.265.4.3 Y

```
float Y [get], [set]
```

Gets or sets the y component.

6.265.4.4 Z

```
float Z [get], [set]
```

Gets or sets the z component.

6.266 RangeExAttribute Class Reference

Represents an attribute that specifies a range of values for a field or property.

Inherits [DrawerPropertyAttribute](#).

Public Member Functions

- [RangeExAttribute \(double min, double max\)](#)

Initializes a new instance of the [RangeExAttribute](#) class with the specified minimum and maximum values.

Public Attributes

- bool [ClampMax](#) = true

Gets or sets a value indicating whether the maximum value should be clamped.

- bool [ClampMin](#) = true

Gets or sets a value indicating whether the minimum value should be clamped.

- bool [UseSlider](#) = true

Gets or sets a value indicating whether a slider should be used for the range.

Properties

- double **Max** [get]
Gets the maximum value of the range.
- double **Min** [get]
Gets the minimum value of the range.

6.266.1 Detailed Description

Represents an attribute that specifies a range of values for a field or property.

6.266.2 Constructor & Destructor Documentation

6.266.2.1 RangeExAttribute()

```
RangeExAttribute (
    double min,
    double max )
```

Initializes a new instance of the [RangeExAttribute](#) class with the specified minimum and maximum values.

Parameters

<i>min</i>	The minimum value of the range.
<i>max</i>	The maximum value of the range.

6.266.3 Member Data Documentation

6.266.3.1 ClampMax

```
bool ClampMax = true
```

Gets or sets a value indicating whether the maximum value should be clamped.

6.266.3.2 ClampMin

```
bool ClampMin = true
```

Gets or sets a value indicating whether the minimum value should be clamped.

6.266.3.3 UseSlider

```
bool UseSlider = true
```

Gets or sets a value indicating whether a slider should be used for the range.

6.266.4 Property Documentation

6.266.4.1 Max

```
double Max [get]
```

Gets the maximum value of the range.

6.266.4.2 Min

```
double Min [get]
```

Gets the minimum value of the range.

6.267 ReadOnlyAttribute Class Reference

Attribute used to mark a field as read-only.

Inherits [DecoratingPropertyAttribute](#).

Properties

- bool `InEditMode` = true [get, set]
Should the field be read-only in edit mode?
- bool `InPlayMode` = true [get, set]
Should the field be read-only in play mode?

Additional Inherited Members

6.267.1 Detailed Description

Attribute used to mark a field as read-only.

6.267.2 Property Documentation

6.267.2.1 InEditMode

```
bool InEditMode = true [get], [set]
```

Should the field be read-only in edit mode?

6.267.2.2 InPlayMode

```
bool InPlayMode = true [get], [set]
```

Should the field be read-only in play mode?

6.268 ReadWriteUtils Class Reference

Provides utility methods for reading and writing data.

Static Public Member Functions

- static float [ReadFloat](#) (int *data)
Reads a float value from the provided memory location.
- static [NetworkBehaviour](#) [ReadNetworkBehaviourRef](#) (int *data, [NetworkRunner](#) runner, out bool isValid)
Reads a NetworkBehaviour reference from the provided memory location. Null is considered valid (0,1).
- static Quaternion [ReadQuaternion](#) (int *data)
Reads a Quaternion value from the provided memory location.
- static Vector2 [ReadVector2](#) (int *data)
Reads a Vector2 value from the provided memory location.
- static Vector3 [ReadVector3](#) (int *data)
Reads a Vector3 value from the provided memory location.
- static Vector4 [ReadVector4](#) (int *data)
Reads a Vector4 value from the provided memory location.
- static void [WriteEmptyNetworkBehaviourRef](#) (int *data)
Writes an empty NetworkBehaviour reference to the provided memory location.
- static void [WriteFloat](#) (int *data, float f)
Writes a float value to the provided memory location.
- static void [WriteNetworkBehaviourRef](#) (int *data, [NetworkRunner](#) runner, [NetworkBehaviour](#) reference)
Writes a NetworkBehaviour reference to the provided memory location.
- static void [WriteNullBehaviourRef](#) (int *data)
Writes a null NetworkBehaviour reference to the provided memory location.
- static void [WriteQuaternion](#) (int *data, Quaternion value)
Writes a Quaternion value to the provided memory location.
- static void [WriteVector2](#) (int *data, Vector2 value)
Writes a Vector2 value to the provided memory location.
- static void [WriteVector3](#) (int *data, Vector3 value)
Writes a Vector3 value to the provided memory location.
- static void [WriteVector4](#) (int *data, Vector4 value)
Writes a Vector4 value to the provided memory location.

Static Public Attributes

- const float **ACCURACY** = 1 << 10
Accuracy of floating point values when serialized.

6.268.1 Detailed Description

Provides utility methods for reading and writing data.

6.268.2 Member Function Documentation

6.268.2.1 ReadFloat()

```
static float ReadFloat (
    int * data ) [static]
```

Reads a float value from the provided memory location.

Parameters

<i>data</i>	The memory location to read from.
-------------	-----------------------------------

Returns

The float value read from the memory location.

6.268.2.2 ReadNetworkBehaviourRef()

```
static NetworkBehaviour ReadNetworkBehaviourRef (
    int * data,
    NetworkRunner runner,
    out bool isValid ) [static]
```

Reads a [NetworkBehaviour](#) reference from the provided memory location. Null is considered valid (0,1).

Parameters

<i>data</i>	The memory location to read from.
<i>runner</i>	The NetworkRunner associated with the NetworkBehaviour .
<i>isValid</i>	Out parameter indicating whether the read operation was valid.

Returns

The [NetworkBehaviour](#) reference read from the memory location.

6.268.2.3 ReadQuaternion()

```
static Quaternion ReadQuaternion (
    int * data ) [static]
```

Reads a Quaternion value from the provided memory location.

Parameters

<i>data</i>	The memory location to read from.
-------------	-----------------------------------

Returns

The Quaternion value read from the memory location.

6.268.2.4 ReadVector2()

```
static Vector2 ReadVector2 (
    int * data ) [static]
```

Reads a Vector2 value from the provided memory location.

Parameters

<i>data</i>	The memory location to read from.
-------------	-----------------------------------

Returns

The Vector2 value read from the memory location.

6.268.2.5 ReadVector3()

```
static Vector3 ReadVector3 (
    int * data ) [static]
```

Reads a Vector3 value from the provided memory location.

Parameters

<i>data</i>	The memory location to read from.
-------------	-----------------------------------

Returns

The Vector3 value read from the memory location.

6.268.2.6 ReadVector4()

```
static Vector4 ReadVector4 (
    int * data ) [static]
```

Reads a Vector4 value from the provided memory location.

Parameters

<i>data</i>	The memory location to read from.
-------------	-----------------------------------

Returns

The Vector4 value read from the memory location.

6.268.2.7 WriteEmptyNetworkBehaviourRef()

```
static void WriteEmptyNetworkBehaviourRef (
    int * data ) [static]
```

Writes an empty [NetworkBehaviour](#) reference to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
-------------	----------------------------------

6.268.2.8 WriteFloat()

```
static void WriteFloat (
    int * data,
    float f ) [static]
```

Writes a float value to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>f</i>	The float value to write.

6.268.2.9 WriteNetworkBehaviourRef()

```
static void WriteNetworkBehaviourRef (
    int * data,
    NetworkRunner runner,
    NetworkBehaviour reference ) [static]
```

Writes a [NetworkBehaviour](#) reference to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>runner</i>	The NetworkRunner associated with the NetworkBehaviour .
<i>reference</i>	The NetworkBehaviour reference to write.

6.268.2.10 WriteNullBehaviourRef()

```
static void WriteNullBehaviourRef (
    int * data ) [static]
```

Writes a null [NetworkBehaviour](#) reference to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
-------------	----------------------------------

6.268.2.11 WriteQuaternion()

```
static void WriteQuaternion (
    int * data,
    Quaternion value ) [static]
```

Writes a Quaternion value to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>value</i>	The Quaternion value to write.

6.268.2.12 WriteVector2()

```
static void WriteVector2 (
    int * data,
    Vector2 value ) [static]
```

Writes a Vector2 value to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>value</i>	The Vector2 value to write.

6.268.2.13 WriteVector3()

```
static void WriteVector3 (
    int * data,
    Vector3 value ) [static]
```

Writes a Vector3 value to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>value</i>	The Vector3 value to write.

6.268.2.14 WriteVector4()

```
static void WriteVector4 (
    int * data,
    Vector4 value ) [static]
```

Writes a Vector4 value to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>value</i>	The Vector4 value to write.

6.268.3 Member Data Documentation

6.268.3.1 ACCURACY

```
const float ACCURACY = 1 << 10 [static]
```

Accuracy of floating point values when serialized.

6.269 ReadWriteUtilsForWeaver Class Reference

Provides utility methods for reading and writing data.

Static Public Member Functions

- static unsafe int [GetByteArrayHashCode](#) (byte *ptr, int length)
Gets the byte count of a string in UTF8 format without a hash.
- static int [GetByteCountUtf8NoHash](#) (string value)
Gets the word count of a string with optional caching.
- static int [GetStringHashCode](#) (string value, int maxLength)
Reads a boolean value from the provided memory location.
- static int [GetWordCountString](#) (int capacity, bool withCaching)
Reads a string from the provided memory location in UTF32 format without a hash.
- static bool [ReadBoolean](#) (int *data)
Reads a string from the provided memory location in UTF32 format with a hash.
- static unsafe int [ReadStringUtf32NoHash](#) (int *ptr, int maxLength, out string result)
Reads a string from the provided memory location in UTF8 format without a hash.
- static unsafe int [ReadStringUtf32WithHash](#) (int *ptr, int maxLength, ref string cache)
Reads a string from the provided memory location in UTF8 format with a hash.
- static int [ReadStringUtf8NoHash](#) (void *source, out string result)
Verifies the byte count of a network unwrapped object. Throws an exception if the actual byte count exceeds the maximum allowed byte count.
- static int [VerifyRawNetworkUnwrap< T >](#) (int actual, int maxBytes)
Verifies the byte count of a network wrapped object. Throws an exception if the actual byte count exceeds the maximum allowed byte count.
- static int [VerifyRawNetworkWrap< T >](#) (int actual, int maxBytes)
Writes a boolean value to the provided memory location.
- static unsafe int [WriteStringUtf32NoHash](#) (int *ptr, int maxLength, string value)
Writes a string to the provided memory location in UTF32 format without a hash.
- static unsafe int [WriteStringUtf32WithHash](#) (int *ptr, int maxLength, string value, ref string cache)
Writes a string to the provided memory location in UTF32 format with a hash.
- static int [WriteStringUtf8NoHash](#) (void *destination, string str)
Writes a string to the provided memory location in UTF8 format without a hash.

6.269.1 Detailed Description

Provides utility methods for reading and writing data.

6.269.2 Member Function Documentation

6.269.2.1 GetByteArrayHashCode()

```
static unsafe int GetByteArrayHashCode (
    byte * ptr,
    int length ) [static]
```

Parameters

<i>ptr</i>	
<i>length</i>	

Returns

6.269.2.2 GetByteCountUtf8NoHash()

```
static int GetByteCountUtf8NoHash (
    string value ) [static]
```

Gets the byte count of a string in UTF8 format without a hash.

Parameters

<i>value</i>	The string to get the byte count of.
--------------	--------------------------------------

Returns

The byte count of the string in UTF8 format.

6.269.2.3 GetStringHashCode()

```
static int GetStringHashCode (
    string value,
    int maxLength ) [static]
```

Parameters

<i>value</i>	
<i>maxLength</i>	

Returns**6.269.2.4 GetWordCountString()**

```
static int GetWordCountString (
    int capacity,
    bool withCaching ) [static]
```

Gets the word count of a string with optional caching.

Parameters

<i>capacity</i>	The capacity of the string.
<i>withCaching</i>	Indicates whether caching is used.

Returns

The word count of the string.

6.269.2.5 ReadBoolean()

```
static bool ReadBoolean (
    int * data ) [static]
```

Reads a boolean value from the provided memory location.

Parameters

<i>data</i>	The memory location to read from.
-------------	-----------------------------------

Returns

The boolean value read from the memory location.

6.269.2.6 ReadStringUtf32NoHash()

```
static unsafe int ReadStringUtf32NoHash (
    int * ptr,
    int maxLength,
    out string result ) [static]
```

Reads a string from the provided memory location in UTF32 format without a hash.

Parameters

<i>ptr</i>	The memory location to read from.
<i>maxLength</i>	The maximum length of the string.
<i>result</i>	The string read from the memory location.

Returns

The number of bytes read.

6.269.2.7 ReadStringUtf32WithHash()

```
static unsafe int ReadStringUtf32WithHash (
    int * ptr,
    int maxLength,
    ref string cache ) [static]
```

Reads a string from the provided memory location in UTF32 format with a hash.

Parameters

<i>ptr</i>	The memory location to read from.
<i>maxLength</i>	The maximum length of the string.
<i>cache</i>	A reference to a cache string. This will be updated with the read string if it matches the cached hashcode.

Returns

The number of bytes read.

6.269.2.8 ReadStringUtf8NoHash()

```
static int ReadStringUtf8NoHash (
    void * source,
    out string result ) [static]
```

Reads a string from the provided memory location in UTF8 format without a hash.

Parameters

<i>source</i>	The memory location to read from.
<i>result</i>	The string read from the memory location.

Returns

The number of bytes read.

6.269.2.9 VerifyRawNetworkUnwrap< T >()

```
static int VerifyRawNetworkUnwrap< T > (
    int actual,
    int maxBytes ) [static]
```

Verifies the byte count of a network unwrapped object. Throws an exception if the actual byte count exceeds the maximum allowed byte count.

Template Parameters

<i>T</i>	The type of the network unwrapped object.
----------	---

Parameters

<i>actual</i>	The actual byte count.
<i>maxBytes</i>	The maximum allowed byte count.

Returns

The actual byte count if it does not exceed the maximum allowed byte count.

6.269.2.10 VerifyRawNetworkWrap< T >()

```
static int VerifyRawNetworkWrap< T > (
    int actual,
    int maxBytes ) [static]
```

Verifies the byte count of a network wrapped object. Throws an exception if the actual byte count exceeds the maximum allowed byte count.

Template Parameters

<i>T</i>	The type of the network wrapped object.
----------	---

Parameters

<i>actual</i>	The actual byte count.
<i>maxBytes</i>	The maximum allowed byte count.

Returns

The actual byte count if it does not exceed the maximum allowed byte count.

6.269.2.11 WriteBoolean()

```
static void WriteBoolean (
    int * data,
    bool value ) [static]
```

Writes a boolean value to the provided memory location.

Parameters

<i>data</i>	The memory location to write to.
<i>value</i>	The boolean value to write.

6.269.2.12 WriteStringUtf32NoHash()

```
static unsafe int WriteStringUtf32NoHash (
    int * ptr,
    int maxLength,
    string value ) [static]
```

Writes a string to the provided memory location in UTF32 format without a hash.

Parameters

<i>ptr</i>	The memory location to write to.
<i>maxLength</i>	The maximum length of the string.
<i>value</i>	The string to write.

Returns

The number of bytes written.

6.269.2.13 WriteStringUtf32WithHash()

```
static unsafe int WriteStringUtf32WithHash (
    int * ptr,
    int maxLength,
    string value,
    ref string cache ) [static]
```

Writes a string to the provided memory location in UTF32 format with a hash.

Parameters

<i>ptr</i>	The memory location to write to.
<i>maxLength</i>	The maximum length of the string.
<i>value</i>	The string to write.
<i>cache</i>	A reference to a cache string. This will be updated with the trimmed value of the input string.

Returns

The number of bytes written.

6.269.2.14 WriteStringUtf8NoHash()

```
static int WriteStringUtf8NoHash (
    void * destination,
    string str ) [static]
```

Writes a string to the provided memory location in UTF8 format without a hash.

Parameters

<i>destination</i>	The memory location to write to.
<i>str</i>	The string to write.

Returns

The number of bytes written.

6.270 ReflectionUtils Class Reference

Provides utility methods for reflection.

Static Public Member Functions

- static [IEnumerable< Type > GetAllNetworkBehaviourTypes \(\)](#)
Gets all types that are assignable from NetworkBehaviour from all assemblies.
- static [IEnumerable< Type > GetAllSimulationBehaviourTypes \(\)](#)
Gets all types that are assignable from SimulationBehaviour from all assemblies.
- static [IEnumerable< Assembly > GetAllWeavedAssemblies \(\)](#)
Gets all assemblies that have been weaved.
- static [IEnumerable< Type > GetAllWeavedNetworkBehaviourTypes \(\)](#)
Gets all types that are assignable from NetworkBehaviour from all weaved assemblies.
- static [IEnumerable< Type > GetAllWeavedSimulationBehaviourTypes \(\)](#)
Gets all types that are assignable from SimulationBehaviour from all weaved assemblies.

- static `IEnumerable< Type > GetAllWeaverGeneratedTypes ()`
Gets all types that have the `WeaverGeneratedAttribute` from all weaved assemblies.
- static `T GetCustomAttributeOrThrow< T >` (`this MemberInfo member, bool inherit`)
Retrieves a custom attribute of type `T` from the provided member.
- static `NetworkBehaviourWeavedAttribute GetWeavedAttributeOrThrow (Type type)`
Gets the `NetworkBehaviourWeavedAttribute` for the specified type. Throws an `InvalidOperationException` if the type has not been weaved.

6.270.1 Detailed Description

Provides utility methods for reflection.

6.270.2 Member Function Documentation

6.270.2.1 GetAllNetworkBehaviourTypes()

```
static IEnumerable<Type> GetAllNetworkBehaviourTypes () [static]
```

Gets all types that are assignable from `NetworkBehaviour` from all assemblies.

Returns

An `IEnumerable` of all types that are assignable from `NetworkBehaviour`.

6.270.2.2 GetAllSimulationBehaviourTypes()

```
static IEnumerable<Type> GetAllSimulationBehaviourTypes () [static]
```

Gets all types that are assignable from `SimulationBehaviour` from all assemblies.

Returns

An `IEnumerable` of all types that are assignable from `SimulationBehaviour`.

6.270.2.3 GetAllWeavedAssemblies()

```
static IEnumerable<Assembly> GetAllWeavedAssemblies () [static]
```

Gets all assemblies that have been weaved.

Returns

An `IEnumerable` of all weaved assemblies.

6.270.2.4 GetAllWeavedNetworkBehaviourTypes()

```
static IEnumerable<Type> GetAllWeavedNetworkBehaviourTypes () [static]
```

Gets all types that are assignable from [NetworkBehaviour](#) from all weaved assemblies.

Returns

An `IEnumerable` of all types that are assignable from [NetworkBehaviour](#) in weaved assemblies.

6.270.2.5 GetAllWeavedSimulationBehaviourTypes()

```
static IEnumerable<Type> GetAllWeavedSimulationBehaviourTypes () [static]
```

Gets all types that are assignable from [SimulationBehaviour](#) from all weaved assemblies.

Returns

An `IEnumerable` of all types that are assignable from [SimulationBehaviour](#) in weaved assemblies.

6.270.2.6 GetAllWeaverGeneratedTypes()

```
static IEnumerable<Type> GetAllWeaverGeneratedTypes () [static]
```

Gets all types that have the [WeaverGeneratedAttribute](#) from all weaved assemblies.

Returns

An `IEnumerable` of all types that have the [WeaverGeneratedAttribute](#) in weaved assemblies.

6.270.2.7 GetCustomAttributeOrThrow< T >()

```
static T GetCustomAttributeOrThrow< T > (
    this MemberInfo member,
    bool inherit ) [static]
```

Retrieves a custom attribute of type `T` from the provided member.

Template Parameters

<code>T</code>	The type of the attribute to retrieve. Must be a subclass of <code>Attribute</code> .
----------------	---

Parameters

<i>member</i>	The member to retrieve the attribute from.
<i>inherit</i>	Specifies whether to search this member's inheritance chain to find the attributes.

Returns

The custom attribute of type T.

Exceptions

<i>ArgumentOutOfRangeException</i>	Thrown when the provided member does not have an attribute of type T.
<i>InvalidOperationException</i>	Thrown when the provided member has more than one attribute of type T.

Type Constraints

T : Attribute

6.270.2.8 GetWeavedAttributeOrThrow()

```
static NetworkBehaviourWeavedAttribute GetWeavedAttributeOrThrow (
    Type type ) [static]
```

Gets the [NetworkBehaviourWeavedAttribute](#) for the specified type. Throws an [InvalidOperationException](#) if the type has not been weaved.

Parameters

<i>type</i>	The type to get the NetworkBehaviourWeavedAttribute for.
-------------	--

Returns

The [NetworkBehaviourWeavedAttribute](#) for the specified type.

Exceptions

<i>InvalidOperationException</i>	Thrown when the type has not been weaved.
----------------------------------	---

6.271 RenderAttribute Class Reference

Override default render settings for [Networked] properties.

Inherits Attribute.

Public Member Functions

- `RenderAttribute ()`
Default constructor for RenderAttribute
- `RenderAttribute (RenderTimeframe timeframe, RenderSource source)`
RenderAttribute Constructor

Properties

- string `Method` [get, set]
- `RenderSource Source` [get, set]
- `RenderTimeframe Timeframe` [get, set]

6.271.1 Detailed Description

Override default render settings for [Networked] properties.

6.271.2 Constructor & Destructor Documentation

6.271.2.1 `RenderAttribute()` [1/2]

```
RenderAttribute ( )
```

Default constructor for `RenderAttribute`

6.271.2.2 `RenderAttribute()` [2/2]

```
RenderAttribute (
    RenderTimeframe timeframe,
    RenderSource source )
```

`RenderAttribute` Constructor

Parameters

<code>timeframe</code>	<code>RenderTimeframe</code> reference
<code>source</code>	<code>RenderSource</code> reference

6.271.3 Property Documentation

6.271.3.1 Method

```
string Method [get], [set]
```

Override the default interpolation method for this property. The method's signature must match:

```
static T MethodName(T from, T to, float alpha) { /* ... */ }
```

6.271.3.2 Source

```
RenderSource Source [get], [set]
```

Force this property to be rendered using this [RenderSource](#) (in the chosen [RenderTimeframe](#)).

This setting is prioritized over [NetworkBehaviour](#) and [NetworkObject](#) overrides.

6.271.3.3 Timeframe

```
RenderTimeframe Timeframe [get], [set]
```

Force this property to be rendered in this [RenderTimeframe](#).

This setting is prioritized over [NetworkBehaviour](#) and [NetworkObject](#) overrides.

6.272 RenderTimeline Struct Reference

Can be used to acquire interpolated data for different points in time.

Static Public Member Functions

- static void [GetRenderBuffers](#) ([NetworkBehaviour](#) behaviour, [out NetworkBehaviourBuffer](#) from, [out NetworkBehaviourBuffer](#) to, [out float](#) alpha)

Get the render data for the given [NetworkBehaviour](#).

6.272.1 Detailed Description

Can be used to acquire interpolated data for different points in time.

6.272.2 Member Function Documentation

6.272.2.1 GetRenderBuffers()

```
static void GetRenderBuffers (
    NetworkBehaviour behaviour,
    out NetworkBehaviourBuffer from,
    out NetworkBehaviourBuffer to,
    out float alpha ) [static]
```

Get the render data for the given [NetworkBehaviour](#).

Parameters

<i>behaviour</i>	Network behaviour to get render data for.
<i>from</i>	Render data for the previous point in time.
<i>to</i>	Render data for the next point in time.
<i>alpha</i>	Interpolation alpha.

6.273 RenderWeavedAttribute Class Reference

Render Weaved Attribute

Inherits Attribute.

Public Member Functions

- `RenderWeavedAttribute ()`
RenderWeavedAttribute Constructor

6.273.1 Detailed Description

Render Weaved Attribute

6.273.2 Constructor & Destructor Documentation

6.273.2.1 RenderWeavedAttribute()

`RenderWeavedAttribute ()`

RenderWeavedAttribute Constructor

6.274 ResolveNetworkPrefabSourceAttribute Class Reference

Resolve Network Prefab Source Attribute

Inherits PropertyAttribute.

6.274.1 Detailed Description

Resolve Network Prefab Source Attribute

6.275 RpcAttribute Class Reference

Flags a method as being a networked Remote Procedure Call. Only usable in a [NetworkBehaviour](#). Calls to this method (from the indicated allowed [RpcSources](#)) will generate a network message, which will execute the method remotely on the indicated [RpcTargets](#). The RPC method can include an empty [RpclInfo](#) argument, that will include meta information about the RPC on the receiving peer.

Inherits Attribute.

Public Member Functions

- [RpcAttribute \(\)](#)
Constructor for RpcAttributes.
- [RpcAttribute \(RpcSources sources, RpcTargets targets\)](#)
Constructor for RpcAttributes.

Static Public Attributes

- const int [MaxPayloadSize = SimulationMessage.MAX_PAYLOAD_SIZE](#)
Maximum allowed size for the payload of the RPC message.

Properties

- [RpcChannel Channel = RpcChannel.Reliable \[get, set\]](#)
Specifies which RpcChannel to use. Default value is [RpcChannel.Reliable](#).
- [RpcHostMode HostMode = RpcHostMode.SourcesIsServer \[get, set\]](#)
Options for when the game is run in [SimulationModes.Host](#) mode and RPC is invoked by the host.
- bool [InvokeLocal = true \[get, set\]](#)
Indicates if the method should be called locally (on the RPC caller). This happens immediately. Default value is true.
- int [Sources \[get\]](#)
The legal [RpcSources](#) types that can trigger this Rpc. Cast to int. Default value is (int)[RpcSources.All](#).
- int [Targets \[get\]](#)
The [RpcTargets](#) types that will receive and invoke this method. Cast to int. Default value is (int)[RpcTargets.All](#).
- bool [TickAligned = true \[get, set\]](#)
Indicates if this RPC's execution will be postponed until the local simulation catches up with the sender's [Tick](#) number. Even if set to false, the order of Rpcs is always preserved. Rpcs are deferred until all preceding Rpcs have executed. Default value is true.

6.275.1 Detailed Description

Flags a method as being a networked Remote Procedure Call. Only usable in a [NetworkBehaviour](#). Calls to this method (from the indicated allowed [RpcSources](#)) will generate a network message, which will execute the method remotely on the indicated [RpcTargets](#). The RPC method can include an empty [RpclInfo](#) argument, that will include meta information about the RPC on the receiving peer.

Example:

```
| [Rpc(RpcSources.All, RpcTargets.All, InvokeLocal = false, InvokeResim =  
false, Channel = RpcChannel.Reliable, TickAligned = true)]  
| public void RPC_Configure(NetworkObject no, string name, Color color, RpclInfo  
info = default) {} To target a specific Player, use the RpcTargetAttribute: | [Rpc] | public  
void RpcFoo([RpcTarget] PlayerRef targetPlayer) {} Use RpclInfo as a return value  
to access meta information about the RPC send attempt, such as failure to send reasons, message size, etc.
```

Non-static RPCs are only valid on a [NetworkBehaviour](#). Static RPCs can be implemented on [SimulationBehaviours](#), and do not require a [NetworkObject](#) instance. Static RPC require the first argument to be [NetworkRunner](#).

Static RPC Example: | [Rpc] | public static void RPC_Configure ([NetworkRunner](#)
runner) {}

6.275.2 Constructor & Destructor Documentation

6.275.2.1 RpcAttribute() [1/2]

```
RpcAttribute ( )
```

Constructor for RpcAttributes.

6.275.2.2 RpcAttribute() [2/2]

```
RpcAttribute (
    RpcSources sources,
    RpcTargets targets )
```

Constructor for RpcAttributes.

Parameters

<code>sources</code>	The legal RpcSources types that can trigger this Rpc. Default is RpcSources.All
<code>targets</code>	The RpcTargets types that will receive and invoke this method. Default is RpcTargets.All

6.275.3 Member Data Documentation

6.275.3.1 MaxPayloadSize

```
const int MaxPayloadSize = SimulationMessage.MAX\_PAYLOAD\_SIZE [static]
```

Maximum allowed size for the payload of the RPC message.

6.275.4 Property Documentation

6.275.4.1 Channel

```
RpcChannel Channel = RpcChannel.Reliable [get], [set]
```

Specifies which RpcChannel to use. Default value is [RpcChannel.Reliable](#)

6.275.4.2 HostMode

```
RpcHostMode HostMode = RpcHostMode.SourceIsServer [get], [set]
```

Options for when the game is run in [SimulationModes.Host](#) mode and RPC is invoked by the host.

6.275.4.3 InvokeLocal

```
bool InvokeLocal = true [get], [set]
```

Indicates if the method should be called locally (on the RPC caller). This happens immediately. Default value is true.

6.275.4.4 Sources

```
int Sources [get]
```

The legal [RpcSources](#) types that can trigger this Rpc. Cast to int. Default value is (int)[RpcSources.All](#).

6.275.4.5 Targets

```
int Targets [get]
```

The [RpcTargets](#) types that will receive and invoke this method. Cast to int. Default value is (int)[RpcTargets.All](#).

6.275.4.6 TickAligned

```
bool TickAligned = true [get], [set]
```

Indicates if this RPC's execution will be postponed until the local simulation catches up with the sender's [Tick](#) number. Even if set to false, the order of Rpcs is always preserved. Rpcs are deferred until all preceding Rpcs have executed. Default value is true.

6.276 RpcHeader Struct Reference

Header for RPC messages.

Public Member Functions

- override string [ToString \(\)](#)
Returns a string that represents the current [RpcHeader](#).

Static Public Member Functions

- static [RpcHeader Create](#) (int staticRpcKey)
Creates a new [RpcHeader](#) with the provided staticRpcKey.
- static [RpcHeader Create](#) ([NetworkId](#) id, int behaviour, int method)
Creates a new [RpcHeader](#) with the provided [NetworkId](#), behaviour, and method.
- static [RpcHeader Read](#) (byte *data, out int size)
Reads the [RpcHeader](#) from the provided byte pointer.
- static int [ReadSize](#) (byte *data)
Reads the size of the [RpcHeader](#) from the provided byte pointer.
- static int [Write](#) ([RpcHeader](#) header, byte *data)
Writes the [RpcHeader](#) to the provided byte pointer.

Public Attributes

- ushort [Behaviour](#)
The behaviour associated with the RPC message.
- ushort [Method](#)
The method associated with the RPC message.
- [NetworkId Object](#)
The [NetworkId](#) of the object associated with the RPC message.

Static Public Attributes

- const int [SIZE](#) = NetworkId.SIZE + 2 + 2
The size of the [RpcHeader](#) structure in bytes.

6.276.1 Detailed Description

Header for RPC messages.

6.276.2 Member Function Documentation

6.276.2.1 Create() [1/2]

```
static RpcHeader Create (
    int staticRpcKey )  [static]
```

Creates a new [RpcHeader](#) with the provided staticRpcKey.

Parameters

<i>staticRpcKey</i>	The staticRpcKey associated with the RPC message.
---------------------	---

Returns

Returns a new [RpcHeader](#) with the provided staticRpcKey.

6.276.2.2 Create() [2/2]

```
static RpcHeader Create (
    NetworkId id,
    int behaviour,
    int method )  [static]
```

Creates a new [RpcHeader](#) with the provided [NetworkId](#), behaviour, and method.

Parameters

<i>id</i>	The NetworkId of the object associated with the RPC message.
<i>behaviour</i>	The behaviour associated with the RPC message.
<i>method</i>	The method associated with the RPC message.

Returns

Returns a new [RpcHeader](#) with the provided parameters.

6.276.2.3 Read()

```
static RpcHeader Read (
    byte * data,
    out int size )  [static]
```

Reads the [RpcHeader](#) from the provided byte pointer.

Parameters

<i>data</i>	The byte pointer to read the RpcHeader from.
<i>size</i>	The size of the RpcHeader structure in bytes.

Returns

Returns the [RpcHeader](#) read from the byte pointer.

6.276.2.4 ReadSize()

```
static int ReadSize (
    byte * data ) [static]
```

Reads the size of the [RpcHeader](#) from the provided byte pointer.

Parameters

<i>data</i>	The byte pointer to read the RpcHeader size from.
-------------	---

Returns

Returns the size of the [RpcHeader](#) structure in bytes.

6.276.2.5 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [RpcHeader](#).

Returns

Returns a string that represents the current [RpcHeader](#).

6.276.2.6 Write()

```
static int Write (
    RpcHeader header,
    byte * data ) [static]
```

Writes the [RpcHeader](#) to the provided byte pointer.

Parameters

<i>header</i>	The RpcHeader to write.
<i>data</i>	The byte pointer to write the RpcHeader to.

Returns

Returns the size of the [RpcHeader](#) structure in bytes.

6.276.3 Member Data Documentation

6.276.3.1 Behaviour

ushort [Behaviour](#)

The behaviour associated with the RPC message.

6.276.3.2 Method

ushort [Method](#)

The method associated with the RPC message.

6.276.3.3 Object

[NetworkId](#) [Object](#)

The [NetworkId](#) of the object associated with the RPC message.

6.276.3.4 SIZE

const int SIZE = NetworkId.SIZE + 2 + 2 [static]

The size of the [RpcHeader](#) structure in bytes.

6.277 RpcInfo Struct Reference

[RpcInfo](#) is a struct that contains information about the RPC message.

Public Member Functions

- override string [ToString](#) ()
Returns a string that represents the current [RpcInfo](#).

Static Public Member Functions

- static [RpcInfo](#) [FromLocal](#) ([NetworkRunner](#) runner, [RpcChannel](#) channel, [RpcHostMode](#) hostMode)
Creates a new [RpcInfo](#) instance for a local RPC message.
- static unsafe [RpcInfo](#) [FromMessage](#) ([NetworkRunner](#) runner, [SimulationMessage](#) *message, [RpcHostMode](#) hostMode)
Creates a new [RpcInfo](#) instance from a [SimulationMessage](#).

Public Attributes

- [RpcChannel Channel](#)
Represents the channel through which the RPC message was sent.
- [bool IsInvokeLocal](#)
Indicates whether the RPC message is invoked locally.
- [PlayerRef Source](#)
Represents the player who sent the RPC message.
- [Tick Tick](#)
Represents the tick at which the RPC message was sent.

6.277.1 Detailed Description

[RpcInfo](#) is a struct that contains information about the RPC message.

6.277.2 Member Function Documentation

6.277.2.1 FromLocal()

```
static RpcInfo FromLocal (
    NetworkRunner runner,
    RpcChannel channel,
    RpcHostMode hostMode ) [static]
```

Creates a new [RpcInfo](#) instance for a local RPC message.

Parameters

<i>runner</i>	The NetworkRunner associated with the RPC message.
<i>channel</i>	The RpcChannel through which the RPC message was sent.
<i>hostMode</i>	The RpcHostMode of the RPC message.

Returns

Returns a new [RpcInfo](#) instance with the provided parameters.

6.277.2.2 FromMessage()

```
static unsafe RpcInfo FromMessage (
    NetworkRunner runner,
    SimulationMessage * message,
    RpcHostMode hostMode ) [static]
```

Creates a new [RpcInfo](#) instance from a [SimulationMessage](#).

Parameters

<i>runner</i>	The NetworkRunner associated with the RPC message.
<i>message</i>	The SimulationMessage from which to create the RpcInfo instance.
<i>hostMode</i>	The RpcHostMode of the RPC message.

Returns

Returns a new [RpcInfo](#) instance with the provided parameters.

6.277.2.3 [ToString\(\)](#)

```
override string ToString ()
```

Returns a string that represents the current [RpcInfo](#).

Returns

Returns a string that represents the current [RpcInfo](#).

6.277.3 Member Data Documentation

6.277.3.1 [Channel](#)

```
RpcChannel Channel
```

Represents the channel through which the RPC message was sent.

6.277.3.2 [IsInvokeLocal](#)

```
bool IsInvokeLocal
```

Indicates whether the RPC message is invoked locally.

6.277.3.3 [Source](#)

```
PlayerRef Source
```

Represents the player who sent the RPC message.

6.277.3.4 Tick

`Tick` `Tick`

Represents the tick at which the RPC message was sent.

6.278 RpcInvokeData Struct Reference

Represents the data required to invoke an RPC message.

Public Member Functions

- override string [ToString \(\)](#)
Returns a string that represents the current `RpcInvokeData`.

Public Attributes

- `RpcInvokeDelegate Delegate`
Represents the delegate to be invoked for the RPC message.
- int `Key`
Represents the key associated with the RPC message.
- int `Sources`
Represents the sources of the RPC message.
- int `Targets`
Represents the targets of the RPC message.

6.278.1 Detailed Description

Represents the data required to invoke an RPC message.

6.278.2 Member Function Documentation

6.278.2.1 `ToString()`

`override string ToString ()`

Returns a string that represents the current `RpcInvokeData`.

Returns

Returns a string that represents the current `RpcInvokeData`.

6.278.3 Member Data Documentation

6.278.3.1 Delegate

`RpcInvokeDelegate` Delegate

Represents the delegate to be invoked for the RPC message.

6.278.3.2 Key

`int Key`

Represents the key associated with the RPC message.

6.278.3.3 Sources

`int Sources`

Represents the sources of the RPC message.

6.278.3.4 Targets

`int Targets`

Represents the targets of the RPC message.

6.279 `RpcInvokeInfo` Struct Reference

May be used as an optional `RpcAttribute` return value. Contains meta data about the RPC send, such as failure to send reasons, culling, message size, etc.

Public Member Functions

- override string `ToString ()`
Returns a string that represents the current `RpcInvokeInfo`.

Public Attributes

- [RpcLocalInvokeResult LocalInvokeResult](#)
Represents the result of the local RPC invocation.
- [RpcSendCullResult SendCullResult](#)
Represents the result of the RPC message send operation.
- [RpcSendResult SendResult](#)
Contains detailed information about the RPC send operation result.

6.279.1 Detailed Description

May be used as an optional [RpcAttribute](#) return value. Contains meta data about the RPC send, such as failure to send reasons, culling, message size, etc.

Example:

```
| [Rpc] | public RpcInvokeInfo RpcFoo(int value) { | return default; | } | |
public override void FixedUpdateNetwork() { | var info = RpcFoo(); | Debug.←
Log(info); | }
```

6.279.2 Member Function Documentation

6.279.2.1 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [RpcInvokeInfo](#).

6.279.3 Member Data Documentation

6.279.3.1 LocalInvokeResult

```
RpcLocalInvokeResult LocalInvokeResult
```

Represents the result of the local RPC invocation.

6.279.3.2 SendCullResult

```
RpcSendCullResult SendCullResult
```

Represents the result of the RPC message send operation.

6.279.3.3 SendResult

[RpcSendResult](#) SendResult

Contains detailed information about the RPC send operation result.

6.280 RpcSendResult Struct Reference

RPC send operation result information.

Public Member Functions

- override string [ToString](#) ()
Returns a string that represents the current [RpcSendResult](#).

Public Attributes

- int [MessageSize](#)
The size of the RPC message.
- [RpcSendMessageResult](#) [Result](#)
Result flags for the RPC send operation.

6.280.1 Detailed Description

RPC send operation result information.

6.280.2 Member Function Documentation

6.280.2.1 [ToString\(\)](#)

override string [ToString](#) ()

Returns a string that represents the current [RpcSendResult](#).

6.280.3 Member Data Documentation

6.280.3.1 MessageSize

```
int MessageSize
```

The size of the RPC message.

6.280.3.2 Result

```
RpcSendMessageResult Result
```

Result flags for the RPC send operation.

6.281 RpcTargetAttribute Class Reference

RPC attribute used to indicate a specific target player for an RPC when sending from one player to another. RPC is sent to the server, and then is forwarded to the specified player. Usage:

Inherits Attribute.

Public Member Functions

- [RpcTargetAttribute \(\)](#)
RPC Attribute constructor.

6.281.1 Detailed Description

RPC attribute used to indicate a specific target player for an RPC when sending from one player to another. RPC is sent to the server, and then is forwarded to the specified player. Usage:

```
| [Rpc] | public void RpcFoo([RpcTarget] PlayerRef targetPlayer) { }
```

6.281.2 Constructor & Destructor Documentation

6.281.2.1 RpcTargetAttribute()

```
RpcTargetAttribute ( )
```

RPC Attribute constructor.

6.282 SceneLoadDoneArgs Struct Reference

Struct that contains information about a scene after it has been loaded.

Public Member Functions

- `SceneLoadDoneArgs (SceneRef sceneRef, NetworkObject[] sceneObjects, Scene scene=default, GameObject[] rootGameObjects=default)`
Constructs a new `SceneLoadDoneArgs` struct.

Public Attributes

- `readonly GameObject[] RootGameObjects`
Array of root GameObjects present in the loaded Unity scene.
- `readonly Scene Scene`
The loaded Unity scene.
- `readonly NetworkObject[] SceneObjects`
Array of NetworkObjects present in the loaded scene.
- `readonly SceneRef SceneRef`
Reference to the loaded scene.

6.282.1 Detailed Description

Struct that contains information about a scene after it has been loaded.

6.282.2 Constructor & Destructor Documentation

6.282.2.1 SceneLoadDoneArgs()

```
SceneLoadDoneArgs (
    SceneRef sceneRef,
    NetworkObject[] sceneObjects,
    Scene scene = default,
    GameObject[] rootGameObjects = default )
```

Constructs a new `SceneLoadDoneArgs` struct.

Parameters

<code>sceneRef</code>	Reference to the loaded scene.
<code>sceneObjects</code>	Array of NetworkObjects present in the loaded scene.
<code>scene</code>	The loaded Unity scene.
<code>rootGameObjects</code>	Array of root GameObjects present in the loaded Unity scene.

6.282.3 Member Data Documentation

6.282.3.1 RootGameObjects

```
readonly GameObject [ ] RootGameObjects
```

Array of root GameObjects present in the loaded Unity scene.

6.282.3.2 Scene

```
readonly Scene Scene
```

The loaded Unity scene.

6.282.3.3 SceneObjects

```
readonly NetworkObject [ ] SceneObjects
```

Array of NetworkObjects present in the loaded scene.

6.282.3.4 SceneRef

```
readonly SceneRef SceneRef
```

Reference to the loaded scene.

6.283 ScenePathAttribute Class Reference

Specifies that a string field represents a scene path.

Inherits [DrawerPropertyAttribute](#).

6.283.1 Detailed Description

Specifies that a string field represents a scene path.

6.284 SceneRef Struct Reference

Scene reference struct. Can be used to reference a scene by index or by path.

Inherits [INetworkStruct](#), and [IEquatable< SceneRef >](#).

Public Member Functions

- override bool [Equals](#) (object obj)
Determines whether the specified object is equal to the current [SceneRef](#).
- bool [Equals](#) ([SceneRef](#) other)
Determines whether the specified [SceneRef](#) is equal to the current [SceneRef](#).
- override int [GetHashCode](#) ()
Serves as the default hash function.
- bool [IsPath](#) (string path)
Checks if the [SceneRef](#) corresponds to a specific path.
- override string [ToString](#) ()
Returns a string that represents the current [SceneRef](#).
- string [ToString](#) (bool brackets, bool prefix)
Returns a string that represents the current [SceneRef](#), with optional formatting.

Static Public Member Functions

- static [SceneRef FromIndex](#) (int index)
Creates a [SceneRef](#) from an index.
- static [SceneRef FromPath](#) (string path)
Creates a scene ref from a path. The most common use case for this method is when using Unity's addressable scenes. The path is hashed (31 bit), so on rare occasion there may be a hash collision. In such case consider renaming a scene or construct your own hash and use [FromRaw](#). To check if a scene ref is was created for a specific path, use [IsPath](#).
- static [SceneRef FromRaw](#) (uint rawValue)
Creates a [SceneRef](#) from a raw value.
- static bool [operator!=](#) ([SceneRef](#) a, [SceneRef](#) b)
Returns true if the values are not equal.
- static bool [operator==](#) ([SceneRef](#) a, [SceneRef](#) b)
Returns true if the values are equal.

Public Attributes

- uint [RawValue](#)
The raw value of the [SceneRef](#). This can represent either an index or a path hash, depending on the flag.

Static Public Attributes

- const uint [FLAG_ADDRESSABLE](#) = 1u << 31
A constant representing the flag for addressable scenes.
- const int [SIZE](#) = 4
The size of the [SceneRef](#) structure in bytes.

Properties

- int [AsIndex](#) [get]
Returns lower 32 bits as an index.
- uint [AsPathHash](#) [get]
Gets the path hash of the [SceneRef](#).
- bool [IsIndex](#) [get]
Returns true if this scene ref is an index.
- bool [IsValid](#) [get]
If this scene index is valid
- static [SceneRef None](#) [get]
None scene

6.284.1 Detailed Description

Scene reference struct. Can be used to reference a scene by index or by path.

6.284.2 Member Function Documentation

6.284.2.1 Equals() [1/2]

```
override bool Equals (
    object obj )
```

Determines whether the specified object is equal to the current [SceneRef](#).

Parameters

<i>obj</i>	The object to compare with the current SceneRef .
------------	---

Returns

true if the specified object is equal to the current [SceneRef](#); otherwise, false.

6.284.2.2 Equals() [2/2]

```
bool Equals (
    SceneRef other )
```

Determines whether the specified [SceneRef](#) is equal to the current [SceneRef](#).

Parameters

<i>other</i>	The SceneRef to compare with the current SceneRef .
--------------	---

Returns

true if the specified [SceneRef](#) is equal to the current [SceneRef](#); otherwise, false.

6.284.2.3 FromIndex()

```
static SceneRef FromIndex (
    int index ) [static]
```

Creates a [SceneRef](#) from an index.

Parameters

<i>index</i>	The index to create the SceneRef from.
--------------	--

Returns

A [SceneRef](#) that represents the index.

Exceptions

ArgumentOutOfRangeException	Thrown when the index is less than 0 or equal to int.MaxValue.
---	--

6.284.2.4 FromPath()

```
static SceneRef FromPath (
    string path ) [static]
```

Creates a scene ref from a path. The most common use case for this method is when using Unity's addressable scenes. The path is hashed (31 bit), so on rare occasion there may be a hash collision. In such case consider renaming a scene or construct your own hash and use [FromRaw](#). To check if a scene ref was created for a specific path, use [IsPath](#).

Parameters

<i>path</i>	The path to create the SceneRef from.
-------------	---

Returns

A [SceneRef](#) that represents the path.

6.284.2.5 FromRaw()

```
static SceneRef FromRaw (
    uint rawValue ) [static]
```

Creates a [SceneRef](#) from a raw value.

Parameters

<i>rawValue</i>	The raw value to create the SceneRef from.
-----------------	--

Returns

A [SceneRef](#) that represents the raw value.

6.284.2.6 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current [SceneRef](#).

6.284.2.7 IsPath()

```
bool IsPath (
    string path )
```

Checks if the [SceneRef](#) corresponds to a specific path.

Parameters

<i>path</i>	The path to check.
-------------	--------------------

Returns

true if the [SceneRef](#) corresponds to the path; otherwise, false.

6.284.2.8 operator"!=()

```
static bool operator!= (
    SceneRef a,
    SceneRef b ) [static]
```

Returns true if the values are not equal.

Parameters

a	SceneRef a
b	SceneRef b

Returns

true if the values are not equal; otherwise, false.

6.284.2.9 operator==()

```
static bool operator== (
    SceneRef a,
    SceneRef b ) [static]
```

Returns true if the values are equal.

Parameters

a	SceneRef a
b	SceneRef b

Returns

true if the values are equal; otherwise, false.

6.284.2.10 ToString() [1/2]

```
override string ToString ( )
```

Returns a string that represents the current [SceneRef](#).

Returns

A string that represents the current [SceneRef](#).

6.284.2.11 ToString() [2/2]

```
string ToString (
    bool brackets,
    bool prefix )
```

Returns a string that represents the current [SceneRef](#), with optional formatting.

Parameters

<i>brackets</i>	If true, the string will be enclosed in brackets.
<i>prefix</i>	If true, the string will be prefixed with "Scene:".

Returns

A string that represents the current [SceneRef](#), formatted according to the provided parameters.

6.284.3 Member Data Documentation**6.284.3.1 FLAG_ADDRESSABLE**

```
const uint FLAG_ADDRESSABLE = 1u << 31 [static]
```

A constant representing the flag for addressable scenes.

6.284.3.2 RawValue

```
uint RawValue
```

The raw value of the [SceneRef](#). This can represent either an index or a path hash, depending on the flag.

6.284.3.3 SIZE

```
const int SIZE = 4 [static]
```

The size of the [SceneRef](#) structure in bytes.

6.284.4 Property Documentation

6.284.4.1 AsIndex

```
int AsIndex [get]
```

Returns lower 32 bits as an index.

6.284.4.2 AsPathHash

```
uint AsPathHash [get]
```

Gets the path hash of the [SceneRef](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the SceneRef is an index, not a path.
----------------------------------	---

6.284.4.3 IsIndex

```
bool IsIndex [get]
```

Returns true if this scene ref is an index.

6.284.4.4 IsValid

```
bool IsValid [get]
```

If this scene index is valid

6.284.4.5 None

```
SceneRef None [static], [get]
```

None scene

6.285 ScriptHelpAttribute Class Reference

Defines the appearance of the script header in the Unity inspector.

Inherits PropertyAttribute.

Properties

- `ScriptHeaderBackColor BackColor = ScriptHeaderBackColor.Gray [get, set]`
Color of the inspector header for this component type. None indicates no header graphic should be used.
- `bool Hide [get, set]`
Hide the script header in the Unity inspector.
- `ScriptHeaderStyle Style = ScriptHeaderStyle.Photon [get, set]`
- `string Url [get, set]`

6.285.1 Detailed Description

Defines the appearance of the script header in the Unity inspector.

6.285.2 Property Documentation

6.285.2.1 BackColor

```
ScriptHeaderBackColor BackColor = ScriptHeaderBackColor.Gray [get], [set]
```

Color of the inspector header for this component type. None indicates no header graphic should be used.

6.285.2.2 Hide

```
bool Hide [get], [set]
```

Hide the script header in the Unity inspector.

6.285.2.3 Style

```
ScriptHeaderStyle Style = ScriptHeaderStyle.Photon [get], [set]
```

6.285.2.4 Url

```
string Url [get], [set]
```

6.286 SerializableDictionary< TKey, TValue > Class Template Reference

A serializable dictionary.

Inherits SerializableDictionary, IDictionary< TKey, TValue >, and ISerializationCallbackReceiver.

Public Member Functions

- void [Add](#) (TKey key, TValue value)
Adds the specified key and value to the SerializableDictionary.
- virtual void [Clear](#) ()
Removes all keys and values from the SerializableDictionary.
- bool [ContainsKey](#) (TKey key)
Determines whether the SerializableDictionary contains the specified key.
- Dictionary< TKey, TValue >.Enumerator [GetEnumerator](#) ()
Returns an enumerator that iterates through the SerializableDictionary.
- bool [Remove](#) (TKey key)
Removes the value with the specified key from the SerializableDictionary.
- void [Reset](#) ()
Resets the SerializableDictionary, clearing its internal dictionary.
- void [Store](#) ()
Stores the SerializableDictionary's data into an array for serialization. This includes handling duplicates and null keys.
- bool [TryGetValue](#) (TKey key, out TValue value)
Gets the value associated with the specified key.

Static Public Member Functions

- static SerializableDictionary< TKey, TValue > [Create< TKey, TValue >](#) ()
Creates a new serializable dictionary.
- static SerializableDictionary< TKey, TValue > [Wrap](#) (Dictionary< TKey, TValue > dictionary)
Wraps an existing Dictionary into a SerializableDictionary.

Static Public Attributes

- const string [EntryKeyPropertyName](#) = nameof(Entry.Key)
The property path for the key in the Entry structure.
- const string [ItemsPropertyName](#) = nameof(_items)
The property path for the items in the SerializableDictionary.

Properties

- int **Count** [get]
Gets the number of key/value pairs contained in the [SerializableDictionary](#).
- bool **IsReadOnly** [get]
Gets a value indicating whether the [SerializableDictionary](#) is read-only. This value is always false.
- Dictionary< TKey, TValue >.KeyCollection **Keys** [get]
Gets a collection containing the keys in the [SerializableDictionary](#).
- TValue **this[TKey key]** [get, set]
Gets or sets the value associated with the specified key.
- Dictionary< TKey, TValue >.ValueCollection **Values** [get]
Gets a collection containing the values in the [SerializableDictionary](#).

6.286.1 Detailed Description

A serializable dictionary.

Template Parameters

<i>TKey</i>	The type of the dictionary key.
<i>TValue</i>	The type of the dictionary value.

This class is not thread-safe.

6.286.2 Member Function Documentation

6.286.2.1 Add()

```
void Add (
    TKey key,
    TValue value )
```

Adds the specified key and value to the [SerializableDictionary](#).

Parameters

<i>key</i>	The key of the element to add.
<i>value</i>	The value of the element to add.

6.286.2.2 Clear()

```
virtual void Clear ( ) [virtual]
```

Removes all keys and values from the [SerializableDictionary](#).

6.286.2.3 ContainsKey()

```
bool ContainsKey (
    TKey key )
```

Determines whether the [SerializableDictionary](#) contains the specified key.

Parameters

<i>key</i>	The key to locate in the SerializableDictionary .
------------	---

Returns

true if the [SerializableDictionary](#) contains an element with the specified key; otherwise, false.

6.286.2.4 Create< TKey, TValue >()

```
static SerializableDictionary< TKey, TValue > Create< TKey, TValue > ( ) [static]
```

Creates a new serializable dictionary.

Template Parameters

<i>TKey</i>	The type of the dictionary key.
<i>TValue</i>	The type of the dictionary value.

Returns

A new serializable dictionary.

6.286.2.5 GetEnumerator()

```
Dictionary< TKey, TValue >.Enumerator GetEnumerator ( )
```

Returns an enumerator that iterates through the [SerializableDictionary](#).

Returns

A `Dictionary{TKey,TValue}.Enumerator` structure for the [SerializableDictionary](#).

6.286.2.6 Remove()

```
bool Remove (
    TKey key )
```

Removes the value with the specified key from the [SerializableDictionary](#).

Parameters

<i>key</i>	The key of the element to remove.
------------	-----------------------------------

Returns

true if the element is successfully found and removed; otherwise, false. This method returns false if key is not found in the [SerializableDictionary](#).

6.286.2.7 Reset()

```
void Reset ( )
```

Resets the [SerializableDictionary](#), clearing its internal dictionary.

6.286.2.8 Store()

```
void Store ( )
```

Stores the [SerializableDictionary](#)'s data into an array for serialization. This includes handling duplicates and null keys.

6.286.2.9 TryGetValue()

```
bool TryGetValue (
    TKey key,
    out TValue value )
```

Gets the value associated with the specified key.

Parameters

<i>key</i>	The key of the value to get.
<i>value</i>	When this method returns, contains the value associated with the specified key, if the key is found; otherwise, the default value for the type of the value parameter. This parameter is passed uninitialized.

Returns

true if the [SerializableDictionary](#) contains an element with the specified key; otherwise, false.

6.286.2.10 Wrap()

```
static SerializableDictionary<TKey, TValue> Wrap (
    Dictionary< TKey, TValue > dictionary ) [static]
```

Wraps an existing Dictionary into a [SerializableDictionary](#).

Parameters

<i>dictionary</i>	The Dictionary to be wrapped.
-------------------	-------------------------------

Returns

A new [SerializableDictionary](#) that wraps the provided Dictionary.

6.286.3 Member Data Documentation

6.286.3.1 EntryKeyPropertyName

```
const string EntryKeyPropertyName = nameof(Entry.Key) [static]
```

The property path for the key in the Entry structure.

6.286.3.2 ItemsPropertyName

```
const string ItemsPropertyName = nameof(_items) [static]
```

The property path for the items in the [SerializableDictionary](#).

6.286.4 Property Documentation

6.286.4.1 Count

```
int Count [get]
```

Gets the number of key/value pairs contained in the [SerializableDictionary](#).

6.286.4.2 IsReadOnly

```
bool IsReadOnly [get]
```

Gets a value indicating whether the [SerializableDictionary](#) is read-only. This value is always false.

6.286.4.3 Keys

```
Dictionary< TKey, TValue >.KeyCollection Keys [get]
```

Gets a collection containing the keys in the [SerializableDictionary](#).

6.286.4.4 this[TKey key]

```
TValue this[TKey key] [get], [set]
```

Gets or sets the value associated with the specified key.

Parameters

key	The key of the value to get or set.
-----	-------------------------------------

Returns

The value associated with the specified key. If the specified key is not found, a get operation throws a `KeyNotFoundException`, and a set operation creates a new element with the specified key.

6.286.4.5 Values

```
Dictionary< TKey, TValue >.ValueCollection Values [get]
```

Gets a collection containing the values in the [SerializableDictionary](#).

6.287 SerializableType< BaseType > Struct Template Reference

A System.Type wrapper that can be serialized.

Inherits IEquatable< SerializableType >, and IEquatable< SerializableType< BaseType >>.

Public Member Functions

- `SerializableType AsShort ()`
Converts AssemblyQualifiedName and returns a short form, without version, culture etc.
- `SerializableType< BaseType > AsShort ()`
- `override bool Equals (object obj)`
Returns true if obj is SerializableType and the AssemblyQualifiedName is the same.
- `override bool Equals (object obj)`
- `bool Equals (SerializableType other)`
Returns true if the AssemblyQualifiedName is the same.
- `bool Equals (SerializableType< BaseType > other)`
- `override int GetHashCode ()`
Returns the hash code of the AssemblyQualifiedName.
- `override int GetHashCode ()`
- `SerializableType (string type)`
Create a new instance and stores type as AssemblyQualifiedName.
- `SerializableType (Type type)`
Create a new instance and stores full Type.AssemblyQualifiedName. To use shorter form, use AsShort.
- `SerializableType (Type type)`

Static Public Member Functions

- `static string GetShortAssemblyQualifiedName (Type type)`
Converts the Type.AssemblyQualifiedName to a shorter form, without version, culture etc.
- `static implicit operator SerializableType (Type type)`
Implicitly convert a Type to a SerializableType.
- `static implicit operator SerializableType< BaseType > (Type type)`
- `static implicit operator Type (SerializableType serializableType)`
Implicitly convert a SerializableType to a Type.
- `static implicit operator Type (SerializableType< BaseType > serializableType)`

Public Attributes

- `string AssemblyQualifiedName`
Type's assembly qualified name.

Static Public Attributes

- `static readonly Regex s_shortNameRegex = new Regex(@", (Version|Culture|PublicKeyToken)=[^, \"]+", RegexOptions.Compiled)`

Properties

- bool `IsValid` [get]
Is the type valid.
- Type `Value` [get]
Retrieve the type. The value is obtained using `Type.GetType(string)` and cached in a static

6.287.1 Detailed Description

A System.Type wrapper that can be serialized.

A generic version of [SerializableType](#) that can be used to store types that inherit from a specific base type.

Template Parameters

<code>BaseType</code>	The base type of the type stored
-----------------------	----------------------------------

6.287.2 Constructor & Destructor Documentation

6.287.2.1 SerializableType() [1/2]

```
SerializableType (
    Type type )
```

Create a new instance and stores full Type.AssemblyQualifiedName. To use shorter form, use [AsShort](#).

Parameters

<code>type</code>	Type to store. Can be null.
-------------------	-----------------------------

6.287.2.2 SerializableType() [2/2]

```
SerializableType (
    string type )
```

Create a new instance and stores `type` as [AssemblyQualifiedName](#).

Parameters

<code>type</code>	Type name.
-------------------	------------

6.287.3 Member Function Documentation

6.287.3.1 AsShort()

```
SerializableType AsShort ( )
```

Converts [AssemblyQualifiedName](#) and returns a short form, without version, culture etc.

6.287.3.2 Equals() [1/2]

```
override bool Equals (
    object obj )
```

Returns `true` if *obj* is [SerializableType](#) and the [AssemblyQualifiedName](#) is the same.

6.287.3.3 Equals() [2/2]

```
bool Equals (
    SerializableType< BaseType > other )
```

Returns `true` if the [AssemblyQualifiedName](#) is the same.

6.287.3.4 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code of the [AssemblyQualifiedName](#).

6.287.3.5 GetShortAssemblyQualifiedName()

```
static string GetShortAssemblyQualifiedName (
    Type type ) [static]
```

Converts the `Type.AssemblyQualifiedName` to a shorter form, without version, culture etc.

6.287.3.6 operator SerializableType()

```
static implicit operator SerializableType (
    Type type ) [static]
```

Implicitly convert a Type to a [SerializableType](#).

6.287.3.7 operator Type()

```
static implicit operator Type (
    SerializableType< BaseType > serializableType ) [static]
```

Implicitly convert a [SerializableType](#) to a Type.

6.287.4 Member Data Documentation

6.287.4.1 AssemblyQualifiedName

```
string AssemblyQualifiedName
```

Type's assembly qualified name.

6.287.5 Property Documentation

6.287.5.1 IsValid

```
bool IsValid [get]
```

Is the type valid.

6.287.5.2 Value

```
Type Value [get]
```

Retrieve the type. The value is obtained using Type.GetType(string) and cached in a static

6.288 SerializableTypeAttribute Class Reference

Specifies that either a string field represents a type name or sets additional options for [SerializableType](#) field.

Inherits [PropertyAttribute](#).

Properties

- Type [BaseType](#) [get, set]
The base type of the picked type.
- bool [UseFullAssemblyQualifiedName](#) [get, set]
Should the type be stored as a full assembly qualified name.
- bool [WarnIfNoPreserveAttribute](#) [get, set]
Should a warning be shown if the field does not have a PreserveAttribute.

6.288.1 Detailed Description

Specifies that either a string field represents a type name or sets additional options for [SerializableType](#) field.

6.288.2 Property Documentation

6.288.2.1 BaseType

Type [BaseType](#) [get], [set]

The base type of the picked type.

6.288.2.2 UseFullAssemblyQualifiedName

bool [UseFullAssemblyQualifiedName](#) [get], [set]

Should the type be stored as a full assembly qualified name.

6.288.2.3 WarnIfNoPreserveAttribute

bool [WarnIfNoPreserveAttribute](#) [get], [set]

Should a warning be shown if the field does not have a PreserveAttribute.

6.289 SerializeReferenceTypePickerAttribute Class Reference

Attribute used to show a type picker for a field with [SerializeReference].

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [SerializeReferenceTypePickerAttribute](#) (params Type[] types)

Initializes a new instance of the [SerializeReferenceTypePickerAttribute](#) class.

Public Attributes

- bool [GroupTypesByNamespace](#) = true
Should the types be grouped by namespace?
- bool [ShowFullName](#) = false
Should the full name be shown?

Properties

- Type[] [Types](#) [get]
Gets the types to be picked.

Additional Inherited Members

6.289.1 Detailed Description

Attribute used to show a type picker for a field with [SerializeReference].

6.289.2 Constructor & Destructor Documentation

6.289.2.1 SerializeReferenceTypePickerAttribute()

```
SerializeReferenceTypePickerAttribute (
    params Type[ ] types )
```

Initializes a new instance of the [SerializeReferenceTypePickerAttribute](#) class.

Parameters

<i>types</i>	The types to be picked.
--------------	-------------------------

6.289.3 Member Data Documentation

6.289.3.1 GroupTypesByNamespace

```
bool GroupTypesByNamespace = true
```

Should the types be grouped by namespace?

6.289.3.2 ShowFullName

```
bool ShowFullName = false
```

Should the full name be shown?

6.289.4 Property Documentation

6.289.4.1 Types

```
Type [ ] Types [get]
```

Gets the types to be picked.

6.290 SessionInfo Class Reference

Holds information about the Game Session

Public Member Functions

- override string [ToString \(\)](#)
String representation of a SessionInfo
- bool [UpdateCustomProperties \(Dictionary< string, SessionProperty > customProperties\)](#)
Update or change the Custom Properties of the current joined Room

Static Public Member Functions

- static implicit [operator bool \(SessionInfo sessionInfo\)](#)
Check if the SessionInfo reference is not Null and is Valid.

Properties

- bool?? **IsOpen** [get, set]
Signal if the current connected Room is open
- bool **IsValid** [get]
Flag to signal if the [SessionInfo](#) is ready for use
- bool?? **IsVisible** [get, set]
Signal if the current connected Room is visible
- int **MaxPlayers** [get]
Max number of peer that can join this Session, this value always include an extra slot for the Server/Host
- string **Name** [get]
Stores the current Room Name
- int **PlayerCount** [get]
Current number of peers inside this Session, this includes the Server/Host and Clients
- **ReadOnlyDictionary< string, SessionProperty > Properties** [get]
Room Custom Properties
- string **Region** [get]
Stores the current connected Region

6.290.1 Detailed Description

Holds information about the Game Session

6.290.2 Member Function Documentation

6.290.2.1 operator bool()

```
static implicit operator bool (
    SessionInfo sessionInfo ) [static]
```

Check if the [SessionInfo](#) reference is not Null and is Valid.

Parameters

<code>sessionInfo</code>	Session Info
--------------------------	--------------

6.290.2.2 ToString()

```
override string ToString ( )
```

String representation of a [SessionInfo](#)

Returns

Formatted [SessionInfo](#)

6.290.2.3 UpdateCustomProperties()

```
bool UpdateCustomProperties (
    Dictionary< string, SessionProperty > customProperties )
```

Update or change the Custom Properties of the current joined Room

Parameters

<i>customProperties</i>	New custom properties
-------------------------	-----------------------

6.290.3 Property Documentation

6.290.3.1 IsOpen

```
bool?? IsOpen [get], [set]
```

Signal if the current connected Room is open

6.290.3.2 IsValid

```
bool IsValid [get]
```

Flag to signal if the [SessionInfo](#) is ready for use

6.290.3.3 IsVisible

```
bool?? IsVisible [get], [set]
```

Signal if the current connected Room is visible

6.290.3.4 MaxPlayers

```
int MaxPlayers [get]
```

Max number of peer that can join this Session, this value always include an extra slot for the Server/Host

6.290.3.5 Name

```
string Name [get]
```

Stores the current Room Name

6.290.3.6 PlayerCount

```
int PlayerCount [get]
```

Current number of peers inside this Session, this includes the Server/Host and Clients

6.290.3.7 Properties

```
ReadOnlyDictionary<string, SessionProperty> Properties [get]
```

Room Custom Properties

6.290.3.8 Region

```
string Region [get]
```

Stores the current connected Region

6.291 Simulation Class Reference

Main simulation class

Inherits ILogSourceProxy, and INetPeerGroupCallbacks.

Classes

- struct [AreaOfInterest](#)
Area of Interest Definition

Public Member Functions

- void [GetAreaOfInterestGizmoData](#) (List<(Vector3 center, Vector3 size, int playerCount, int objectCount)> result)

Clears the provided list and populates it with data about the current Area of Interest (AOI) cells. Each tuple in the list represents one AOI cell, containing its center, size, player count, and object count.
- [PlayerRef GetInputAuthority](#) ([NetworkObject](#) networkObject)

Get the Input Authority [PlayerRef](#) for a [NetworkObject](#)
- [SimulationInput GetInputForPlayer](#) ([PlayerRef](#) player)

Get the [Simulation](#) Input for a specific Player
- void [GetObjectsAndPlayersInAreaOfInterestCell](#) (int cellKey, List< [PlayerRef](#) > players, List< [NetworkId](#) > objects)

Used by [RunnerAOIGizmos](#) component. Supplies data about current active AOI cells.
- List< [NetworkId](#) > [GetObjectsInAreaOfInterestForPlayer](#) ([PlayerRef](#) player)

Retrieves a list of network object IDs that are in the area of interest for the specified player.
- [PlayerRef GetStateAuthority](#) ([NetworkObject](#) networkObject)

Get the State Authority [PlayerRef](#) for a [NetworkObject](#)
- bool [HasAnyActiveConnections](#) ()

Signal if the Server has any Active Connection with any number of Clients.
- bool [IsInputAuthority](#) ([NetworkObject](#) networkObject, [PlayerRef](#) playerRef)

Check if a Player is the Input Authority over an [NetworkObject](#)
- bool? [IsInterestedIn](#) ([NetworkObject](#) obj, [PlayerRef](#) player)

Check if a [NetworkObject](#) is interested by a specific Player
- bool [IsLocalSimulationInputAuthority](#) ([NetworkObject](#) obj)

Check if the Local Player is the Input Authority over a [NetworkObject](#)
- bool [IsLocalSimulationStateAuthority](#) ([NetworkId](#) id)

Check if a Player is the State Authority over a [NetworkObject](#) by [NetworkId](#)
- bool [IsLocalSimulationStateAuthority](#) ([NetworkObject](#) obj)

Check if the Local Player is the State Authority over a [NetworkObject](#)
- bool [IsLocalSimulationStateOrInputSource](#) ([NetworkObject](#) obj)

Check if the Local Player is the State Authority or Input Authority over a [NetworkObject](#)
- bool [IsStateAuthority](#) ([NetworkObject](#) networkObject, [PlayerRef](#) playerRef)

Check if a Player is the State Authority over a [NetworkObject](#)
- bool [IsStateAuthority](#) ([PlayerRef](#) stateSource, [PlayerRef](#) playerRef)

Check if a Player is the State Authority in relation to another Player
- bool [TryGetHostPlayer](#) (out [PlayerRef](#) player)

Try to get the Host Player
- int [Update](#) (double dt)

Forwards the [Simulation](#) based on the Delta Time

Protected Member Functions

- virtual void [AfterSimulation](#) ()

Callback invoked after the [Simulation](#) Update
- virtual void [AfterUpdate](#) ()

Callback invoked After the [Simulation](#) Update
- virtual void [BeforeFirstTick](#) ()

Callback invoked before the First Tick
- virtual int [BeforeSimulation](#) ()

Callback invoked before the [Simulation](#) Loop
- virtual void [BeforeUpdate](#) ()

Callback invoked before the [Update](#)

- virtual void **NetworkConnected** (NetConnection *connection)

*Callback invoked before the **Simulation** Update*
- virtual void **NetworkDisconnected** (NetConnection *connection, **NetDisconnectReason** reason)

Callback invoked on the Connected
- virtual void **NetworkReceiveDone** ()

Callback invoked on the Disconnected
- virtual void **NoSimulation** ()

Callback invoked when the Network Receive is completed
- virtual void **NoSimulation** ()

Callback invoked when there is no simulation

Properties

- virtual IEnumerable< **PlayerRef** > **ActivePlayers** [get]

*List of Active players in the **Simulation***
- **SimulationConfig Config** [get]

*The **SimulationConfig** file used by this **Simulation**.*
- float **DeltaTime** [get]

*Gets the fixed tick time interval. Derived from the **SimulationRuntimeConfig.TickRate**.*
- int **InputCount** [get]

The current input collection size
- bool **IsClient** [get]

If this peer is a client. True for client peers in Server/Client topologies, and true for all peers in Shared Mode.
- bool **IsFirstTick** [get]

Use in conjunction with IsResimulation/IsForward inside of FixedUpdateNetwork to determine if the current tick being simulated is the first tick of the resimulation or forward phase of the simulation loop.
- bool **IsForward** [get]

Use inside of FixedUpdateNetwork to determine if the tick currently being simulated has NOT previously been simulated locally.
- bool **IsLastTick** [get]

Use in conjunction with IsResimulation/IsForward inside of FixedUpdateNetwork to determine if the current tick being simulated is the last tick of the resimulation or forward phase of the simulation loop.
- bool **IsLocalPlayerFirstExecution** [get]

True if the current stage of the simulation loop is Forward. False during resimulations.
- bool **IsMasterClient** [get]

Only valid in Shared Mode. Indicates if this peer is flagged as the MasterClient, which means it is default State Authority
- bool **IsPlayer** [get]

True for any peer that represents a human player. This is true for all peers except a dedicated server.
- bool **IsResimulation** [get]

Use inside of FixedUpdateNetwork to determine if the tick currently being simulated has previously been simulated locally. Resimulation occurs in client prediction when new states arrive from the StateAuthority. Networked objects are set to the most current authority state tick, and simulations are repeated from that tick to the local current tick.
- bool **IsRunning** [get]

*Signal if the **Simulation** is currently running*
- bool **IsServer** [get]

If this peer is the server. True for the Server or Host peer in Server/Client topologies, and always false for all peers in Shared Mode (the relay is the server).
- bool **IsShutdown** [get]
- bool **IsSinglePlayer** [get]

Indicates that this simulation is operating in Single Player mode, which is a Host that accepts no connections.
- abstract **Tick LatestServerTick** [get]

- latest tick on server we are aware of
- **NetAddress LocalAddress** [get]

Bound Address of the internal socket
- float **LocalAlpha** [get]
- abstract **PlayerRef LocalPlayer** [get]

Get LocalPlayer PlayerRef
- **SimulationModes Mode** [get]

Gets the SimulationModes flags for The type of network peer this simulation represents.
- **NetConfig * NetConfigPointer** [get]

Current NetConfig
- int **ObjectCount** [get]

Returns the number of objects in the simulation.
- **Dictionary< NetworkId, NetworkObjectMeta > Objects** [get]

Returns a map of all objects in the simulation.
- **NetworkProjectConfig ProjectConfig** [get]

The NetworkProjectConfig file used by this Simulation.
- float **RemoteAlpha** [get]

Remote Interpolation Alpha
- **Tick RemoteTick** [get]

Remote Tick
- **Tick RemoteTickPrevious** [get]

Remote previous Tick
- double **SendDelta** [get]

The packet send delta time
- int **SendRate** [get]

The packet send rate
- **SimulationStages Stage** [get]

Gets the current SimulationStages value.
- **Tick Tick** [get]

The tick associated with the current state of networked objects, or the current simulation tick being processed (when evaluated during FixedUpdateNetwork).
- double **TickDeltaDouble** [get]

The delta time of each tick as a double
- float **TickDeltaFloat** [get]

The delta time of each tick as a float
- **Tick TickPrevious** [get]

The previous tick
- int **TickRate** [get]

The current tick rate of the simulation
- int **TickStride** [get]

How large the ticks the current simulation takes are
- double **Time** [get]

The current simulation time in seconds
- **Topologies Topology** [get]

Indicates if a Server/Client or Shared Mode (relay server) topology is being used.

6.291.1 Detailed Description

Main simulation class

6.291.2 Member Function Documentation

6.291.2.1 AfterSimulation()

```
virtual void AfterSimulation ( ) [protected], [virtual]
```

Callback invoked after the [Simulation](#) Update

6.291.2.2 AfterUpdate()

```
virtual void AfterUpdate ( ) [protected], [virtual]
```

Callback invoked After the [Simulation](#) Update

6.291.2.3 BeforeFirstTick()

```
virtual void BeforeFirstTick ( ) [protected], [virtual]
```

Callback invoked before the First [Tick](#)

6.291.2.4 BeforeSimulation()

```
virtual int BeforeSimulation ( ) [protected], [virtual]
```

Callback invoked before the [Simulation](#) Loop

Returns

Total number of re-simulations

6.291.2.5 BeforeUpdate()

```
virtual void BeforeUpdate ( ) [protected], [virtual]
```

Callback invoked before the [Simulation](#) Update

6.291.2.6 GetAreaOfInterestGizmoData()

```
void GetAreaOfInterestGizmoData (   
    List<Vector3 center, Vector3 size, int playerCount, int objectCount> result )
```

Clears the provided list and populates it with data about the current Area of Interest (AOI) cells. Each tuple in the list represents one AOI cell, containing its center, size, player count, and object count.

Parameters

<i>result</i>	The list to be populated with AOI cell data.
---------------	--

6.291.2.7 GetInputAuthority()

```
PlayerRef GetInputAuthority (
    NetworkObject networkObject )
```

Get the Input Authority [PlayerRef](#) for a [NetworkObject](#)

Parameters

<i>networkObject</i>	NetworkObject to check
----------------------	------------------------

Returns

[PlayerRef](#) of the Input Authority for the [NetworkObject](#)

6.291.2.8 GetInputForPlayer()

```
SimulationInput GetInputForPlayer (
    PlayerRef player )
```

Get the [Simulation](#) Input for a specific Player

Parameters

<i>player</i>	Player to check for the Simulation Input
---------------	--

Returns

[Simulation](#) Input for a specific Player

6.291.2.9 GetObjectsAndPlayersInAreaOfInterestCell()

```
void GetObjectsAndPlayersInAreaOfInterestCell (
    int cellKey,
    List< PlayerRef > players,
    List< NetworkId > objects )
```

Used by RunnerAOIGizmos component. Supplies data about current active AOI cells.

6.291.2.10 GetObjectsInAreaOfInterestForPlayer()

```
List<NetworkId> GetObjectsInAreaOfInterestForPlayer (
    PlayerRef player )
```

Retrieves a list of network object IDs that are in the area of interest for the specified player.

Parameters

<code>player</code>	The player for whom the area of interest is being queried.
---------------------	--

Returns

A list of network object IDs in the area of interest for the player.

6.291.2.11 GetStateAuthority()

```
PlayerRef GetStateAuthority (
    NetworkObject networkObject )
```

Get the State Authority [PlayerRef](#) for a [NetworkObject](#)

Parameters

<code>networkObject</code>	NetworkObject to check
----------------------------	--

Returns

[PlayerRef](#) of the State Authority for the [NetworkObject](#)

6.291.2.12 HasAnyActiveConnections()

```
bool HasAnyActiveConnections ( )
```

Signal if the Server has any Active Connection with any number of Clients.

Returns

True, if at least one connection is active, false otherwise.

6.291.2.13 IsInputAuthority()

```
bool IsInputAuthority (
    NetworkObject networkObject,
    PlayerRef playerRef )
```

Check if a Player is the Input Authority over an [NetworkObject](#)

Parameters

<i>networkObject</i>	NetworkObject to check
<i>playerRef</i>	Player to check

Returns

True if the Player is the Input Authority over the [NetworkObject](#), false otherwise

6.291.2.14 IsInterestedIn()

```
bool? IsInterestedIn (
    NetworkObject obj,
    PlayerRef player )
```

Check if a [NetworkObject](#) is interested by a specific Player

Parameters

<i>obj</i>	NetworkObject to check
<i>player</i>	Player to check

Returns

True if the Player is interested in the [NetworkObject](#), false otherwise

6.291.2.15 IsLocalSimulationInputAuthority()

```
bool IsLocalSimulationInputAuthority (
    NetworkObject obj )
```

Check if the Local Player is the Input Authority over a [NetworkObject](#)

Parameters

<i>obj</i>	NetworkObject to check
------------	--

Returns

True if the Player is the Input Authority, false otherwise

6.291.2.16 IsLocalSimulationStateAuthority() [1/2]

```
bool IsLocalSimulationStateAuthority (
    NetworkId id )
```

Check if a Player is the State Authority over a [NetworkObject](#) by [NetworkId](#)

Parameters

<i>id</i>	NetworkId of the NetworkObject to check
-----------	---

Returns

True if the Player is the State Authority, false otherwise

6.291.2.17 IsLocalSimulationStateAuthority() [2/2]

```
bool IsLocalSimulationStateAuthority (
    NetworkObject obj )
```

Check if the Local Player is the State Authority over a [NetworkObject](#)

Parameters

<i>obj</i>	NetworkObject to check
------------	--

Returns

True if the Player is the State Authority, false otherwise

6.291.2.18 IsLocalSimulationStateOrInputSource()

```
bool IsLocalSimulationStateOrInputSource (
    NetworkObject obj )
```

Check if the Local Player is the State Authority or Input Authority over a [NetworkObject](#)

Parameters

<i>obj</i>	NetworkObject to check
------------	--

Returns

True if the Player is the State Authority or Input Authority, false otherwise

6.291.2.19 IsStateAuthority() [1/2]

```
bool IsStateAuthority (
    NetworkObject networkObject,
    PlayerRef playerRef )
```

Check if a Player is the State Authority over a NetworkObject

Parameters

<i>networkObject</i>	NetworkObject to check
<i>playerRef</i>	Player to check

Returns

True if the Player is the State Authority, false otherwise

6.291.2.20 IsStateAuthority() [2/2]

```
bool IsStateAuthority (
    PlayerRef stateSource,
    PlayerRef playerRef )
```

Check if a Player is the State Authority in relation to another Player

Parameters

<i>stateSource</i>	State Source Player
<i>playerRef</i>	Player to check

Returns

True if the Player is the State Authority, false otherwise

6.291.2.21 NetworkConnected()

```
virtual void NetworkConnected (
    NetConnection * connection ) [protected], [virtual]
```

Callback invoked on the Connected

Parameters

<i>connection</i>	Connection that was connected
-------------------	-------------------------------

6.291.2.22 NetworkDisconnected()

```
virtual void NetworkDisconnected (
    NetConnection * connection,
    NetDisconnectReason reason ) [protected], [virtual]
```

Callback invoked on the Disconnected

Parameters

<i>connection</i>	Connection that was disconnected
<i>reason</i>	Reason for the disconnection

6.291.2.23 NetworkReceiveDone()

```
virtual void NetworkReceiveDone ( ) [protected], [virtual]
```

Callback invoked when the Network Receive is completed

6.291.2.24 NoSimulation()

```
virtual void NoSimulation ( ) [protected], [virtual]
```

Callback invoked when there is no simulation

6.291.2.25 TryGetHostPlayer()

```
bool TryGetHostPlayer (
    out PlayerRef player )
```

Try to get the Host Player

Parameters

<i>player</i>	Host Player
---------------	-------------

Returns

True if the Host Player was found, false otherwise

6.291.2.26 Update()

```
int Update (
    double dt )
```

Forwards the [Simulation](#) based on the Delta Time

Parameters

<i>dt</i>	Delta Time used to forward the simulation
-----------	---

Returns

How many Ticks executed on this Update

6.291.3 Property Documentation

6.291.3.1 ActivePlayers

```
virtual IEnumerable<PlayerRef> ActivePlayers [get]
```

List of Active players in the [Simulation](#)

6.291.3.2 Config

```
SimulationConfig Config [get]
```

The [SimulationConfig](#) file used by this [Simulation](#).

6.291.3.3 DeltaTime

```
float DeltaTime [get]
```

Gets the fixed tick time interval. Derived from the [SimulationRuntimeConfig.TickRate](#).

6.291.3.4 InputCount

```
int InputCount [get]
```

The current input collection size

6.291.3.5 IsClient

```
bool IsClient [get]
```

If this peer is a client. True for client peers in Server/Client topologies, and true for all peers in Shared Mode.

6.291.3.6 IsFirstTick

```
bool IsFirstTick [get]
```

Use in conjunction with IsResimulation/IsForward inside of FixedUpdateNetwork to determine if the current tick being simulated is the first tick of the resimulation or forward phase of the simulation loop.

'Resimulation' describes simulating a tick that has been previously been simulated.

'Forward' describes simulating a tick that is being simulated for the first time locally.

'Prediction' describes simulating ticks higher than the most current known StateAuthority snapshot tick.

6.291.3.7 IsForward

```
bool IsForward [get]
```

Use inside of FixedUpdateNetwork to determine if the tick currently being simulated has NOT previously been simulated locally.

6.291.3.8 IsLastTick

```
bool IsLastTick [get]
```

Use in conjunction with IsResimulation/IsForward inside of FixedUpdateNetwork to determine if the current tick being simulated is the last tick of the resimulation or forward phase of the simulation loop.

'Resimulation' describes simulating a tick that has been previously been simulated.

'Forward' describes simulating a tick that is being simulated for the first time locally.

'Prediction' describes simulating ticks higher than the most current known StateAuthority snapshot tick.

6.291.3.9 IsLocalPlayerFirstExecution

```
bool IsLocalPlayerFirstExecution [get]
```

True if the current stage of the simulation loop is Forward. False during resimulations.

'Resimulation' describes simulating a tick that has been previously been simulated.

'Forward' describes simulating a tick that is being simulated for the first time locally.

'Prediction' describes simulating ticks higher than the most current known StateAuthority snapshot tick.

6.291.3.10 IsMasterClient

```
bool IsMasterClient [get]
```

Only valid in Shared Mode. Indicates if this peer is flagged as the MasterClient, which means it is default StateAuthority

6.291.3.11 IsPlayer

```
bool IsPlayer [get]
```

True for any peer that represents a human player. This is true for all peers except a dedicated server.

6.291.3.12 IsResimulation

```
bool IsResimulation [get]
```

Use inside of FixedUpdateNetwork to determine if the tick currently being simulated has previously been simulated locally. Resimulation occurs in client prediction when new states arrive from the StateAuthority. Networked objects are set to the most current authority state tick, and simulations are repeated from that tick to the local current tick.

6.291.3.13 IsRunning

```
bool IsRunning [get]
```

Signal if the [Simulation](#) is currently running

6.291.3.14 IsServer

```
bool IsServer [get]
```

If this peer is the server. True for the Server or Host peer in Server/Client topologies, and always false for all peers in Shared Mode (the relay is the server).

6.291.3.15 IsShutdown

```
bool IsShutdown [get]
```

6.291.3.16 IsSinglePlayer

```
bool IsSinglePlayer [get]
```

Indicates that this simulation is operating in Single Player mode, which is a Host that accepts no connections.

6.291.3.17 LatestServerTick

```
abstract Tick LatestServerTick [get]
```

latest tick on server we are aware of

6.291.3.18 LocalAddress

```
NetAddress LocalAddress [get]
```

Bound Address of the internal socket

6.291.3.19 LocalAlpha

```
float LocalAlpha [get]
```

6.291.3.20 LocalPlayer

```
abstract PlayerRef LocalPlayer [get]
```

Get LocalPlayer [PlayerRef](#)

6.291.3.21 Mode

`SimulationModes` Mode [get]

Gets the `SimulationModes` flags for The type of network peer this simulation represents.

6.291.3.22 NetConfigPointer

`NetConfig*` NetConfigPointer [get]

Current NetConfig

6.291.3.23 ObjectCount

`int` ObjectCount [get]

Returns the number of objects in the simulation.

6.291.3.24 Objects

`Dictionary<NetworkId, NetworkObjectMeta>` Objects [get]

Returns a map of all objects in the simulation.

6.291.3.25 ProjectConfig

`NetworkProjectConfig` ProjectConfig [get]

The `NetworkProjectConfig` file used by this `Simulation`.

6.291.3.26 RemoteAlpha

`float` RemoteAlpha [get]

Remote Interpolation Alpha

6.291.3.27 RemoteTick

`Tick` `RemoteTick` [get]

Remote [Tick](#)

6.291.3.28 RemoteTickPrevious

`Tick` `RemoteTickPrevious` [get]

Remote previous [Tick](#)

6.291.3.29 SendDelta

`double` `SendDelta` [get]

The packet send delta time

6.291.3.30 SendRate

`int` `SendRate` [get]

The packet send rate

6.291.3.31 Stage

`SimulationStages` `Stage` [get]

Gets the current [SimulationStages](#) value.

6.291.3.32 Tick

`Tick` `Tick` [get]

The tick associated with the current state of networked objects, or the current simulation tick being processed (when evaluated during `FixedUpdateNetwork`).

6.291.3.33 TickDeltaDouble

```
double TickDeltaDouble [get]
```

The delta time of each tick as a double

6.291.3.34 TickDeltaFloat

```
float TickDeltaFloat [get]
```

The delta time of each tick as a float

6.291.3.35 TickPrevious

```
Tick TickPrevious [get]
```

The previous tick

6.291.3.36 TickRate

```
int TickRate [get]
```

The current tick rate of the simulation

6.291.3.37 TickStride

```
int TickStride [get]
```

How large the ticks the current simulation takes are

6.291.3.38 Time

```
double Time [get]
```

The current simulation time in seconds

6.291.3.39 Topology

[Topologies](#) Topology [get]

Indicates if a Server/Client or Shared Mode (relay server) topology is being used.

6.292 Simulation.AreaOfInterest Struct Reference

Area of Interest Definition

Static Public Member Functions

- static int int int z **ClampCellCoords** (int **x**, int **y**, int **z**)
Get the size of each cell in the AOI grid.
- static int **GetCellSize** ()
Convert a sphere into a set of AOI cells.
- static void **SphereToCells** (Vector3 position, float radius, HashSet< int > cells)
Convert a position into the respective cell index.
- static int **ToCell** (int **x**, int **y**, int **z**)
Convert a cell coordinate into the respective cell index.
- static int **ToCell** (Vector3 position)
Convert a position into the respective cell index.
- static Vector3 **ToCellCenter** (int index)
Convert a cell index into the respective cell center position.
- static int int int z **ToCellCoords** (int index)
Get the size of the AOI grid.
- static int int int z **ToCellCoords** (Vector3 position)

Static Public Attributes

- static int **CELL_SIZE** = SIZE_DEFAULT
Size of each cell in the AOI grid.
- static int **x**
Get the size of the AOI grid.
- static int **y**

6.292.1 Detailed Description

Area of Interest Definition

6.292.2 Member Function Documentation

6.292.2.1 GetCellSize()

```
static int GetCellSize ( ) [static]
```

Get the size of each cell in the AOI grid.

Returns

The size of each cell in the AOI grid.

6.292.2.2 SphereToCells()

```
static void SphereToCells (
    Vector3 position,
    float radius,
    HashSet< int > cells ) [static]
```

Convert a sphere into a set of AOI cells.

Parameters

<i>position</i>	Sphere center
<i>radius</i>	Sphere radius. Max allowed radius is MAX_SHARED_RADIUS
<i>cells</i>	Resulting set of cells

6.292.2.3 ToCell() [1/2]

```
static int ToCell (
    int x,
    int y,
    int z ) [static]
```

Convert a cell coordinate into the respective cell index.

Parameters

<i>x</i>	X coordinate
<i>y</i>	Y coordinate
<i>z</i>	Z coordinate

Returns

Cell index

6.292.2.4 ToCell() [2/2]

```
static int ToCell (
    Vector3 position ) [static]
```

Convert a position into the respective cell index.

Parameters

<i>position</i>	Position
-----------------	----------

Returns

Cell index

6.292.2.5 ToCellCenter()

```
static Vector3 ToCellCenter (
    int index ) [static]
```

Convert a cell index into the respective cell center position.

Parameters

<i>index</i>	Cell index
--------------	------------

Returns

Cell center position

6.292.3 Member Data Documentation

6.292.3.1 CELL_SIZE

```
int CELL_SIZE = SIZE_DEFAULT [static]
```

Size of each cell in the AOI grid.

6.292.3.2 x

```
static int x [static]
```

Get the size of the AOI grid.

Clamp cell coordinates to the valid range.

Converts a cell index into its corresponding cell coordinates.

Convert a position into the respective cell coordinate.

Returns

The size of the AOI grid.

Parameters

<i>position</i>	Position
-----------------	----------

Returns

Cell coordinate

Parameters

<i>index</i>	The index of the cell to be converted.
--------------	--

Returns

A tuple containing the x, y, and z coordinates of the cell.

Parameters

<i>x</i>	X coordinate
<i>y</i>	Y coordinate
<i>z</i>	Z coordinate

Returns

Clamped cell coordinates

6.293 SimulationBehaviour Class Reference

Base class for a [Fusion](#) aware [Behaviour](#) (derived from `UnityEngine.MonoBehaviour`). If a [SimulationBehaviour](#) is found on a [NetworkRunner](#) game object during the runner initialisation, the [SimulationBehaviour](#) is automatically registered. Objects derived from this object can be associated with a [NetworkRunner](#) and [Simulation](#) using [NetworkRunner.AddGlobal\(\)](#).

Inherits [Behaviour](#).

Inherited by [HitboxManager](#), and [NetworkBehaviour](#).

Public Member Functions

- virtual void [FixedUpdateNetwork \(\)](#)
Fusion FixedUpdate timing callback.
- virtual void [Render \(\)](#)
Post simulation frame rendering callback. Runs after all simulations have finished. Use in place of Unity's Update when Fusion is handling Physics.

Properties

- bool [CanReceiveRenderCallback \[get\]](#)
Gets a value indicating whether this instance can receive render callbacks.
- bool [CanReceiveSimulationCallback \[get\]](#)
Gets a value indicating whether this instance can receive simulation callbacks.
- [NetworkObject Object \[get\]](#)
The NetworkObject this component is associated with.
- [NetworkRunner Runner \[get\]](#)
The NetworkRunner this component is associated with.

Additional Inherited Members

6.293.1 Detailed Description

Base class for a [Fusion](#) aware [Behaviour](#) (derived from `UnityEngine.MonoBehaviour`). If a [SimulationBehaviour](#) is found on a [NetworkRunner](#) game object during the runner initialisation, the [SimulationBehaviour](#) is automatically registered. Objects derived from this object can be associated with a [NetworkRunner](#) and [Simulation](#) using [NetworkRunner.AddGlobal\(\)](#).

6.293.2 Member Function Documentation

6.293.2.1 [FixedUpdateNetwork\(\)](#)

```
virtual void FixedUpdateNetwork ( ) [virtual]
```

[Fusion](#) FixedUpdate timing callback.

Reimplemented in [NetworkBehaviour](#).

6.293.2.2 [Render\(\)](#)

```
virtual void Render ( ) [virtual]
```

Post simulation frame rendering callback. Runs after all simulations have finished. Use in place of Unity's Update when [Fusion](#) is handling Physics.

Reimplemented in [NetworkTransform](#), and [NetworkMecanimAnimator](#).

6.293.3 Property Documentation

6.293.3.1 CanReceiveRenderCallback

```
bool CanReceiveRenderCallback [get]
```

Gets a value indicating whether this instance can receive render callbacks.

`true` if this instance can receive render callbacks; otherwise, `false`.

This property checks the current flags of the instance against various conditions to determine if it can receive render callbacks.

6.293.3.2 CanReceiveSimulationCallback

```
bool CanReceiveSimulationCallback [get]
```

Gets a value indicating whether this instance can receive simulation callbacks.

`true` if this instance can receive simulation callbacks; otherwise, `false`.

This property checks the current flags of the instance against various conditions to determine if it can receive simulation callbacks.

6.293.3.3 Object

```
NetworkObject Object [get]
```

The [NetworkObject](#) this component is associated with.

6.293.3.4 Runner

```
NetworkRunner Runner [get]
```

The [NetworkRunner](#) this component is associated with.

6.294 SimulationBehaviourAttribute Class Reference

Attribute for specifying which [SimulationStages](#) and [SimulationModes](#) this [SimulationBehaviour](#) will execute in. Can be used to limit execution to only Host, Server or Client peers, or to only execute on Resimulation or Forward ticks.
Usage:

Inherits Attribute.

Properties

- **SimulationModes Modes** [get, set]
Flag for which indicated peers in `SimulationModes` will execute this script.
- **SimulationStages Stages** [get, set]
Flag for which stages of the simulation loop this component will execute this script.
- **Topologies Topologies** [get, set]
Flag for which topologies this script will execute in

6.294.1 Detailed Description

Attribute for specifying which `SimulationStages` and `SimulationModes` this `SimulationBehaviour` will execute in. Can be used to limit execution to only Host, Server or Client peers, or to only execute on Resimulation or Forward ticks.
Usage:

```
[SimulationBehaviour(Stages = SimulationStages.Forward, Modes = SimulationModes.Server  
| SimulationModes.Host)]
```

6.294.2 Property Documentation

6.294.2.1 Modes

`SimulationModes Modes` [get], [set]

Flag for which indicated peers in `SimulationModes` will execute this script.

6.294.2.2 Stages

`SimulationStages Stages` [get], [set]

Flag for which stages of the simulation loop this component will execute this script.

6.294.2.3 Topologies

`Topologies Topologies` [get], [set]

Flag for which topologies this script will execute in

6.295 SimulationBehaviourListScope Struct Reference

Provides a scope for a SimulationBehaviourUpdater.BehaviourList, incrementing its lock count on creation and decrementing it on disposal. If the lock count reaches zero on disposal, all pending removals in the list are performed.

Inherits IDisposable.

Public Member Functions

- void [Dispose \(\)](#)
Dispose unmanaged resources.

6.295.1 Detailed Description

Provides a scope for a SimulationBehaviourUpdater.BehaviourList, incrementing its lock count on creation and decrementing it on disposal. If the lock count reaches zero on disposal, all pending removals in the list are performed.

6.295.2 Member Function Documentation

6.295.2.1 Dispose()

```
void Dispose ( )
```

Dispose unmanaged resources.

6.296 SimulationConfig Class Reference

Project configuration settings specific to how the [Simulation](#) class behaves.

Public Types

- enum class [DataConsistency](#)
- enum class [InputTransferModes](#)
- enum class [SimulationTimeMode](#)

The time mode that the [NetworkRunnerUpdaterDefault](#) uses to calculate the delta time for the simulation update.

Public Attributes

- bool [HostMigration](#)
If, in host mode, we should allow host migration if the current host leaves.
- int [InputDataWordCount](#)
Input Data Word Count
- [InputTransferModes](#) [InputTransferMode](#)
The way which input is transferred
- [DataConsistency](#) [ObjectDataConsistency](#)
How the server chooses to send [NetworkObject](#) updates to balance consistency and bandwidth consumption.
- int [PlayerCount](#) = 10
The default number of players allowed to join a game instance. Can also be changed in code when starting [Fusion](#).
- [NetworkProjectConfig](#).[ReplicationFeatures](#) [ReplicationFeatures](#) = [NetworkProjectConfig](#).[ReplicationFeatures](#).[Scheduling](#)
Features to enabled to replication such as area of interest, etc.
- [SimulationTimeMode](#) [SimulationUpdateTimeMode](#) = [SimulationTimeMode](#).[UnscaledDeltaTime](#)
The time mode that the Runner uses to update the simulation.
- [TickRate](#).[Selection](#) [TickRateSelection](#) = [TickRate](#).[Default](#)
The default tick rate to use. Can also be changed in code when starting [Fusion](#).
- [Topologies](#) [Topology](#)
The topology used

Properties

- bool [AreaOfInterestEnabled](#) [get]
Signal if AOI is enabled
- int [InputTotalWordCount](#) [get]
- bool [SchedulingEnabled](#) [get]
Signal if scheduling is enabled
- bool [SchedulingWithoutAOI](#) [get]
Signal if scheduling is running without AOI

6.296.1 Detailed Description

Project configuration settings specific to how the [Simulation](#) class behaves.

6.296.2 Member Enumeration Documentation

6.296.2.1 DataConsistency

```
enum DataConsistency [strong]
```

Enumerator

Full	When a NetworkBehaviour 's data changes, the server will send all properties whose changes have not been acknowledged. This option consumes more bandwidth, but guarantees that each NetworkBehaviour has consistent state.
Eventual	When a NetworkBehaviour 's data changes, the server will only send the newly changed properties. This option consumes less bandwidth, but a NetworkBehaviour may have inconsistent state at times (some properties up-to-date but not others).

6.296.2.2 InputTransferModes

```
enum InputTransferModes [strong]
```

Enumerator

Redundancy	Send delta compressed and redundant input, used for most games
RedundancyUncompressed	Send redundant input, use for games with small input structs (like fps games) and high player count
LatestState	Only send latest input state, useful for VR, etc.

6.296.2.3 SimulationTimeMode

```
enum SimulationTimeMode [strong]
```

The time mode that the [NetworkRunnerUpdaterDefault](#) uses to calculate the delta time for the simulation update.

Enumerator

UnscaledDeltaTime	Use Time.UnscaledDeltaTime. Time is not affected by timescale and keeps running when the editor is paused.
DeltaTime	Use Time.DeltaTime. Only for GameMode.Single . Can be used to allow for timescale changes in gameplay or to pause the game and continue without interruption when stopping at a debug breakpoint.

6.296.3 Member Data Documentation**6.296.3.1 HostMigration**

```
bool HostMigration
```

If, in host mode, we should allow host migration if the current host leaves.

6.296.3.2 InputDataWordCount

```
int InputDataWordCount
```

Input Data Word Count

6.296.3.3 InputTransferMode

```
InputTransferModes InputTransferMode
```

The way which input is transferred

6.296.3.4 ObjectDataConsistency

```
DataConsistency ObjectDataConsistency
```

How the server chooses to send [NetworkObject](#) updates to balance consistency and bandwidth consumption.

6.296.3.5 PlayerCount

```
int PlayerCount = 10
```

The default number of players allowed to join a game instance. Can also be changed in code when starting [Fusion](#).

6.296.3.6 ReplicationFeatures

```
NetworkProjectConfig.ReplicationFeatures ReplicationFeatures = NetworkProjectConfig.ReplicationFeatures.Schedu
```

Features to enabled to replication such as area of interest, etc.

6.296.3.7 SimulationUpdateTimeMode

```
SimulationTimeMode SimulationUpdateTimeMode = SimulationTimeMode.UnscaledDeltaTime
```

The time mode that the Runner uses to update the simulation.

6.296.3.8 TickRateSelection

```
TickRate.Selection TickRateSelection = TickRate.Default
```

The default tick rate to use. Can also be changed in code when starting [Fusion](#).

6.296.3.9 Topology

```
Topologies Topology
```

The topology used

6.296.4 Property Documentation

6.296.4.1 AreaOfInterestEnabled

```
bool AreaOfInterestEnabled [get]
```

Signal if AOI is enabled

6.296.4.2 InputTotalWordCount

```
int InputTotalWordCount [get]
```

6.296.4.3 SchedulingEnabled

```
bool SchedulingEnabled [get]
```

Signal if scheduling is enabled

6.296.4.4 SchedulingWithoutAOI

```
bool SchedulingWithoutAOI [get]
```

Signal if scheduling is running without AOI

6.297 SimulationInput Class Reference

[Simulation](#) Input

Classes

- class [Buffer](#)
Buffer for SimulationInputs.

Public Member Functions

- void [Clear](#) (int wordCount)
Clear a total of wordCount words from this input.
- void [CopyFrom](#) ([SimulationInput](#) source, int wordCount)
Copy wordCount words from source to this input.

Properties

- int * [Data](#) [get]
Data for this input.
- [SimulationInputHeader](#) * [Header](#) [get]
Header for this input.
- [PlayerRef](#) [Player](#) [get, set]
Player that owns this input.
- int [Sent](#) [get, set]
Simulation input sent count.

6.297.1 Detailed Description

[Simulation](#) Input

6.297.2 Member Function Documentation

6.297.2.1 Clear()

```
void Clear (
    int wordCount )
```

Clear a total of *wordCount* words from this input.

Parameters

wordCount	Word count to clear.
---------------------------	----------------------

6.297.2.2 CopyFrom()

```
void CopyFrom (
    SimulationInput source,
    int wordCount )
```

Copy *wordCount* words from *source* to this input.

Parameters

<i>source</i>	Input to copy from.
<i>wordCount</i>	Word count to copy.

6.297.3 Property Documentation

6.297.3.1 Data

```
int* Data [get]
```

Data for this input.

6.297.3.2 Header

```
SimulationInputHeader* Header [get]
```

Header for this input.

6.297.3.3 Player

```
PlayerRef Player [get], [set]
```

Player that owns this input.

6.297.3.4 Sent

```
int Sent [get], [set]
```

Simulation input sent count.

6.298 SimulationInput.Buffer Class Reference

Buffer for SimulationInputs.

Public Member Functions

- bool [Add](#) ([SimulationInput](#) input, double? insertTime=null)
Adds an input to the buffer.
- Buffer ([NetworkProjectConfig](#) cfg)
Creates a new Buffer.
- void [Clear](#) ()
Clears the buffer.
- bool [Contains](#) ([Tick](#) tick)
Whether the buffer contains an input for tick .
- int [CopySortedTo](#) ([SimulationInput](#)[] array)
Copies the buffer to an array and sorts it.
- [SimulationInput](#) [Get](#) ([Tick](#) tick)
Gets the input for tick .
- double? [GetInsertTime](#) ([Tick](#) tick)
Gets the insert time for tick .
- [SimulationInputHeader](#) [GetLastUsedInputHeader](#) ()
Retrieves the last used input header data.
- bool [Remove](#) ([Tick](#) tick, out [SimulationInput](#) removed)
Removes an input for tick from the buffer.

Properties

- int [Count](#) [get]
Number of inputs in the buffer.
- bool [Full](#) [get]
Whether the buffer is full.

6.298.1 Detailed Description

Buffer for SimulationInputs.

6.298.2 Constructor & Destructor Documentation

6.298.2.1 Buffer()

```
Buffer (
    NetworkProjectConfig cfg )
```

Creates a new Buffer.

Parameters

<i>cfg</i>	Network project configuration.
------------	--------------------------------

6.298.3 Member Function Documentation

6.298.3.1 Add()

```
bool Add (
    SimulationInput input,
    double? insertTime = null )
```

Adds an input to the buffer.

Parameters

<i>input</i>	Input to add.
<i>insertTime</i>	Insert time for <i>input</i> .

Returns

Whether the input was added.

6.298.3.2 Clear()

```
void Clear ( )
```

Clears the buffer.

6.298.3.3 Contains()

```
bool Contains (
    Tick tick )
```

Whether the buffer contains an input for *tick*.

Parameters

<i>tick</i>	Tick to check.
-------------	----------------

Returns

Whether the buffer contains an input for *tick*.

6.298.3.4 CopySortedTo()

```
int CopySortedTo (
    SimulationInput[] array )
```

Copies the buffer to an array and sorts it.

Parameters

<i>array</i>	Array to copy to.
--------------	-------------------

Returns

Number of elements copied.

6.298.3.5 Get()

```
SimulationInput Get (
    Tick tick )
```

Gets the input for *tick*.

Parameters

<i>tick</i>	Tick to get input for.
-------------	------------------------

Returns

Input for *tick*.

6.298.3.6 GetInsertTime()

```
double? GetInsertTime (
    Tick tick )
```

Gets the insert time for *tick*.

Parameters

<i>tick</i>	Tick to get insert time for.
-------------	--

Returns

Insert time for *tick*.

6.298.3.7 GetLastUsedInputHeader()

```
SimulationInputHeader GetLastUsedInputHeader ()
```

Retrieves the last used input header data.

Returns

The last used input header data.

6.298.3.8 Remove()

```
bool Remove (
    Tick tick,
    out SimulationInput removed )
```

Removes an input for *tick* from the buffer.

Parameters

<i>tick</i>	Tick to remove.
<i>removed</i>	Removed input.

Returns

Whether an input was removed.

6.298.4 Property Documentation**6.298.4.1 Count**

```
int Count [get]
```

Number of inputs in the buffer.

6.298.4.2 Full

```
bool Full [get]
```

Whether the buffer is full.

6.299 SimulationInputHeader Struct Reference

[Simulation](#) Input Header

Public Attributes

- float [InterpAlpha](#)
Interpolation alpha of the input.
- [Tick InterpFrom](#)
Interpolation from tick of the input.
- [Tick InterpTo](#)
Interpolation to tick of the input.
- [Tick Tick](#)
Tick of the input.

Static Public Attributes

- const int [SIZE](#) = [WORD_COUNT](#) * [Allocator.REPLICATE_WORD_SIZE](#)
Size of the header.
- const int [WORD_COUNT](#) = 4
Word count of the header.

6.299.1 Detailed Description

[Simulation](#) Input Header

6.299.2 Member Data Documentation

6.299.2.1 InterpAlpha

```
float InterpAlpha
```

Interpolation alpha of the input.

6.299.2.2 InterpFrom

`Tick` `InterpFrom`

Interpolation from tick of the input.

6.299.2.3 InterpTo

`Tick` `InterpTo`

Interpolation to tick of the input.

6.299.2.4 SIZE

```
const int SIZE = WORD_COUNT * Allocator.REPLICATE_WORD_SIZE [static]
```

Size of the header.

6.299.2.5 Tick

`Tick` `Tick`

`Tick` of the input.

6.299.2.6 WORD_COUNT

```
const int WORD_COUNT = 4 [static]
```

Word count of the header.

6.300 SimulationMessage Struct Reference

`Simulation` Message

Inherits ILogDumpable.

Public Member Functions

- void ILogDumpable. **Dump** (StringBuilder builder)
Get if a flag is set on this SimulationMessage
- bool **GetFlag** (int flag)
Signal if this SimulationMessage is Targeted
- bool **IsTargeted** ()
Signal if this SimulationMessage is Targeted
- void **ReferenceCountAdd** ()
Add a reference to this SimulationMessage
- bool **ReferenceCountSub** ()
Subtract a reference from this SimulationMessage
- void **SetDummy** ()
Set this SimulationMessage as Dummy
- void **SetNotTickAligned** ()
Set this SimulationMessage as Not Tick Aligned
- void **SetStatic** ()
Set this SimulationMessage as Static
- void **SetTarget** (PlayerRef target)
Set the player target of this SimulationMessage
- void **SetUnreliable** ()
Set this SimulationMessage as Unreliable
- override string **ToString** ()
Simulation Message ToString
- string **ToString** (bool useBrackets)
Simulation Message ToString

Static Public Member Functions

- static SimulationMessage * **Allocate** (Simulation sim, int capacityInBytes)
Allocate a new SimulationMessage
- static bool **CanAllocateUserPayload** (int capacityInBytes)
Checks if a message with given size can be allocated.
- static SimulationMessage * **Clone** (Simulation sim, SimulationMessage *message)
Create a copy of a SimulationMessage
- static byte * **GetData** (SimulationMessage *message)
Get the byte pointer content of a SimulationMessage
- static int **ReadInt** (SimulationMessage *message)
Read a int from a SimulationMessage
- static NetworkId **ReadNetworkedObjectRef** (SimulationMessage *message)
Read a NetworkId from a SimulationMessage
- static Vector3 **ReadVector3** (SimulationMessage *message)
Read a Vector3 from a SimulationMessage
- static void **WriteInt** (SimulationMessage *message, int value)
Write a int to a SimulationMessage
- static void **WriteNetworkedObjectRef** (SimulationMessage *message, NetworkId value)
Write a NetworkId to a SimulationMessage
- static void **WriteVector3** (SimulationMessage *message, Vector3 value)
Write a Vector3 to a SimulationMessage

Public Attributes

- int **Capacity**
Capacity in Bits of this [SimulationMessage](#)
- int **Flags**
Flags
- int **Offset**
Current offset in Bits
- int **References**
Reference Count
- **PlayerRef Source**
Source Player of this [SimulationMessage](#)
- **PlayerRef Target**
Target Player of this [SimulationMessage](#)
- int **Tick**
Tick of this [SimulationMessage](#)

Static Public Attributes

- const int **FLAG_DUMMY** = 1 << 8
Flag for dummy messages.
- const int **FLAG_INTERNAL** = 1 << 6
Flag for internal messages.
- const int **FLAG_NOT_TICK_ALIGNED** = 1 << 7
Flag for messages that are not tick aligned.
- const int **FLAG_REMOTE** = 1 << 1
Flag for remote messages.
- const int **FLAG_STATIC** = 1 << 2
Flag for static messages.
- const int **FLAG_TARGET_PLAYER** = 1 << 4
Flag for targeted messages to a player.
- const int **FLAG_TARGET_SERVER** = 1 << 5
Flag for targeted messages to the server.
- const int **FLAG_UNRELIABLE** = 1 << 3
Flag for unreliable messages.
- const int **FLAG_USER_FLAGS_START** = 1 << 16
Flag for user flags.
- const int **FLAG_USER_MESSAGE** = 1 << 0
Flag for user messages.
- const int **FLAGS_RESERVED** = 0xFFFF
Flag for reserved flags.
- const int **FLAGS_RESERVED_BITS** = 16
Flag for reserved bits.
- const int **MAX_PAYLOAD_SIZE** = 512
Max user message size in bytes.
- const int **SIZE** = 28
SimulationMessage size in bytes.

Properties

- bool `IsUnreliable` [get]
Signal if this `SimulationMessage` is Unreliable

6.300.1 Detailed Description

`Simulation` Message

6.300.2 Member Function Documentation

6.300.2.1 Allocate()

```
static SimulationMessage* Allocate (
    Simulation sim,
    int capacityInBytes ) [static]
```

Allocate a new `SimulationMessage`

Parameters

<code>sim</code>	<code>Simulation</code> to get the Memory from
<code>capacityInBytes</code>	Size in bytes of the new <code>SimulationMessage</code>

Returns

Pointer to the new `SimulationMessage`

6.300.2.2 CanAllocateUserPayload()

```
static bool CanAllocateUserPayload (
    int capacityInBytes ) [static]
```

Checks if a message with given size can be allocated.

6.300.2.3 Clone()

```
static SimulationMessage* Clone (
    Simulation sim,
    SimulationMessage * message ) [static]
```

Create a copy of a `SimulationMessage`

Parameters

<i>sim</i>	Simulation to allocate from
<i>message</i>	SimulationMessage to copy

Returns

Copy of the SimulationMessage

6.300.2.4 GetData()

```
static byte* GetData (
    SimulationMessage * message ) [static]
```

Get the byte pointer content of a SimulationMessage

Parameters

<i>message</i>	SimulationMessage to get the byte pointer of
----------------	--

Returns

Byte pointer of the SimulationMessage

6.300.2.5 GetFlag()

```
bool GetFlag (
    int flag )
```

Get if a flag is set on this SimulationMessage

Parameters

<i>flag</i>	Flag to check
-------------	---------------

Returns

True if the flag is set

6.300.2.6 IsTargeted()

```
bool IsTargeted ( )
```

Signal if this SimulationMessage is Targeted

Returns

True if this [SimulationMessage](#) is Targeted

6.300.2.7 ReadInt()

```
static int ReadInt (
    SimulationMessage * message ) [static]
```

Read a int from a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to read from
----------------	--

Returns

int read

6.300.2.8 ReadNetworkedObjectRef()

```
static NetworkId ReadNetworkedObjectRef (
    SimulationMessage * message ) [static]
```

Read a [NetworkId](#) from a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to read from
----------------	--

Returns

[NetworkId](#) read

6.300.2.9 ReadVector3()

```
static Vector3 ReadVector3 (
    SimulationMessage * message ) [static]
```

Read a Vector3 from a [SimulationMessage](#)

Parameters

<code>message</code>	SimulationMessage to read from
----------------------	--

Returns

Vector3 read

6.300.2.10 ReferenceCountAdd()

`void ReferenceCountAdd ()`

Add a reference to this [SimulationMessage](#)

6.300.2.11 ReferenceCountSub()

`bool ReferenceCountSub ()`

Subtract a reference from this [SimulationMessage](#)

Returns

True if the reference count is now 0

6.300.2.12 SetDummy()

`void SetDummy ()`

Set this [SimulationMessage](#) as Dummy

6.300.2.13 SetNotTickAligned()

`void SetNotTickAligned ()`

Set this [SimulationMessage](#) as Not Tick Aligned

6.300.2.14 SetStatic()

`void SetStatic ()`

Set this [SimulationMessage](#) as Static

6.300.2.15 SetTarget()

```
void SetTarget (
    PlayerRef target )
```

Set the player target of this [SimulationMessage](#)

Parameters

<i>target</i>	Target Player
---------------	---------------

6.300.2.16 SetUnreliable()

```
void SetUnreliable ( )
```

Set this [SimulationMessage](#) as Unreliable

6.300.2.17 ToString() [1/2]

```
override string ToString ( )
```

[Simulation Message](#) ToString

6.300.2.18 ToString() [2/2]

```
string ToString (   
    bool useBrackets )
```

[Simulation Message](#) ToString

6.300.2.19 WriteInt()

```
static void WriteInt (   
    SimulationMessage * message,   
    int value ) [static]
```

Write a int to a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to write to
<i>value</i>	int to write

6.300.2.20 WriteNetworkedObjectRef()

```
static void WriteNetworkedObjectRef (
    SimulationMessage * message,
    NetworkId value ) [static]
```

Write a [NetworkId](#) to a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to write to
<i>value</i>	NetworkId to write

6.300.2.21 WriteVector3()

```
static void WriteVector3 (
    SimulationMessage * message,
    Vector3 value ) [static]
```

Write a [Vector3](#) to a [SimulationMessage](#)

Parameters

<i>message</i>	SimulationMessage to write to
<i>value</i>	Vector3 to write

6.300.3 Member Data Documentation

6.300.3.1 Capacity

```
int Capacity
```

Capacity in Bits of this [SimulationMessage](#)

6.300.3.2 FLAG_DUMMY

```
const int FLAG_DUMMY = 1 << 8 [static]
```

Flag for dummy messages.

6.300.3.3 FLAG_INTERNAL

```
const int FLAG_INTERNAL = 1 << 6 [static]
```

Flag for internal messages.

6.300.3.4 FLAG_NOT_TICK_ALIGNED

```
const int FLAG_NOT_TICK_ALIGNED = 1 << 7 [static]
```

Flag for messages that are not tick aligned.

6.300.3.5 FLAG_REMOTE

```
const int FLAG_REMOTE = 1 << 1 [static]
```

Flag for remote messages.

6.300.3.6 FLAG_STATIC

```
const int FLAG_STATIC = 1 << 2 [static]
```

Flag for static messages.

6.300.3.7 FLAG_TARGET_PLAYER

```
const int FLAG_TARGET_PLAYER = 1 << 4 [static]
```

Flag for targeted messages to a player.

6.300.3.8 FLAG_TARGET_SERVER

```
const int FLAG_TARGET_SERVER = 1 << 5 [static]
```

Flag for targeted messages to the server.

6.300.3.9 FLAG_UNRELIABLE

```
const int FLAG_UNRELIABLE = 1 << 3 [static]
```

Flag for unreliable messages.

6.300.3.10 FLAG_USER_FLAGS_START

```
const int FLAG_USER_FLAGS_START = 1 << 16 [static]
```

Flag for user flags.

6.300.3.11 FLAG_USER_MESSAGE

```
const int FLAG_USER_MESSAGE = 1 << 0 [static]
```

Flag for user messages.

6.300.3.12 Flags

```
int Flags
```

Flags

6.300.3.13 FLAGS_RESERVED

```
const int FLAGS_RESERVED = 0xFFFF [static]
```

Flag for reserved flags.

6.300.3.14 FLAGS_RESERVED_BITS

```
const int FLAGS_RESERVED_BITS = 16 [static]
```

Flag for reserved bits.

6.300.3.15 MAX_PAYLOAD_SIZE

```
const int MAX_PAYLOAD_SIZE = 512 [static]
```

Max user message size in bytes.

6.300.3.16 Offset

```
int Offset
```

Current offset in Bits

6.300.3.17 References

```
int References
```

Reference Count

6.300.3.18 SIZE

```
const int SIZE = 28 [static]
```

SimulationMessage size in bytes.

6.300.3.19 Source

```
PlayerRef Source
```

Source Player of this SimulationMessage

6.300.3.20 Target

```
PlayerRef Target
```

Target Player of this SimulationMessage

6.300.3.21 Tick

int `Tick`

Tick of this [SimulationMessage](#)

6.300.4 Property Documentation

6.300.4.1 IsUnreliable

bool `IsUnreliable` [get]

Signal if this [SimulationMessage](#) is Unreliable

6.301 SimulationMessagePtr Struct Reference

[Simulation](#) Message Pointer

Public Attributes

- [SimulationMessage](#) * `Message`

Pointer to the message.

6.301.1 Detailed Description

[Simulation](#) Message Pointer

6.301.2 Member Data Documentation

6.301.2.1 Message

[SimulationMessage](#)* `Message`

Pointer to the message.

6.302 SimulationRuntimeConfig Struct Reference

Stores the runtime configuration of the simulation

Public Attributes

- `PlayerRef HostPlayer`
Current master client (in shared mode)
- `PlayerRef MasterClient`
Current master client (in shared mode)
- `int PlayerMaxCount`
Current player count
- `SimulationModes ServerMode`
Current `Simulation` Mode
- `TickRate.Resolved TickRate`
Current tick rates and send rates for server and client
- `Topologies Topology`
Current master client (in shared mode)

6.302.1 Detailed Description

Stores the runtime configuration of the simulation

6.302.2 Member Data Documentation

6.302.2.1 HostPlayer

`PlayerRef HostPlayer`

Current master client (in shared mode)

6.302.2.2 MasterClient

`PlayerRef MasterClient`

Current master client (in shared mode)

6.302.2.3 PlayerMaxCount

`int PlayerMaxCount`

Current player count

6.302.2.4 ServerMode

`SimulationModes ServerMode`

Current `Simulation` Mode

6.302.2.5 TickRate

`TickRate.Resolved TickRate`

Current tick rates and send rates for server and client

6.302.2.6 Topology

`Topologies Topology`

Current master client (in shared mode)

6.303 NetAddress Struct Reference

Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address

Inherits `IEquatable< NetAddress >`.

Public Member Functions

- bool `Equals (NetAddress other)`
- override bool `Equals (object obj)`
- override int `GetHashCode ()`
- override string `ToString ()`

Static Public Member Functions

- static `NetAddress Any (ushort port=0)`
Create a new `NetAddress` using the "Any" IPv4 Address representation (0.0.0.0) with the Port passed as argument
- static `NetAddress AnyIPv6 (ushort port=0)`
Create a new `NetAddress` using the "Any" IPv6 Address representation (::) with the Port passed as argument
- static `NetAddress CreateFromIpPort (string ip, ushort port)`
Create a new `NetAddress` based on the IP and Port passed as argument
- static `NetAddress FromActorId (int actorId)`
Build a new `NetAddress` based on an ActorId
- static `NetAddress LocalhostIPv4 (ushort port=0)`
Create a new `NetAddress` on the LocalHost address with the desired Port
- static `NetAddress LocalhostIPv6 (ushort port=0)`
Create a new `NetAddress` on the LocalHost IPv6 Address with the desired Port

Public Attributes

- int `_actorId`

Properties

- int `ActorId` [get]
Retrieves the Remote Actor ID which this `NetAddress` Represents
- bool `HasAddress` [get]
Signal if this `NetAddress` has a valid IP Address
- bool `IsIPv4` [get]
Signal if the `NetAddress` represents an IPv4 Address
- bool `IsIPv6` [get]
Signal if the `NetAddress` represents an IPv6 Address
- bool `IsRelayAddr` [get]
Signal if the `NetAddress` is a Relyed connection
- bool `IsValid` [get]
Signal if this `NetAddress` is not default/empty

6.303.1 Detailed Description

Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address

6.303.2 Member Function Documentation

6.303.2.1 Any()

```
static NetAddress Any (
    ushort port = 0 ) [static]
```

Create a new `NetAddress` using the "Any" IPv4 Address representation (0.0.0.0) with the Port passed as argument

Parameters

<code>port</code>	Port used to build the <code>NetAddress</code>
-------------------	--

Returns

New `NetAddress` reference

6.303.2.2 AnyIPv6()

```
static NetAddress AnyIPv6 (
    ushort port = 0 ) [static]
```

Create a new [NetAddress](#) using the "Any" IPv6 Address representation (::) with the Port passed as argument

Parameters

<i>port</i>	Port used to build the NetAddress
-------------	---

Returns

New [NetAddress](#) reference

6.303.2.3 CreateFromIpPort()

```
static NetAddress CreateFromIpPort (
    string ip,
    ushort port ) [static]
```

Create a new [NetAddress](#) based on the IP and Port passed as argument

Parameters

<i>ip</i>	String representation of an IP, either IPv4 or IPv6
<i>port</i>	Port used to build the NetAddress

Returns

New [NetAddress](#) reference

Exceptions

<i>ArgumentException</i>	If IP is empty/null or an invalid IP, or port < 0
<i>AssertException</i>	If unable to parse IP

6.303.2.4 FromActorId()

```
static NetAddress FromActorId (
    int actorId ) [static]
```

Build a new [NetAddress](#) based on an ActorId

Parameters

<i>actor</i> ↪ <i>Id</i>	ActorId used to build the NetAddress
-----------------------------	--

Returns

Relay [NetAddress](#) that references the ActorId

ActorId must be 0 or greater

6.303.2.5 LocalhostIPv4()

```
static NetAddress LocalhostIPv4 (
    ushort port = 0 ) [static]
```

Create a new [NetAddress](#) on the LocalHost address with the desired Port

Parameters

<i>port</i>	Port used to build the NetAddress
-------------	---

Returns

New [NetAddress](#) reference

6.303.2.6 LocalhostIPv6()

```
static NetAddress LocalhostIPv6 (
    ushort port = 0 ) [static]
```

Create a new [NetAddress](#) on the LocalHost IPv6 Address with the desired Port

Parameters

<i>port</i>	Port used to build the NetAddress
-------------	---

Returns

New [NetAddress](#) reference

6.303.3 Property Documentation

6.303.3.1 ActorId

```
int ActorId [get]
```

Retrieves the Remote Actor ID which this [NetAddress](#) Represents

6.303.3.2 HasAddress

```
bool HasAddress [get]
```

Signal if this [NetAddress](#) has a valid IP Address

6.303.3.3 IsIPv4

```
bool IsIPv4 [get]
```

Signal if the [NetAddress](#) represents an IPv4 Address

6.303.3.4 IsIPv6

```
bool IsIPv6 [get]
```

Signal if the [NetAddress](#) represents an IPv6 Address

6.303.3.5 IsRelayAddr

```
bool IsRelayAddr [get]
```

Signal if the [NetAddress](#) is a Relayed connection

6.303.3.6 IsValid

```
bool IsValid [get]
```

Signal if this [NetAddress](#) is not default/empty

6.304 NetBitBufferList Struct Reference

Represents a linked list of Fusion.Sockets.NetBitBuffer

Public Member Functions

- void [AddFirst](#) (NetBitBuffer *item)
Add a Fusion.Sockets.NetBitBuffer at the beginning of the List
- void [AddLast](#) (NetBitBuffer *item)
Add a Fusion.Sockets.NetBitBuffer at the end of the list.
- bool [IsInList](#) (NetBitBuffer *item)
Check if a specific Fusion.Sockets.NetBitBuffer is in the list
- void [Remove](#) (NetBitBuffer *item)
Remove a specific Fusion.Sockets.NetBitBuffer from the list
- NetBitBuffer * [RemoveHead](#) ()
Removes the first element of the list

Public Attributes

- int **Count**
- NetBitBuffer * **Head**
- NetBitBuffer * **Tail**

6.304.1 Detailed Description

Represents a linked list of Fusion.Sockets.NetBitBuffer

6.304.2 Member Function Documentation

6.304.2.1 AddFirst()

```
void AddFirst (
    NetBitBuffer * item )
```

Add a Fusion.Sockets.NetBitBuffer at the beginning of the List

Parameters

<i>item</i>	NetBitBuffer to add to the list
-------------	---------------------------------

6.304.2.2 AddLast()

```
void AddLast (
    NetBitBuffer * item )
```

Add a Fusion.Sockets.NetBitBuffer at the end of the list.

Parameters

<i>item</i>	NetBitBuffer to add to the list
-------------	---------------------------------

6.304.2.3 IsInList()

```
bool IsInList (
    NetBitBuffer * item )
```

Check if a specific Fusion.Sockets.NetBitBuffer is in the list

Parameters

<i>item</i>	NetBitBuffer to check
-------------	-----------------------

Returns

True if the list contains the item, false otherwise

6.304.2.4 Remove()

```
void Remove (
    NetBitBuffer * item )
```

Remove a specific Fusion.Sockets.NetBitBuffer from the list

Parameters

<i>item</i>	NetBitBuffer to remove
-------------	------------------------

6.304.2.5 RemoveHead()

```
NetBitBuffer* RemoveHead ( )
```

Removes the first element of the list

Returns

NetBitBuffer reference

6.305 NetCommandAccepted Struct Reference

Accepted Command, sent by the server when a remote client connection is accepted

Static Public Member Functions

- static [NetCommandAccepted Create](#) (NetConnectionId localId, NetConnectionId remoteId, uint counter)

Public Attributes

- NetConnectionId **AcceptedLocalId**
- NetConnectionId **AcceptedRemoteId**
- uint **Counter**
- [NetCommandHeader Header](#)

6.305.1 Detailed Description

Accepted Command, sent by the server when a remote client connection is accepted

6.306 NetCommandConnect Struct Reference

Connect Command used to signal a remote server that a client is trying to connect to it

Static Public Member Functions

- static int **ClampTokenLength** (int tokenLength)
- static [NetCommandConnect Create](#) (NetConnectionId id, byte *token=null, int tokenLength=0, byte *uniqueId=null)
- static byte[] **GetTokenDataAsArray** ([NetCommandConnect command](#))
- static byte[] **GetUniqueIdAsArray** ([NetCommandConnect command](#))

Public Attributes

- NetConnectionId **ConnectionId**
- [NetCommandHeader Header](#)
- fixed byte **TokenData** [TOKEN_MAX_LENGTH_BYTES]
- int **TokenLength**
- fixed byte **UniqueId** [UNIQUE_ID_LENGTH_BYTES]

Static Public Attributes

- const int **SIZE_BITS** = SIZE_BYTES * 8
- const int **SIZE_BYTES** = 16 + TOKEN_MAX_LENGTH_BYTES + UNIQUE_ID_LENGTH_BYTES
- const int **TOKEN_MAX_LENGTH_BYTES** = 128
- const int **UNIQUE_ID_LENGTH_BYTES** = NetConnection.UNIQUE_ID_SIZE

6.306.1 Detailed Description

Connect Command used to signal a remote server that a client is trying to connect to it

6.307 NetCommandDisconnect Struct Reference

Disconnect Command, it can be used by either side of the connection

Static Public Member Functions

- static [NetCommandDisconnect](#) **Create** ([NetDisconnectReason](#) reason, byte *token, int tokenLength)
- static [NetCommandDisconnect](#) **Create** ([NetDisconnectReason](#) reason, byte[] token)

Public Attributes

- [NetCommandHeader](#) **Header**
- [NetDisconnectReason](#) **Reason**
- fixed byte **TokenData** [TOKEN_MAX_LENGTH_BYTES]
- int **TokenLength**

Static Public Attributes

- const int **TOKEN_MAX_LENGTH_BYTES** = 128

6.307.1 Detailed Description

Disconnect Command, it can be used by either side of the connection

6.308 NetCommandHeader Struct Reference

Network Command Header Describe its type and usual settings for all commands

Static Public Member Functions

- static [NetCommandHeader](#) **Create** ([NetCommands](#) command)
Create a new [NetCommandHeader](#) based on a [NetCommands](#) type
- static implicit **operator NetCommandHeader** ([NetCommands](#) commands)

Public Attributes

- [NetCommands Command](#)
- [NetPacketType PacketType](#)

Static Public Attributes

- const int **SIZE_BITS** = SIZE_BYTES * 8
- const int **SIZE_BYTES** = 2

6.308.1 Detailed Description

Network Command Header Describe its type and usual settings for all commands

6.308.2 Member Function Documentation

6.308.2.1 Create()

```
static NetCommandHeader Create (
    NetCommands command ) [static]
```

Create a new [NetCommandHeader](#) based on a [NetCommands](#) type

Parameters

<i>command</i>	Type of Command that should be created
----------------	--

Returns

New [NetCommandHeader](#) reference based on the Command Type

6.309 NetCommandRefused Struct Reference

Refuse Command, sent by the server when the connection was refused. This happens when the server has reached its max connection capacity.

Static Public Member Functions

- static [NetCommandRefused Create](#) ([NetConnectFailedReason](#) reason)

Public Attributes

- [NetCommandHeader Header](#)
- [NetConnectFailedReason Reason](#)

Static Public Attributes

- const int **SIZE_IN_BITS** = SIZE_IN_BYTES * 8
- const int **SIZE_IN_BYTES** = 3

6.309.1 Detailed Description

Refuse Command, sent by the server when the connection was refused. This happens when the server has reached its max connection capacity.

6.310 NetConfig Struct Reference

General configuration used to drive the behavior of the Socket library

Public Attributes

- **NetAddress Address**
Network Address used to bind the internal Socket
- int **ConnectAttempts**
Number of Connection Attempts tried by the peer before cancel the connection
- double **ConnectInterval**
Interval in Seconds between attempts to connect to a remote server
- double **ConnectionDefaultRtt**
Initial RTT
- int **ConnectionGroups**
Number of Connection Groups supported by the local instance
- double **ConnectionPingInterval**
Interval in Seconds between ping being sent to a remote end
- int **ConnectionSendBuffers**
Pre-allocated number of data buffers used to send data
- double **ConnectionShutdownTime**
Timeout in Seconds to allow a disconnected Connection to be released from the Group Mapping
- double **ConnectionTimeout**
Connection Timeout in seconds
- int **MaxConnections**
Max Number of Connections supported by the local instance
- NetConfigNotify **Notify**
Package acknowledgment system configuration
- double **OperationExpireTime**
Max Allowed time for the Send and Receive operations, in milliseconds
- int **PacketSize**
UDP Packet Size in Bytes
- NetConfigSimulation **Simulation**
Network simulation system configuration
- int **SocketRecvBuffer**
Size of the internal Socket receive buffer
- int **SocketSendBuffer**
Size of the internal Socket send buffer

Properties

- int `ConnectionsPerGroup` [get]
Max number of Connection per Group based on the `ConnectionGroups` and `MaxConnections`
- static `NetConfig Defaults` [get]
Builds a `NetConfig` with the default values
- int `PacketSizeInBits` [get]
UDP Packet Size in Bits based on `PacketSize`

6.310.1 Detailed Description

General configuration used to drive the behavior of the Socket library

6.310.2 Member Data Documentation

6.310.2.1 Address

`NetAddress Address`

Network Address used to bind the internal Socket

6.310.2.2 ConnectAttempts

`int ConnectAttempts`

Number of Connection Attempts tried by the peer before cancel the connection

6.310.2.3 ConnectInterval

`double ConnectInterval`

Interval in Seconds between attempts to connect to a remote server

6.310.2.4 ConnectionDefaultRtt

`double ConnectionDefaultRtt`

Initial RTT

6.310.2.5 ConnectionGroups

```
int ConnectionGroups
```

Number of Connection Groups supported by the local instance

6.310.2.6 ConnectionPingInterval

```
double ConnectionPingInterval
```

Interval in Seconds between ping being sent to a remote end

6.310.2.7 ConnectionSendBuffers

```
int ConnectionSendBuffers
```

Pre-allocated number of data buffers used to send data

6.310.2.8 ConnectionShutdownTime

```
double ConnectionShutdownTime
```

Timeout in Seconds to allow a disconnected Connection to be released from the Group Mapping

NetPeerGroup.UpdateShutdown

6.310.2.9 ConnectionTimeout

```
double ConnectionTimeout
```

Connection Timeout in seconds

6.310.2.10 MaxConnections

```
int MaxConnections
```

Max Number of Connections supported by the local instance

6.310.2.11 Notify

```
NetConfigNotify Notify
```

Package acknowledgment system configuration

6.310.2.12 OperationExpireTime

```
double OperationExpireTime
```

Max Allowed time for the Send and Receive operations, in milliseconds

6.310.2.13 PacketSize

```
int PacketSize
```

UDP Packet Size in Bytes

6.310.2.14 Simulation

```
NetConfigSimulation Simulation
```

Network simulation system configuration

6.310.2.15 SocketRecvBuffer

```
int SocketRecvBuffer
```

Size of the internal Socket receive buffer

6.310.2.16 SocketSendBuffer

```
int SocketSendBuffer
```

Size of the internal Socket send buffer

6.310.3 Property Documentation

6.310.3.1 ConnectionsPerGroup

```
int ConnectionsPerGroup [get]
```

Max number of Connection per Group based on the [ConnectionGroups](#) and [MaxConnections](#)

6.310.3.2 Defaults

```
NetConfig Defaults [static], [get]
```

Builds a [NetConfig](#) with the default values

6.310.3.3 PacketSizeInBits

```
int PacketSizeInBits [get]
```

UDP Packet Size in Bits based on [PacketSize](#)

6.311 StunServers.StunServer Class Reference

Stores Addresses of a STUN Server

Public Member Functions

- override string [ToString \(\)](#)

Public Attributes

- [NetAddress IPv4Addr](#)
- [NetAddress IPv6Addr](#)

Properties

- bool [HasIPv4Support](#) [get]
- bool [HasIPv6Support](#) [get]
- static IEqualityComparer< [StunServer](#) > [StunServerEqualityComparer](#) = new Pv4AddrEqualityComparer() [get]

6.311.1 Detailed Description

Stores Addresses of a STUN Server

6.312 StartGameArgs Struct Reference

Fusion Start Arguments, used to configure the simulation mode and other settings

Public Member Functions

- override string **ToString** ()
StartGameArgs ToString()

Public Attributes

- **NetAddress?** **Address**
Peer Binding Address
- **AuthenticationValues** **AuthValues**
Custom Authentication Data
- **NetworkProjectConfig** **Config**
Custom NetworkProjectConfig used to start the simulation
- byte[] **ConnectionToken**
Connection token sent by client to server. Not used in shared mode.
- Type[] **CustomCallbackInterfaces**
User defined callback interfaces we will provide O(1) constant time lookup for
- string **CustomLobbyName**
Session Custom Lobby to be published in
- FusionAppSettings **CustomPhotonAppSettings**
Custom Photon Application Settings
- **NetAddress?** **CustomPublicAddress**
Custom Public Reflexive Address
- string **CustomSTUNServer**
Specify a Custom STUN Server used to Resolve the peer Reflexive Addresses It can be used to specify multiple STUN Servers separated by semicolon ;)
- bool **DisableNATPunchthrough**
Flag to disable the NAT Punchthrough implementation and connect only via Relay
- bool? **EnableClientSessionCreation**
Enables the Session creation when starting a Client with an specific Session Name
- **GameMode** **GameMode**
Fusion.GameMode in which this peer will start
- Action< **NetworkRunner** > **HostMigrationResume**
Callback invoked when the new Host is migrating from the old Host state
- **HostMigrationToken** **HostMigrationToken**
Host Migration Token used when restarting the Fusion Simulation
- bool? **isOpen**
Session should be created Open or Closed to accept joins
- bool? **isVisible**

- Session should be Visible or not in the Session Lobby list
- MatchmakingMode? [MatchmakingMode](#)

Session Join Matchmaking Mode when joining a Session. For more information, check Fusion.Photon.Realtime. ↵
MatchmakingMode
- [INetworkObjectInitializer ObjectInitializer](#)

See INetworkRunnerUpdater
- [INetworkObjectProvider ObjectProvider](#)

Object pool to use
- Action< [NetworkRunner](#) > [OnGameStarted](#)

Callback that is invoked when the Fusion has fully started
- int? [PlayerCount](#)

Number of players allowed to connect to the session.
- [NetworkSceneInfo? Scene](#)

Scene that will be set as the starting Scene.
- [INetworkSceneManager SceneManager](#)

See INetworkSceneManager.
- string [SessionName](#)

Photon Cloud Session Name used either to Create or Join a Session.
- Func< string > [SessionNameGenerator](#)

Used to generate a new Session Name when creating a Session.
- Dictionary< string, SessionProperty > [SessionProperties](#)

Custom Session Properties. This dictionary can be used to either setup the initial Session Properties when creating a Session but also to set the matchmaking filters when joining a Random Session.
- CancellationToken [StartGameCancellationToken](#)

Optional CancellationToken used to cancel the NetworkRunner start up process and shutdown
- [INetworkRunnerUpdater Updater](#)

See INetworkRunnerUpdater
- bool? [UseCachedRegions](#)

Enables the usage of the previous cached regions pings. This speeds up the region ping process and the runner startup process.
- bool? [UseDefaultPhotonCloudPorts](#)

Signal if the internal Realtime Client should use the Default Photon ports to connect to the Photon Cloud. By default, Fusion uses ports: 27000, 27001 and 27002. Set this to True to use ports: 5058, 5055 and 5056.

6.312.1 Detailed Description

Fusion Start Arguments, used to configure the simulation mode and other settings

More about matchmaking: <https://doc.photonengine.com/en-us/fusion/current/manual/matchmaking>

6.312.2 Member Function Documentation

6.312.2.1 [ToString\(\)](#)

```
override string ToString ( )
```

[StartGameArgs ToString\(\)](#)

6.312.3 Member Data Documentation

6.312.3.1 Address

`NetAddress? Address`

Peer Binding Address

Default: `NetAddress.Any(ushort)`

6.312.3.2 AuthValues

`AuthenticationValues AuthValues`

Custom Authentication Data

Default: null (default authentication values)

6.312.3.3 Config

`NetworkProjectConfig Config`

Custom `NetworkProjectConfig` used to start the simulation

Default: Global `NetworkProjectConfig`

More about `NetworkProjectConfig`: <https://doc.photonengine.com/en-us/fusion/current/manual/network-project-config>

6.312.3.4 ConnectionToken

`byte[] ConnectionToken`

Connection token sent by client to server. Not used in shared mode.

Default: null (empty connection token)

6.312.3.5 CustomCallbackInterfaces

`Type[] CustomCallbackInterfaces`

User defined callback interfaces we will provide O(1) constant time lookup for

Default: null

6.312.3.6 CustomLobbyName

```
string CustomLobbyName
```

Session Custom Lobby to be published in

Default: null (default Lobby for each Session Type, LobbyClientServer or LobbyShared)

6.312.3.7 CustomPhotonAppSettings

```
FusionAppSettings CustomPhotonAppSettings
```

Custom Photon Application Settings

Default: null (Global PhotonAppSettings)

6.312.3.8 CustomPublicAddress

```
NetAddress? CustomPublicAddress
```

Custom Public Reflexive Address

Default: null

6.312.3.9 CustomSTUNServer

```
string CustomSTUNServer
```

Specify a Custom STUN Server used to Resolve the peer Reflexive Addresses It can be used to specify multiple STUN Servers separated by semicolon (;

Default: null (no custom STUN Server)

6.312.3.10 DisableNATPunchthrough

```
bool DisableNATPunchthrough
```

Flag to disable the NAT Punchthrough implementation and connect only via Relay

Default: false

6.312.3.11 EnableClientSessionCreation

```
bool? EnableClientSessionCreation
```

Enables the Session creation when starting a Client with an specific Session Name

Default: false (clients *can not* create new Sessions)

6.312.3.12 GameMode

`GameMode GameMode`

`Fusion.GameMode` in which this peer will start

6.312.3.13 HostMigrationResume

`Action<NetworkRunner> HostMigrationResume`

Callback invoked when the new Host is migrating from the old Host state

Default: null

6.312.3.14 HostMigrationToken

`HostMigrationToken HostMigrationToken`

Host Migration Token used when restarting the `Fusion Simulation`

Default: null

6.312.3.15 IsOpen

`bool? IsOpen`

Session should be created Open or Closed to accept joins

Default: true

6.312.3.16 IsVisible

`bool? IsVisible`

Session should be Visible or not in the Session Lobby list

Default: true

6.312.3.17 MatchmakingMode

`MatchmakingMode? MatchmakingMode`

Session Join Matchmaking Mode when joining a Session. For more information, check `Fusion.Photon.Realtime.` ↪ `MatchmakingMode`

Default: `MatchmakingMode.FillRoom`

6.312.3.18 ObjectInitializer

`INetworkObjectInitializer` ObjectInitializer

See [INetworkRunnerUpdater](#)

6.312.3.19 ObjectProvider

`INetworkObjectProvider` ObjectProvider

Object pool to use

Default: null

6.312.3.20 OnGameStarted

`Action<NetworkRunner>` OnGameStarted

Callback that is invoked when the [Fusion](#) has fully started

Default: null

6.312.3.21 PlayerCount

`int?` PlayerCount

Number of players allowed to connect to the session.

Default: DefaultPlayers from the Global [NetworkProjectConfig](#)

6.312.3.22 Scene

`NetworkSceneInfo?` Scene

Scene that will be set as the starting Scene.

Default: null (no scene set)

6.312.3.23 SceneManager

`INetworkSceneManager` SceneManager

See [INetworkSceneManager](#).

Default: null

More about Scene Loading: <https://doc.photonengine.com/en-us/fusion/current/manual/scene-load>

6.312.3.24 SessionName

```
string SessionName
```

Photon Cloud Session Name used either to Create or Join a Session.

Default: null (random session matching)

6.312.3.25 SessionNameGenerator

```
Func<string> SessionNameGenerator
```

Used to generate a new Session Name when creating a Session.

Default: null (a random session name is generated based on a GUID)

6.312.3.26 SessionProperties

```
Dictionary<string, SessionProperty> SessionProperties
```

Custom Session Properties. This dictionary can be used to either setup the initial Session Properties when creating a Session but also to set the matchmaking filters when joining a Random Session.

Default: null (empty custom properties)

6.312.3.27 StartGameCancellationToken

```
CancellationToken StartGameCancellationToken
```

Optional CancellationToken used to cancel the [NetworkRunner](#) start up process and shutdown

Defaults: null

6.312.3.28 Updater

```
INetworkRunnerUpdater Updater
```

See [INetworkRunnerUpdater](#)

6.312.3.29 UseCachedRegions

```
bool? UseCachedRegions
```

Enables the usage of the previous cached regions pings. This speeds up the region ping process and the runner startup process.

Defaults: true

6.312.3.30 UseDefaultPhotonCloudPorts

```
bool? UseDefaultPhotonCloudPorts
```

Signal if the internal Realtime Client should use the Default Photon ports to connect to the Photon Cloud. By default, [Fusion](#) uses ports: 27000, 27001 and 27002. Set this to True to use ports: 5058, 5055 and 5056.

Default: false (uses ports 27000, 27001 and 27002)

6.313 StartGameResult Class Reference

Represents the result of starting the [Fusion Simulation](#)

Public Member Functions

- override string [ToString](#) ()
String representation of the StartGameResult

Properties

- string [ErrorMessage](#) [get]
Custom Error Message filled with data about the Shutdown. Usually used to store custom data when the StartGame fails.
- bool [Ok](#) [get]
Signal if the Start was OK
- [ShutdownReason](#) [ShutdownReason](#) [get]
Start Game Shutdown Reason
- string [StackTrace](#) [get]
Optional Exception StackTrace

6.313.1 Detailed Description

Represents the result of starting the [Fusion Simulation](#)

6.313.2 Member Function Documentation

6.313.2.1 [ToString\(\)](#)

```
override string ToString ( )
```

String representation of the [StartGameResult](#)

6.313.3 Property Documentation

6.313.3.1 ErrorMessage

```
string ErrorMessage [get]
```

Custom Error Message filled with data about the Shutdown. Usually used to store custom data when the StartGame fails.

6.313.3.2 Ok

```
bool Ok [get]
```

Signal if the Start was OK

6.313.3.3 ShutdownReason

```
ShutdownReason ShutdownReason [get]
```

Start Game Shutdown Reason

6.313.3.4 StackTrace

```
string StackTrace [get]
```

Optional Exception StackTrace

6.314 BehaviourStatisticsManager Class Reference

[Behaviour](#) statistics manager will provide access to the behaviour statistics snapshots.

Properties

- [BehaviourStatisticsSnapshot CompletedSnapshot \[get\]](#)

The completed snapshot of statistics related to [Fusion Behaviour](#) type execution from the previous update.

6.314.1 Detailed Description

[Behaviour](#) statistics manager will provide access to the behaviour statistics snapshots.

6.314.2 Property Documentation

6.314.2.1 CompletedSnapshot

`BehaviourStatisticsSnapshot CompletedSnapshot [get]`

The completed snapshot of statistics related to [Fusion Behaviour](#) type execution from the previous update.

6.315 BehaviourStatisticsSnapshot Class Reference

Represents a snapshot of statistics related to [Fusion Behaviour](#) type execution.

Properties

- int `FixedUpdateNetworkExecutionCount [get]`
Gets the count of FixedUpdateNetwork executions.
- double `FixedUpdateNetworkExecutionTime [get]`
Gets the total execution time of FixedUpdateNetwork in milliseconds.
- int `RenderExecutionCount [get]`
Gets the count of Render executions.
- double `RenderExecutionTime [get]`
Gets the total execution time of Render in milliseconds.

6.315.1 Detailed Description

Represents a snapshot of statistics related to [Fusion Behaviour](#) type execution.

6.315.2 Property Documentation

6.315.2.1 FixedUpdateNetworkExecutionCount

`int FixedUpdateNetworkExecutionCount [get]`

Gets the count of FixedUpdateNetwork executions.

6.315.2.2 FixedUpdateNetworkExecutionTime

```
double FixedUpdateNetworkExecutionTime [get]
```

Gets the total execution time of FixedUpdateNetwork in milliseconds.

6.315.2.3 RenderExecutionCount

```
int RenderExecutionCount [get]
```

Gets the count of Render executions.

6.315.2.4 RenderExecutionTime

```
double RenderExecutionTime [get]
```

Gets the total execution time of Render in milliseconds.

6.316 FusionStatisticsManager Class Reference

Represents a fusion statistics manager.

Properties

- [FusionStatisticsSnapshot CompleteSnapshot \[get\]](#)
Provides access to a complete snapshot of the fusion statistics.
- [NetworkObjectStatisticsManager ObjectStatisticsManager \[get\]](#)
Manages the network object statistics.

6.316.1 Detailed Description

Represents a fusion statistics manager.

6.316.2 Property Documentation

6.316.2.1 CompleteSnapshot

`FusionStatisticsSnapshot CompleteSnapshot [get]`

Provides access to a complete snapshot of the fusion statistics.

This property behaves as a read-only property and provides the collected snapshot of the fusion statistics.

6.316.2.2 ObjectStatisticsManager

`NetworkObjectStatisticsManager ObjectStatisticsManager [get]`

Manages the network object statistics.

6.317 FusionStatisticsSnapshot Class Reference

Represents a snapshot of [Fusion](#) statistics.

Properties

- int `ForwardTicks` [get]
Gets or sets the number of forward ticks.
- int `GeneralAllocMemoryFreeInBytes` [get]
The number of free segments allocated for general purposes on the simulation.
- int `GeneralAllocMemoryUsedInBytes` [get]
The number of used segments allocated for general purposes on the simulation.
- float `InBandwidth` [get]
Gets or sets the in-bandwidth value.
- int `InObjectUpdates` [get]
The number of received objects updates per packet.
- int `InPackets` [get]
Gets or sets the number of incoming packets.
- float `InputInBandwidth` [get]
Gets the input in bandwidth.
- float `InputOutBandwidth` [get]
Gets the input out bandwidth.
- float `InputReceiveDelta` [get]
Gets the Input Receive Delta.
- float `InterpolationOffset` [get]
Gets the Interpolation Offset.
- float `InterpolationSpeed` [get]
Gets the Interpolation Speed.
- int `ObjectsAllocMemoryFreeInBytes` [get]
The number of free segments allocated for objects on the simulation.
- int `ObjectsAllocMemoryUsedInBytes` [get]
The number of used segments allocated for objects on the simulation.
- float `OutBandwidth` [get]
Gets or sets the outbandwith property.

- int [OutObjectUpdates](#) [get]
The number of sent objects updates per packet.
- int [OutPackets](#) [get]
Gets or sets the number of outgoing packets.
- int [Resimulations](#) [get]
Gets or sets the number of resimulations.
- float [RoundTripTime](#) [get]
Gets or sets the round trip time (RTT) in seconds.
- float [SimulationSpeed](#) [get]
Gets the [Simulation Speed](#).
- float [SimulationTimeOffset](#) [get]
Gets the [Simulation Time Offset](#).
- float [StateReceiveDelta](#) [get]
Gets the [State Receive Delta](#).
- int [TimeResets](#) [get]
Gets the [Time Resets count](#).
- int [WordsReadCount](#) [get]
The number of words that were read.
- int [WordsReadSize](#) [get]
The total size in bytes of all read words.
- int [WordsWrittenCount](#) [get]
The number of words that were written.
- int [WordsWrittenSize](#) [get]
The total size in bytes of all written words.

6.317.1 Detailed Description

Represents a snapshot of [Fusion](#) statistics.

6.317.2 Property Documentation

6.317.2.1 ForwardTicks

int [ForwardTicks](#) [get]

Gets or sets the number of forward ticks.

6.317.2.2 GeneralAllocMemoryFreeInBytes

int [GeneralAllocMemoryFreeInBytes](#) [get]

The number of free segments allocated for general purposes on the simulation.

6.317.2.3 GeneralAllocMemoryUsedInBytes

```
int GeneralAllocMemoryUsedInBytes [get]
```

The number of used segments allocated for general purposes on the simulation.

6.317.2.4 InBandwidth

```
float InBandwidth [get]
```

Gets or sets the in-bandwidth value.

6.317.2.5 InObjectUpdates

```
int InObjectUpdates [get]
```

The number of received objects updates per packet.

6.317.2.6 InPackets

```
int InPackets [get]
```

Gets or sets the number of incoming packets.

6.317.2.7 InputInBandwidth

```
float InputInBandwidth [get]
```

Gets the input in bandwidth.

6.317.2.8 InputOutBandwidth

```
float InputOutBandwidth [get]
```

Gets the input out bandwidth.

6.317.2.9 InputReceiveDelta

```
float InputReceiveDelta [get]
```

Gets the Input Receive Delta.

6.317.2.10 InterpolationOffset

```
float InterpolationOffset [get]
```

Gets the Interpolation Offset.

6.317.2.11 InterpolationSpeed

```
float InterpolationSpeed [get]
```

Gets the Interpolation Speed.

6.317.2.12 ObjectsAllocMemoryFreeInBytes

```
int ObjectsAllocMemoryFreeInBytes [get]
```

The number of free segments allocated for objects on the simulation.

6.317.2.13 ObjectsAllocMemoryUsedInBytes

```
int ObjectsAllocMemoryUsedInBytes [get]
```

The number of used segments allocated for objects on the simulation.

6.317.2.14 OutBandwidth

```
float OutBandwidth [get]
```

Gets or sets the outbandwith property.

6.317.2.15 OutObjectUpdates

```
int OutObjectUpdates [get]
```

The number of sent objects updates per packet.

6.317.2.16 OutPackets

```
int OutPackets [get]
```

Gets or sets the number of outgoing packets.

6.317.2.17 Resimulations

```
int Resimulations [get]
```

Gets or sets the number of resimulations.

6.317.2.18 RoundTripTime

```
float RoundTripTime [get]
```

Gets or sets the round trip time (RTT) in seconds.

6.317.2.19 SimulationSpeed

```
float SimulationSpeed [get]
```

Gets the [Simulation Speed](#).

6.317.2.20 SimulationTimeOffset

```
float SimulationTimeOffset [get]
```

Gets the [Simulation Time Offset](#).

6.317.2.21 StateReceiveDelta

```
float StateReceiveDelta [get]
```

Gets the State Receive Delta.

6.317.2.22 TimeResets

```
int TimeResets [get]
```

Gets the Time Resets count.

6.317.2.23 WordsReadCount

```
int WordsReadCount [get]
```

The number of words that were read.

6.317.2.24 WordsReadSize

```
int WordsReadSize [get]
```

The total size in bytes of all read words.

6.317.2.25 WordsWrittenCount

```
int WordsWrittenCount [get]
```

The number of words that were written.

6.317.2.26 WordsWrittenSize

```
int WordsWrittenSize [get]
```

The total size in bytes of all written words.

6.318 LagCompensationStatisticsSnapshot Class Reference

Represents a snapshot of lag compensation statistics.

Properties

- double `AddOnBufferTime` [get]
Represents the amount of time taken to register new hitboxes on the HitboxBuffer.
- double `AddOnBVHTime` [get]
Represents the amount of time taken to register new hitboxes on the BVH.
- double `AdvanceBufferTime` [get]
Represents the amount of time taken to advance the hitbox buffer and copy previous snapshot information.
- int `BVHMaxDeep` [get]
Represents the deeper level reached by the BVH.
- int `BVHNodesCount` [get]
Represents total nodes count on the BVH.
- int `HitboxesCount` [get]
Represents the total number of HitBoxCollider on the HitBoxSnapshot colliders buffer, including both used and free ones.
- double `RefitBVHTime` [get]
Represents the amount of time taken to refit the BVH bounds after the nodes update.
- double `TotalElapsedTime` [get]
Represents the total time taken to advance, add and update all hitboxes on the Lag Compensation system.
- double `UpdateBufferTime` [get]
Represents the amount of time taken to update the HitboxBuffer.
- double `UpdateBVHTime` [get]
Represents the amount of time taken to update the BVH.

6.318.1 Detailed Description

Represents a snapshot of lag compensation statistics.

6.318.2 Property Documentation

6.318.2.1 AddOnBufferTime

```
double AddOnBufferTime [get]
```

Represents the amount of time taken to register new hitboxes on the HitboxBuffer.

6.318.2.2 AddOnBVHTime

```
double AddOnBVHTime [get]
```

Represents the amount of time taken to register new hitboxes on the BVH.

6.318.2.3 AdvanceBufferTime

```
double AdvanceBufferTime [get]
```

Represents the amount of time taken to advance the hitbox buffer and copy previous snapshot information.

6.318.2.4 BVHMaxDeep

```
int BVHMaxDeep [get]
```

Represents the deeper level reached by the BVH.

6.318.2.5 BVHNodesCount

```
int BVHNodesCount [get]
```

Represents total nodes count on the BVH.

6.318.2.6 HitboxesCount

```
int HitboxesCount [get]
```

Represents the total number of HitBoxCollider on the HitBoxSnapshot colliders buffer, including both used and free ones.

6.318.2.7 RefitBVHTime

```
double RefitBVHTime [get]
```

Represents the amount of time taken to refit the BVH bounds after the nodes update.

6.318.2.8 TotalElapsedTime

```
double TotalElapsedTime [get]
```

Represents the total time taken to advance, add and update all hitboxes on the Lag Compensation system.

6.318.2.9 UpdateBufferTime

```
double UpdateBufferTime [get]
```

Represents the amount of time taken to update the HitboxBuffer.

6.318.2.10 UpdateBVHTime

```
double UpdateBVHTime [get]
```

Represents the amount of time taken to update the BVH.

6.319 MemoryStatisticsSnapshot Struct Reference

Represents a snapshot of specific memory statistics. For basic total allocated memory and total free memory check the [Fusion Statistics](#).

Public Types

- enum class [TargetAllocator](#)
Enum representing the target allocator for memory statistics.

Public Attributes

- int[] [BucketFreeBlocksCount](#)
Represents the number of free blocks in each bucket of the allocator.
- int[] [BucketFullBlocksCount](#)
Represents the number of full blocks in each bucket of the allocator.
- int[] [BucketUsedBlocksCount](#)
Represents the number of used blocks in each bucket of the allocator.
- int [TotalFreeBlocks](#)
Represents the total number of free blocks in the allocator.

Static Public Attributes

- const int [BUCKET_COUNT](#) = Allocator.BUCKET_COUNT
The number of buckets used in the allocator.

6.319.1 Detailed Description

Represents a snapshot of specific memory statistics. For basic total allocated memory and total free memory check the [Fusion Statistics](#).

6.319.2 Member Enumeration Documentation

6.319.2.1 TargetAllocator

```
enum TargetAllocator [strong]
```

Enum representing the target allocator for memory statistics.

6.319.3 Member Data Documentation

6.319.3.1 BUCKET_COUNT

```
const int BUCKET_COUNT = Allocator.BUCKET_COUNT [static]
```

The number of buckets used in the allocator.

6.319.3.2 BucketFreeBlocksCount

```
int [] BucketFreeBlocksCount
```

Represents the number of free blocks in each bucket of the allocator.

6.319.3.3 BucketFullBlocksCount

```
int [] BucketFullBlocksCount
```

Represents the number of full blocks in each bucket of the allocator.

6.319.3.4 BucketUsedBlocksCount

```
int [ ] BucketUsedBlocksCount
```

Represents the number of used blocks in each bucket of the allocator.

6.319.3.5 TotalFreeBlocks

```
int TotalFreeBlocks
```

Represents the total number of free blocks in the allocator.

6.320 NetworkObjectStatisticsManager Class Reference

Manages network object statistics for monitored network objects.

Public Member Functions

- void [ClearMonitoredNetworkObjects \(\)](#)
Clears the list of monitored network objects.
- bool [GetNetworkObjectStatistics \(NetworkId id, out NetworkObjectStatisticsSnapshot objectStatistics\)](#)
Retrieves the network object statistics for a given network ID.
- void [MonitorNetworkObjectStatistics \(NetworkId id, bool monitor\)](#)
Starts or stops monitoring the statistics of a network object.

6.320.1 Detailed Description

Manages network object statistics for monitored network objects.

6.320.2 Member Function Documentation

6.320.2.1 ClearMonitoredNetworkObjects()

```
void ClearMonitoredNetworkObjects ( )
```

Clears the list of monitored network objects.

6.320.2.2 GetNetworkObjectStatistics()

```
bool GetNetworkObjectStatistics (
    NetworkId id,
    out NetworkObjectStatisticsSnapshot objectStatisticsSnapshot )
```

Retrieves the network object statistics for a given network ID.

Returns

Returns true if the object is being monitored and its statistics are successfully retrieved; otherwise, returns false.

6.320.2.3 MonitorNetworkObjectStatistics()

```
void MonitorNetworkObjectStatistics (
    NetworkId id,
    bool monitor )
```

Starts or stops monitoring the statistics of a network object.

Parameters

<i>id</i>	The identifier of the network object to monitor.
<i>monitor</i>	True to start monitoring, false to stop monitoring.

6.321 NetworkObjectStatisticsSnapshot Class Reference

Represents a snapshot of network object statistics.

Properties

- float **InBandwidth** [get]
Gets or sets the in-bandwidth value.
- int **InPackets** [get]
Gets or sets the number of incoming packets.
- float **OutBandwidth** [get]
Gets or sets the outbandwidth property.
- int **OutPackets** [get]
Gets or sets the number of outgoing packets.

6.321.1 Detailed Description

Represents a snapshot of network object statistics.

6.321.2 Property Documentation

6.321.2.1 InBandwidth

```
float InBandwidth [get]
```

Gets or sets the in-bandwidth value.

6.321.2.2 InPackets

```
int InPackets [get]
```

Gets or sets the number of incoming packets.

6.321.2.3 OutBandwidth

```
float OutBandwidth [get]
```

Gets or sets the outbandwith property.

6.321.2.4 OutPackets

```
int OutPackets [get]
```

Gets or sets the number of outgoing packets.

6.322 Tick Struct Reference

A tick is a 32-bit integer that represents a frame number.

Inherits IComparable< Tick >, and IEquatable< Tick >.

Classes

- class [EqualityComparer](#)
Provides a mechanism for comparing two `Tick` objects for equality.
- class [RelationalComparer](#)
Provides a mechanism for comparing two `Tick` objects.

Public Member Functions

- int [CompareTo \(Tick other\)](#)
Compares the current `Tick` with another `Tick`.
- override bool [Equals \(object obj\)](#)
Determines whether the specified object is equal to the current `Tick`.
- bool [Equals \(Tick other\)](#)
Determines whether the specified `Tick` is equal to the current `Tick`.
- override int [GetHashCode \(\)](#)
Serves as the default hash function.
- [Tick Next \(int increment\)](#)
Returns the next `Tick`, incremented by a specified value.
- override string [ToString \(\)](#)
String representation of the `Tick`.

Static Public Member Functions

- static implicit operator bool ([Tick value](#))
Converts a `Tick` to a boolean.
- static implicit operator int ([Tick value](#))
Converts a `Tick` to an integer.
- static implicit operator [Tick \(int value\)](#)
Converts an integer to a `Tick`.
- static bool [operator!= \(Tick a, Tick b\)](#)
Determines whether the first specified `Tick` is not equal to the second specified `Tick`.
- static bool [operator< \(Tick a, Tick b\)](#)
Determines whether the first specified `Tick` is less than the second specified `Tick`.
- static bool [operator<= \(Tick a, Tick b\)](#)
Determines whether the first specified `Tick` is less than or equal to the second specified `Tick`.
- static bool [operator== \(Tick a, Tick b\)](#)
Determines whether the first specified `Tick` is equal to the second specified `Tick`.
- static bool [operator> \(Tick a, Tick b\)](#)
Determines whether the first specified `Tick` is greater than the second specified `Tick`.
- static bool [operator>= \(Tick a, Tick b\)](#)
Determines whether the first specified `Tick` is greater than or equal to the second specified `Tick`.

Public Attributes

- int [Raw](#)
The raw value of the `Tick`. This represents a frame number.

Static Public Attributes

- const int [ALIGNMENT = 4](#)
The alignment of the `Tick` structure in bytes.
- const int [SIZE = 4](#)
The size of the `Tick` structure in bytes.

6.322.1 Detailed Description

A tick is a 32-bit integer that represents a frame number.

6.322.2 Member Function Documentation

6.322.2.1 CompareTo()

```
int CompareTo (
    Tick other )
```

Compares the current [Tick](#) with another [Tick](#).

Parameters

<i>other</i>	A Tick to compare with this Tick .
--------------	--

Returns

A value that indicates the relative order of the objects being compared.

6.322.2.2 Equals() [1/2]

```
override bool Equals (
    object obj )
```

Determines whether the specified object is equal to the current [Tick](#).

Parameters

<i>obj</i>	The object to compare with the current Tick .
------------	---

Returns

true if the specified object is equal to the current [Tick](#); otherwise, false.

6.322.2.3 Equals() [2/2]

```
bool Equals (
    Tick other )
```

Determines whether the specified [Tick](#) is equal to the current [Tick](#).

Parameters

<i>other</i>	The Tick to compare with the current Tick .
--------------	---

Returns

true if the specified [Tick](#) is equal to the current [Tick](#); otherwise, false.

6.322.2.4 GetHashCode()

```
override int GetHashCode ( )
```

Serves as the default hash function.

Returns

A hash code for the current [Tick](#).

6.322.2.5 Next()

```
Tick Next (  
    int increment )
```

Returns the next [Tick](#), incremented by a specified value.

Parameters

<i>increment</i>	The value to increment the Tick by.
------------------	---

Returns

A new [Tick](#) that represents the current [Tick](#) incremented by the specified value.

6.322.2.6 operator bool()

```
static implicit operator bool (   
    Tick value ) [static]
```

Converts a [Tick](#) to a boolean.

Parameters

<code>value</code>	The Tick to convert.
--------------------	--------------------------------------

Returns

true if the [Tick](#)'s raw value is greater than 0; otherwise, false.

6.322.2.7 operator int()

```
static implicit operator int (
    Tick value ) [static]
```

Converts a [Tick](#) to an integer.

Parameters

<code>value</code>	The Tick to convert.
--------------------	--------------------------------------

Returns

An integer that represents the raw value of the specified [Tick](#).

6.322.2.8 operator Tick()

```
static implicit operator Tick (
    int value ) [static]
```

Converts an integer to a [Tick](#).

Parameters

<code>value</code>	The integer to convert.
--------------------	-------------------------

Returns

A [Tick](#) that represents the specified integer. If the integer is less than 0, the [Tick](#)'s raw value is set to 0.

6.322.2.9 operator"!=()

```
static bool operator!= (
    Tick a,
    Tick b ) [static]
```

Determines whether the first specified [Tick](#) is not equal to the second specified [Tick](#).

6.322.2.10 operator<()

```
static bool operator< (
    Tick a,
    Tick b ) [static]
```

Determines whether the first specified [Tick](#) is less than the second specified [Tick](#).

6.322.2.11 operator<=()

```
static bool operator<= (
    Tick a,
    Tick b ) [static]
```

Determines whether the first specified [Tick](#) is less than or equal to the second specified [Tick](#).

6.322.2.12 operator==()

```
static bool operator== (
    Tick a,
    Tick b ) [static]
```

Determines whether the first specified [Tick](#) is equal to the second specified [Tick](#).

6.322.2.13 operator>()

```
static bool operator> (
    Tick a,
    Tick b ) [static]
```

Determines whether the first specified [Tick](#) is greater than the second specified [Tick](#).

6.322.2.14 operator>=()

```
static bool operator>= (
    Tick a,
    Tick b ) [static]
```

Determines whether the first specified [Tick](#) is greater than or equal to the second specified [Tick](#).

6.322.2.15 ToString()

```
override string ToString ( )
```

String representation of the [Tick](#).

6.322.3 Member Data Documentation

6.322.3.1 ALIGNMENT

```
const int ALIGNMENT = 4 [static]
```

The alignment of the [Tick](#) structure in bytes.

6.322.3.2 Raw

```
int Raw
```

The raw value of the [Tick](#). This represents a frame number.

6.322.3.3 SIZE

```
const int SIZE = 4 [static]
```

The size of the [Tick](#) structure in bytes.

6.323 Tick.EqualityComparer Class Reference

Provides a mechanism for comparing two [Tick](#) objects for equality.

Inherits [IEqualityComparer< Tick >](#).

Public Member Functions

- bool [Equals \(Tick x, Tick y\)](#)
Determines whether the specified [Tick](#) objects are equal.
- int [GetHashCode \(Tick obj\)](#)
Serves as a hash function for a [Tick](#) object.

6.323.1 Detailed Description

Provides a mechanism for comparing two [Tick](#) objects for equality.

6.323.2 Member Function Documentation

6.323.2.1 Equals()

```
bool Equals (
    Tick x,
    Tick y )
```

Determines whether the specified [Tick](#) objects are equal.

Parameters

x	The first Tick object to compare.
y	The second Tick object to compare.

Returns

true if the specified [Tick](#) objects are equal; otherwise, false.

6.323.2.2 GetHashCode()

```
int GetHashCode (
    Tick obj )
```

Serves as a hash function for a [Tick](#) object.

Parameters

obj	The Tick object for which to get a hash code.
-----	---

Returns

A hash code for the specified [Tick](#) object.

6.324 Tick.RelationalComparer Class Reference

Provides a mechanism for comparing two [Tick](#) objects.

Inherits [IComparer< Tick >](#).

Public Member Functions

- int [Compare \(Tick x, Tick y\)](#)

Compares two [Tick](#) objects and returns a value indicating whether one is less than, equal to, or greater than the other.

6.324.1 Detailed Description

Provides a mechanism for comparing two [Tick](#) objects.

6.324.2 Member Function Documentation

6.324.2.1 Compare()

```
int Compare (
    Tick x,
    Tick y )
```

Compares two [Tick](#) objects and returns a value indicating whether one is less than, equal to, or greater than the other.

Parameters

x	The first Tick object to compare.
y	The second Tick object to compare.

Returns

A signed integer that indicates the relative values of x and y.

6.325 TickAccumulator Struct Reference

A tick accumulator.

Public Member Functions

- void [AddTicks \(int ticks\)](#)

Adds a specified number of ticks to the [TickAccumulator](#).
- void [AddTime \(double dt, double step, int? maxTicks=null\)](#)

Adds a specified amount of time to the [TickAccumulator](#), incrementing the tick count as necessary.
- float [Alpha \(double step\)](#)

Calculates the alpha value based on the given step.
- bool [ConsumeTick \(out bool last\)](#)

Consumes a tick from the [TickAccumulator](#).
- void [Start \(\)](#)

Starts the [TickAccumulator](#), allowing it to accumulate ticks.
- void [Stop \(\)](#)

Stops the [TickAccumulator](#) from accumulating ticks.

Static Public Member Functions

- static [TickAccumulator StartNew \(\)](#)

Starts a new [TickAccumulator](#).

Public Attributes

- bool [_running](#)
- double [_scale](#)
- int [_ticks](#)
- double [_time](#)

Properties

- int [Pending](#) [get]
Gets the number of pending ticks in the [TickAccumulator](#).
- double [Remainder](#) [get]
Gets the remaining time in the [TickAccumulator](#).
- bool [Running](#) [get]
Gets a value indicating whether the [TickAccumulator](#) is running.
- double [TimeScale](#) [get, set]
Gets or sets the time scale of the [TickAccumulator](#).

6.325.1 Detailed Description

A tick accumulator.

6.325.2 Member Function Documentation

6.325.2.1 AddTicks()

```
void AddTicks (
    int ticks )
```

Adds a specified number of ticks to the [TickAccumulator](#).

Parameters

<i>ticks</i>	The number of ticks to add.
--------------	-----------------------------

6.325.2.2 AddTime()

```
void AddTime (
    double dt,
    double step,
    int? maxTicks = null )
```

Adds a specified amount of time to the [TickAccumulator](#), incrementing the tick count as necessary.

Parameters

<i>dt</i>	The amount of time to add.
<i>step</i>	The time step value.
<i>maxTicks</i>	The maximum number of ticks to add. If this value is reached, the remaining time is set to 0.

6.325.2.3 Alpha()

```
float Alpha (
    double step )
```

Calculates the alpha value based on the given step.

Parameters

<i>step</i>	The step value used to calculate the alpha.
-------------	---

Returns

The calculated alpha value.

6.325.2.4 ConsumeTick()

```
bool ConsumeTick (
    out bool last )
```

Consumes a tick from the [TickAccumulator](#).

Parameters

<i>last</i>	When this method returns, contains a boolean indicating whether the consumed tick was the last one.
-------------	---

Returns

true if a tick was successfully consumed; otherwise, false.

6.325.2.5 Start()

```
void Start ( )
```

Starts the [TickAccumulator](#), allowing it to accumulate ticks.

6.325.2.6 StartNew()

```
static TickAccumulator StartNew ( ) [static]
```

Starts a new [TickAccumulator](#).

Returns

A new [TickAccumulator](#).

6.325.2.7 Stop()

```
void Stop ( )
```

Stops the [TickAccumulator](#) from accumulating ticks.

6.325.3 Property Documentation

6.325.3.1 Pending

```
int Pending [get]
```

Gets the number of pending ticks in the [TickAccumulator](#).

6.325.3.2 Remainder

```
double Remainder [get]
```

Gets the remaining time in the [TickAccumulator](#).

6.325.3.3 Running

```
bool Running [get]
```

Gets a value indicating whether the [TickAccumulator](#) is running.

6.325.3.4 TimeScale

```
double TimeScale [get], [set]
```

Gets or sets the time scale of the [TickAccumulator](#).

Exceptions

<i>InvalidOperationException</i>	Thrown when the value is less than or equal to 0.
----------------------------------	---

6.326 TickRate Struct Reference

A tick rate is a collection of tick rates.

Classes

- struct [Resolved](#)
Represents a resolved tick rate.
- struct [Selection](#)
Represents a selection of tick rates for client and server.

Public Types

- enum class [ValidateResult](#)
Enumerates the possible results of validating a tick rate selection.

Public Member Functions

- [Selection ClampSelection \(Selection selection\)](#)
Clamps the indices in the specified [Selection](#) to valid ranges.
- int [GetDivisor \(int index\)](#)
Gets the divisor for the tick rate at the specified index.
- int [GetTickRate \(int index\)](#)
Gets the tick rate at the specified index.
- [TickRate \(params int\[\] rates\)](#)
- int[] [ToArray \(\)](#)
Converts the tick rates to an array.
- bool [Validate \(\)](#)
Validates the tick rates in the [TickRate](#).
- [ValidateResult ValidateSelection \(Selection selected\)](#)
Validates the tick rates in the specified [Selection](#).

Static Public Member Functions

- static [TickRate Get \(int rate\)](#)
Retrieves the [TickRate](#) associated with the specified tick rate.
- static void [Init \(\)](#)
Initializes the [TickRate](#) class by setting up the valid tick rates and their lookup dictionary.
- static void [InitChecked \(\)](#)
- static bool [IsValid \(int rate\)](#)
Checks if the provided tick rate is valid.
- static bool [IsValid \(TickRate rate\)](#)
Checks if the provided [TickRate](#) is valid.
- static [Resolved Resolve \(Selection selection\)](#)
Resolves the specified [Selection](#) into a [Resolved](#) structure.

Public Attributes

- int **_count**
- fixed int **_rates** [4]

Static Public Attributes

- static Dictionary< int, [TickRate](#) > **_lookup**
- static [TickRate](#)[] **_valid**
- static ReadOnlyCollection< [TickRate](#) > **_validReadOnly**

Properties

- static IReadOnlyList< [TickRate](#) > **Available** [get]
Gets a read-only list of all available TickRates.
- int **Client** [get]
Gets the tick rate for the client.
- int **Count** [get]
Gets the count of tick rates in the [TickRate](#).
- int **this[int index]** [get]
Gets the tick rate at the specified index.

6.326.1 Detailed Description

A tick rate is a collection of tick rates.

6.326.2 Member Enumeration Documentation

6.326.2.1 ValidateResult

```
enum ValidateResult [strong]
```

Enumerates the possible results of validating a tick rate selection.

Enumerator

Ok	The tick rate selection is valid.
Error	An error occurred during validation.
NotFound	The tick rate selection was not found.
InvalidTickRate	The tick rate selection is invalid.
ServerIndexOutOfRange	The server index in the tick rate selection is out of range.
ClientSendIndexOutOfRange	The client send index in the tick rate selection is out of range.
ServerSendIndexOutOfRange	The server send index in the tick rate selection is out of range.
ServerSendRateLargerThanTickRate	The server send rate in the tick rate selection is larger than the tick rate.

6.326.3 Member Function Documentation

6.326.3.1 ClampSelection()

```
Selection ClampSelection (
    Selection selection )
```

Clamps the indices in the specified [Selection](#) to valid ranges.

Parameters

<i>selection</i>	The Selection to clamp.
------------------	---

Returns

A new [Selection](#) with clamped indices. If the [TickRate](#) is invalid, returns a default [Selection](#).

6.326.3.2 Get()

```
static TickRate Get (
    int rate ) [static]
```

Retrieves the [TickRate](#) associated with the specified tick rate.

Parameters

<i>rate</i>	The tick rate to retrieve the TickRate for.
-------------	---

Returns

The [TickRate](#) associated with the specified tick rate.

Exceptions

<i>InvalidOperationException</i>	Thrown when the tick rate is invalid.
----------------------------------	---------------------------------------

6.326.3.3 GetDivisor()

```
int GetDivisor (
    int index )
```

Gets the divisor for the tick rate at the specified index.

Parameters

<i>index</i>	The index of the tick rate to get the divisor for.
--------------	--

Returns

The divisor for the tick rate at the specified index.

Exceptions

<i>ArgumentOutOfRangeException</i>	Thrown when the index is out of range.
<i>InvalidOperationException</i>	Thrown when the client tick rate is not divisible by the tick rate at the specified index.

6.326.3.4 GetTickRate()

```
int GetTickRate (
    int index )
```

Gets the tick rate at the specified index.

Parameters

<i>index</i>	The index of the tick rate to get.
--------------	------------------------------------

Returns

The tick rate at the specified index.

Exceptions

<i>ArgumentOutOfRangeException</i>	Thrown when the index is out of range.
------------------------------------	--

6.326.3.5 Init()

```
static void Init ( ) [static]
```

Initializes the [TickRate](#) class by setting up the valid tick rates and their lookup dictionary.

6.326.3.6 IsValid() [1/2]

```
static bool IsValid (
    int rate ) [static]
```

Checks if the provided tick rate is valid.

Parameters

<i>rate</i>	The tick rate to validate.
-------------	----------------------------

Returns

true if the tick rate is valid; otherwise, false.

6.326.3.7 IsValid() [2/2]

```
static bool IsValid (
    TickRate rate ) [static]
```

Checks if the provided [TickRate](#) is valid.

Parameters

<i>rate</i>	The TickRate to validate.
-------------	---

Returns

true if the [TickRate](#) is valid; otherwise, false.

6.326.3.8 Resolve()

```
static Resolved Resolve (
    Selection selection ) [static]
```

Resolves the specified [Selection](#) into a [Resolved](#) structure.

Parameters

<i>selection</i>	The Selection to resolve.
------------------	---

Returns

A [Resolved](#) structure that represents the resolved tick rates.

6.326.3.9 ToArray()

```
int [ ] ToArray ( )
```

Converts the tick rates to an array.

Returns

An array containing the tick rates.

6.326.3.10 Validate()

```
bool Validate ( )
```

Validates the tick rates in the [TickRate](#).

Returns

true if the tick rates are valid; otherwise, false.

The tick rates are valid if:

- There is at least one tick rate.
- There are no more than four tick rates.
- The first tick rate is greater than 0.
- Each tick rate is a divisor of the first tick rate.

6.326.3.11 ValidateSelection()

```
ValidateResult ValidateSelection (  
    Selection selected )
```

Validates the tick rates in the specified [Selection](#).

Parameters

<code>selected</code>	The Selection to validate.
-----------------------	--

Returns

A ValidateResult that indicates the result of the validation.

The [Selection](#) is valid if:

- The [TickRate](#) is valid.
- The client tick rate in the [Selection](#) matches the client tick rate in the [TickRate](#).
- The server index in the [Selection](#) is within the range of available tick rates.
- The server send index in the [Selection](#) is within the range of available tick rates.
- The client send index in the [Selection](#) is within the range of available tick rates.
- The server send rate in the [Selection](#) is not larger than the server tick rate.

6.326.4 Property Documentation

6.326.4.1 Available

```
IReadOnlyList<TickRate> Available [static], [get]
```

Gets a read-only list of all available TickRates.

This property ensures that the [TickRate](#) class is initialized before returning the list.

6.326.4.2 Client

```
int Client [get]
```

Gets the tick rate for the client.

6.326.4.3 Count

```
int Count [get]
```

Gets the count of tick rates in the [TickRate](#).

6.326.4.4 this[int index]

```
int this[int index] [get]
```

Gets the tick rate at the specified index.

Parameters

<i>index</i>	The index of the tick rate to get.
--------------	------------------------------------

Returns

The tick rate at the specified index.

6.327 TickRate.Resolved Struct Reference

Represents a resolved tick rate.

Public Member Functions

- override string [ToString \(\)](#)

Public Attributes

- int [Client](#)
The tick rate for the client.
- int [ClientSend](#)
The send tick rate for the client.
- int [Server](#)
The tick rate for the server.
- int [ServerSend](#)
The send tick rate for the server.

Static Public Attributes

- const int [SIZE](#) = 16
The size of the [Resolved](#) structure in bytes.
- const int [WORDS](#) = [SIZE](#) / [Allocator.REPLICATE_WORD_SIZE](#)
The size of the [Resolved](#) structure in words.

Properties

- double [ClientSendDelta](#) [get]
Gets the delta time for the client send rate.
- double [ClientTickDelta](#) [get]
Gets the delta time for the client tick rate.
- int [ClientTickStride](#) [get]
Gets the stride of the client tick rate. This is always 1.
- double [ServerSendDelta](#) [get]
Gets the delta time for the server send rate.
- double [ServerTickDelta](#) [get]
Gets the delta time for the server tick rate.
- int [ServerTickStride](#) [get]
Gets the stride of the server tick rate relative to the client tick rate.

6.327.1 Detailed Description

Represents a resolved tick rate.

The tick rate is resolved by the client and server tick rates.

6.327.2 Member Data Documentation

6.327.2.1 Client

```
int Client
```

The tick rate for the client.

6.327.2.2 ClientSend

```
int ClientSend
```

The send tick rate for the client.

6.327.2.3 Server

```
int Server
```

The tick rate for the server.

6.327.2.4 ServerSend

```
int ServerSend
```

The send tick rate for the server.

6.327.2.5 SIZE

```
const int SIZE = 16 [static]
```

The size of the [Resolved](#) structure in bytes.

6.327.2.6 WORDS

```
const int WORDS = SIZE / Allocator.REPLICATE_WORD_SIZE [static]
```

The size of the [Resolved](#) structure in words.

6.327.3 Property Documentation

6.327.3.1 ClientSendDelta

```
double ClientSendDelta [get]
```

Gets the delta time for the client send rate.

6.327.3.2 ClientTickDelta

```
double ClientTickDelta [get]
```

Gets the delta time for the client tick rate.

6.327.3.3 ClientTickStride

```
int ClientTickStride [get]
```

Gets the stride of the client tick rate. This is always 1.

6.327.3.4 ServerSendDelta

```
double ServerSendDelta [get]
```

Gets the delta time for the server send rate.

6.327.3.5 ServerTickDelta

```
double ServerTickDelta [get]
```

Gets the delta time for the server tick rate.

6.327.3.6 ServerTickStride

```
int ServerTickStride [get]
```

Gets the stride of the server tick rate relative to the client tick rate.

6.328 TickRate.Selection Struct Reference

Represents a selection of tick rates for client and server.

Public Attributes

- int [Client](#)
The tick rate for the client.
- int [ClientSendIndex](#)
The index of the client's send tick rate in the available tick rates.
- int [ServerIndex](#)
The index of the server's tick rate in the available tick rates.
- int [ServerSendIndex](#)
The index of the server's send tick rate in the available tick rates.

6.328.1 Detailed Description

Represents a selection of tick rates for client and server.

6.328.2 Member Data Documentation

6.328.2.1 Client

```
int Client
```

The tick rate for the client.

6.328.2.2 ClientSendIndex

```
int ClientSendIndex
```

The index of the client's send tick rate in the available tick rates.

6.328.2.3 ServerIndex

```
int ServerIndex
```

The index of the server's tick rate in the available tick rates.

6.328.2.4 ServerSendIndex

```
int ServerSendIndex
```

The index of the server's send tick rate in the available tick rates.

6.329 TickTimer Struct Reference

A timer that is based on ticks instead of seconds.

Inherits [INetworkStruct](#).

Public Member Functions

- bool [Expired](#) ([NetworkRunner](#) runner)
Checks if the [TickTimer](#) has expired.
- bool [ExpiredOrNotRunning](#) ([NetworkRunner](#) runner)
Checks if the [TickTimer](#) has expired or is not running.
- int? [RemainingTicks](#) ([NetworkRunner](#) runner)
Gets the number of remaining ticks until the [TickTimer](#) expires.
- float? [RemainingTime](#) ([NetworkRunner](#) runner)
Gets the remaining time in seconds until the [TickTimer](#) expires.
- override string [ToString](#) ()
Returns a string that represents the current [TickTimer](#).

Static Public Member Functions

- static [TickTimer CreateFromSeconds](#) ([NetworkRunner](#) runner, float delayInSeconds)
Creates a [TickTimer](#) from a specified delay in seconds.
- static [TickTimer CreateFromTicks](#) ([NetworkRunner](#) runner, int ticks)
Creates a [TickTimer](#) from a specified number of ticks.

Public Attributes

- int [_target](#)

Properties

- bool `IsRunning` [get]
Gets a value indicating whether the `TickTimer` is running.
- static `TickTimer None` [get]
Gets a `TickTimer` that is not running.
- int? `TargetTick` [get]
Gets the target tick of the `TickTimer`.

6.329.1 Detailed Description

A timer that is based on ticks instead of seconds.

6.329.2 Member Function Documentation

6.329.2.1 CreateFromSeconds()

```
static TickTimer CreateFromSeconds (
    NetworkRunner runner,
    float delayInSeconds ) [static]
```

Creates a `TickTimer` from a specified delay in seconds.

Parameters

<code>runner</code>	The <code>NetworkRunner</code> associated with the <code>TickTimer</code> .
<code>delayInSeconds</code>	The delay in seconds to set the <code>TickTimer</code> to.

Returns

A `TickTimer` that will expire after the specified delay in seconds. If the `NetworkRunner` is not alive or not running, returns a default `TickTimer`.

6.329.2.2 CreateFromTicks()

```
static TickTimer CreateFromTicks (
    NetworkRunner runner,
    int ticks ) [static]
```

Creates a `TickTimer` from a specified number of ticks.

Parameters

<i>runner</i>	The NetworkRunner associated with the TickTimer .
<i>ticks</i>	The number of ticks to set the TickTimer to.

Returns

A [TickTimer](#) that will expire after the specified number of ticks. If the [NetworkRunner](#) is not alive or not running, returns a default [TickTimer](#).

6.329.2.3 Expired()

```
bool Expired (
    NetworkRunner runner )
```

Checks if the [TickTimer](#) has expired.

Parameters

<i>runner</i>	The NetworkRunner associated with the TickTimer .
---------------	---

Returns

true if the [TickTimer](#) is alive, the runner is running, and the target tick has been reached or passed; otherwise, false.

6.329.2.4 ExpiredOrNotRunning()

```
bool ExpiredOrNotRunning (
    NetworkRunner runner )
```

Checks if the [TickTimer](#) has expired or is not running.

Parameters

<i>runner</i>	The NetworkRunner associated with the TickTimer .
---------------	---

Returns

true if the [TickTimer](#) is not running, the runner is not running, or the [TickTimer](#) has expired; otherwise, false.

6.329.2.5 RemainingTicks()

```
int? RemainingTicks (
    NetworkRunner runner )
```

Gets the number of remaining ticks until the [TickTimer](#) expires.

Parameters

<i>runner</i>	The NetworkRunner associated with the TickTimer .
---------------	---

Returns

The number of remaining ticks if the [TickTimer](#) is alive and running; otherwise, null.

6.329.2.6 RemainingTime()

```
float? RemainingTime (
    NetworkRunner runner )
```

Gets the remaining time in seconds until the [TickTimer](#) expires.

Parameters

<i>runner</i>	The NetworkRunner associated with the TickTimer .
---------------	---

Returns

The remaining time in seconds if there are remaining ticks; otherwise, null.

6.329.2.7 ToString()

```
override string ToString ( )
```

Returns a string that represents the current [TickTimer](#).

Returns

A string that represents the current [TickTimer](#).

6.329.3 Property Documentation

6.329.3.1 IsRunning

```
bool IsRunning [get]
```

Gets a value indicating whether the [TickTimer](#) is running.

true if the [TickTimer](#) is running; otherwise, false.

6.329.3.2 None

```
TickTimer None [static], [get]
```

Gets a [TickTimer](#) that is not running.

6.329.3.3 TargetTick

```
int? TargetTick [get]
```

Gets the target tick of the [TickTimer](#).

The target tick if the [TickTimer](#) is running; otherwise, null.

6.330 TimeSyncConfiguration Class Reference

Time Synchronization Configuration

Public Attributes

- double [MaxLateInputs](#) = 5.0
- double [MaxLateSnapshots](#) = 5.0
- int [RedundantInputs](#) = 2
- int [RedundantSnapshots](#) = 2
- double [SampleWindowSeconds](#) = 1.0

6.330.1 Detailed Description

Time Synchronization Configuration

6.330.2 Member Data Documentation

6.330.2.1 MaxLateInputs

```
double MaxLateInputs = 5.0
```

The maximum percentage of inputs that should ever arrive late because of jitter.

Decreasing this increases the client's simulation offset.

6.330.2.2 MaxLateSnapshots

```
double MaxLateSnapshots = 5.0
```

The maximum percentage of snapshots that should ever arrive late because of jitter.

Decreasing this increases the client's interpolation delay.

6.330.2.3 RedundantInputs

```
int RedundantInputs = 2
```

The number of consecutive, additional chances each input should have to reach the server before it's needed.

Increasing this increases the client's simulation offset.

6.330.2.4 RedundantSnapshots

```
int RedundantSnapshots = 2
```

The number of consecutive snapshots a client should be able to miss without disrupting their interpolation.

Increasing this increases the client's interpolation delay.

6.330.2.5 SampleWindowSeconds

```
double SampleWindowSeconds = 1.0
```

How big of a window the client looks at to evaluate the latest network conditions.

Increasing this makes the client slower to adapt to changes.

6.331 ToggleLeftAttribute Class Reference

Specifies that the bool field should be drawn as the toggle on the left side of the label.

Inherits [DrawerPropertyAttribute](#).

6.331.1 Detailed Description

Specifies that the bool field should be drawn as the toggle on the left side of the label.

6.332 UnitAttribute Class Reference

Unit Attribute class. Used to mark a field with the respective [Units](#)

Inherits [DecoratingPropertyAttribute](#).

Public Member Functions

- [UnitAttribute \(Units units\)](#)
Initializes a new instance of the [UnitAttribute](#) class with the specified unit.

Properties

- [Units Unit \[get\]](#)
Selected Unit for the field.

Additional Inherited Members

6.332.1 Detailed Description

Unit Attribute class. Used to mark a field with the respective [Units](#)

6.332.2 Constructor & Destructor Documentation

6.332.2.1 UnitAttribute()

```
UnitAttribute (
    Units units )
```

Initializes a new instance of the [UnitAttribute](#) class with the specified unit.

Parameters

<i>units</i>	<input type="text"/>
--------------	----------------------

6.332.3 Property Documentation

6.332.3.1 Unit

[Units](#) Unit [get]

Selected Unit for the field.

6.333 UnityAddressablesRuntimeKeyAttribute Class Reference

Specifies that the string field represents a key for Unity Addressables.

Inherits [PropertyAttribute](#).

6.333.1 Detailed Description

Specifies that the string field represents a key for Unity Addressables.

6.334 UnityAssetGuidAttribute Class Reference

Specifies that the string field represents a GUID of an asset.

Inherits [DrawerPropertyAttribute](#).

6.334.1 Detailed Description

Specifies that the string field represents a GUID of an asset.

6.335 UnityContextMenuItemAttribute Class Reference

Unity ContextMenuItemAttribute

Inherits [Attribute](#).

Public Member Functions

- [UnityContextMenuItemAttribute](#) (string function, string name)
ContextMenuItemAttribute Constructor

Properties

- int `order` [get, set]
ContextMenuItemAttribute Order

6.335.1 Detailed Description

Unity ContextMenuItemAttribute

6.335.2 Constructor & Destructor Documentation

6.335.2.1 UnityContextMenuItemAttribute()

```
UnityContextMenuAttribute (
    string function,
    string name )
```

ContextMenuAttribute Constructor

6.335.3 Property Documentation

6.335.3.1 order

```
int order [get], [set]  
ContextMenuAttribute Order
```

6.336 UnityDelayedAttribute Class Reference

Unity DelayedAttribute

Inherits Attribute.

Properties

- int `order` [get, set]
DelayedAttribute Order

6.336.1 Detailed Description

Unity DelayedAttribute

6.336.2 Property Documentation

6.336.2.1 order

```
int order [get], [set]
```

DelayedAttribute Order

6.337 UnityFormerlySerializedAsAttribute Class Reference

Unity FormerlySerializedAsAttribute

Inherits Attribute.

Public Member Functions

- [UnityFormerlySerializedAsAttribute](#) (string oldName)
FormerlySerializedAsAttribute Constructor

6.337.1 Detailed Description

Unity FormerlySerializedAsAttribute

6.337.2 Constructor & Destructor Documentation

6.337.2.1 UnityFormerlySerializedAsAttribute()

```
UnityFormerlySerializedAsAttribute (
    string oldName )
```

FormerlySerializedAsAttribute Constructor

6.338 UnityHeaderAttribute Class Reference

Unity HeaderAttribute

Inherits Attribute.

Public Member Functions

- [UnityHeaderAttribute](#) (string header)
HeaderAttribute Constructor

Properties

- int [order](#) [get, set]
HeaderAttribute Order

6.338.1 Detailed Description

Unity HeaderAttribute

6.338.2 Constructor & Destructor Documentation

6.338.2.1 UnityHeaderAttribute()

```
UnityHeaderAttribute (
    string header )
```

HeaderAttribute Constructor

6.338.3 Property Documentation

6.338.3.1 order

```
int order [get], [set]
```

HeaderAttribute Order

6.339 UnityMinAttribute Class Reference

Unity MinAttribute

Inherits Attribute.

Public Member Functions

- `UnityMinAttribute (float min)`

MinAttribute Constructor

Properties

- int `order` [get, set]

MinAttribute Order

6.339.1 Detailed Description

Unity MinAttribute

6.339.2 Constructor & Destructor Documentation

6.339.2.1 UnityMinAttribute()

```
UnityMinAttribute (
    float min )
```

MinAttribute Constructor

6.339.3 Property Documentation

6.339.3.1 order

```
int order [get], [set]
```

MinAttribute Order

6.340 UnityMultilineAttribute Class Reference

Unity MultilineAttribute

Inherits Attribute.

Properties

- int `order` [get, set]
MultilineAttribute Order

6.340.1 Detailed Description

Unity MultilineAttribute

6.340.2 Property Documentation

6.340.2.1 `order`

`int order [get], [set]`

MultilineAttribute Order

6.341 UnityNonReorderableAttribute Class Reference

Unity NonReorderableAttribute

Inherits Attribute.

Properties

- int `order` [get, set]
NonReorderableAttribute Order

6.341.1 Detailed Description

Unity NonReorderableAttribute

6.341.2 Property Documentation

6.341.2.1 order

int order [get], [set]

NonReorderableAttribute Order

6.342 UnityNonSerializedAttribute Class Reference

Unity NonSerializedAttribute

Inherits Attribute.

6.342.1 Detailed Description

Unity NonSerializedAttribute

6.343 UnityRangeAttribute Class Reference

Unity RangeAttribute

Inherits Attribute.

Public Member Functions

- [UnityRangeAttribute](#) (float min, float max)
RangeAttribute Constructor

Properties

- int [order](#) [get, set]
RangeAttribute Order

6.343.1 Detailed Description

Unity RangeAttribute

6.343.2 Constructor & Destructor Documentation

6.343.2.1 UnityRangeAttribute()

```
UnityRangeAttribute (
    float min,
    float max )
```

RangeAttribute Constructor

6.343.3 Property Documentation

6.343.3.1 order

```
int order [get], [set]
```

RangeAttribute Order

6.344 UnityResourcePathAttribute Class Reference

Specifies that the string field represents a path to a Unity resource.

Inherits [DrawerPropertyAttribute](#).

Public Member Functions

- [UnityResourcePathAttribute](#) (Type `resourceType`)
Initializes a new instance of the class with the specified resource type.

Properties

- Type [ResourceType](#) [get]
The type of the resource.

6.344.1 Detailed Description

Specifies that the string field represents a path to a Unity resource.

6.344.2 Constructor & Destructor Documentation

6.344.2.1 UnityResourcePathAttribute()

```
UnityResourcePathAttribute (
    Type resourceType )
```

Initializes a new instance of the class with the specified resource type.

Parameters

<i>resourceType</i>	<input type="text"/>
---------------------	----------------------

6.344.3 Property Documentation

6.344.3.1 ResourceType

Type ResourceType [get]

The type of the resource.

6.345 UnitySerializeField Class Reference

Unity SerializeField

Inherits Attribute.

6.345.1 Detailed Description

Unity SerializeField

6.346 UnitySerializeReference Class Reference

Unity SerializeReference

Inherits Attribute.

6.346.1 Detailed Description

Unity SerializeReference

6.347 UnitySpaceAttribute Class Reference

Unity SpaceAttribute

Inherits Attribute.

Public Member Functions

- [UnitySpaceAttribute](#) (float space)

SpaceAttribute Constructor

Properties

- int [order](#) [get, set]

SpaceAttribute Order

6.347.1 Detailed Description

Unity SpaceAttribute

6.347.2 Constructor & Destructor Documentation

6.347.2.1 UnitySpaceAttribute()

```
UnitySpaceAttribute (
    float space )
```

SpaceAttribute Constructor

6.347.3 Property Documentation

6.347.3.1 order

```
int order [get], [set]
```

SpaceAttribute Order

6.348 UnityTooltipAttribute Class Reference

Unity TooltipAttribute

Inherits Attribute.

Public Member Functions

- [UnityTooltipAttribute](#) (string tooltip)

TooltipAttribute Constructor

Properties

- int [order](#) [get, set]

TooltipAttribute Order

6.348.1 Detailed Description

Unity TooltipAttribute

6.348.2 Constructor & Destructor Documentation

6.348.2.1 UnityTooltipAttribute()

```
UnityTooltipAttribute (
    string tooltip )
```

TooltipAttribute Constructor

6.348.3 Property Documentation

6.348.3.1 order

```
int order [get], [set]
```

TooltipAttribute Order

6.349 Vector2Compressed Struct Reference

Represents a compressed Vector2 value for network transmission.

Inherits [INetworkStruct](#), and [IEquatable< Vector2Compressed >](#).

Public Member Functions

- `override bool Equals (object obj)`
Checks if the provided object is a `Vector2Compressed` instance and if it's equal to the current `Vector2Compressed` instance.
- `bool Equals (Vector2Compressed other)`
Checks if the current `Vector2Compressed` instance is equal to the other `Vector2Compressed` instance.
- `override int GetHashCode ()`
Returns the hash code for the current `Vector2Compressed` instance.

Static Public Member Functions

- `static implicit operator Vector2 (Vector2Compressed q)`
Implicit conversion from `Vector2Compressed` to `Vector2`.
- `static implicit operator Vector2Compressed (Vector2 v)`
Implicit conversion from `Vector2` to `Vector2Compressed`.
- `static bool operator!= (Vector2Compressed left, Vector2Compressed right)`
Inequality operator for `Vector2Compressed` struct.
- `static bool operator== (Vector2Compressed left, Vector2Compressed right)`
Equality operator for `Vector2Compressed` struct.

Public Attributes

- `int xEncoded`
Encoded value of the x component.
- `int yEncoded`
Encoded value of the y component.

Properties

- `float X [get, set]`
Gets or sets the x component.
- `float Y [get, set]`
Gets or sets the y component.

6.349.1 Detailed Description

Represents a compressed Vector2 value for network transmission.

6.349.2 Member Function Documentation

6.349.2.1 Equals() [1/2]

```
override bool Equals (
    object obj )
```

Checks if the provided object is a `Vector2Compressed` instance and if it's equal to the current `Vector2Compressed` instance.

Parameters

<i>obj</i>	The object to compare with the current Vector2Compressed instance.
------------	--

Returns

True if the provided object is a [Vector2Compressed](#) instance and it's equal to the current [Vector2Compressed](#) instance, otherwise false.

6.349.2.2 Equals() [2/2]

```
bool Equals (   
    Vector2Compressed other )
```

Checks if the current [Vector2Compressed](#) instance is equal to the other [Vector2Compressed](#) instance.

Parameters

<i>other</i>	The other Vector2Compressed instance to compare with the current Vector2Compressed instance.
--------------	--

Returns

True if the values of both [Vector2Compressed](#) instances are equal, otherwise false.

6.349.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [Vector2Compressed](#) instance.

Returns

A hash code for the current [Vector2Compressed](#) instance.

6.349.2.4 operator Vector2()

```
static implicit operator Vector2 (   
    Vector2Compressed q ) [static]
```

Implicit conversion from [Vector2Compressed](#) to [Vector2](#).

Parameters

<i>q</i>	The Vector2Compressed instance to convert.
----------	--

Returns

The decompressed Vector2 value of the [Vector2Compressed](#) instance.

6.349.2.5 operator [Vector2Compressed\(\)](#)

```
static implicit operator Vector2Compressed (
    Vector2 v ) [static]
```

Implicit conversion from Vector2 to [Vector2Compressed](#).

Parameters

<i>v</i>	The Vector2 value to convert.
----------	-------------------------------

Returns

A new [Vector2Compressed](#) instance with the compressed value of the Vector2.

6.349.2.6 operator"!=()

```
static bool operator!= (
    Vector2Compressed left,
    Vector2Compressed right ) [static]
```

Inequality operator for [Vector2Compressed](#) struct.

Parameters

<i>left</i>	First Vector2Compressed instance.
<i>right</i>	Second Vector2Compressed instance.

Returns

True if the value of the first [Vector2Compressed](#) instance is not equal to the value of the second [Vector2Compressed](#) instance, otherwise false.

6.349.2.7 operator==()

```
static bool operator== (
    Vector2Compressed left,
    Vector2Compressed right ) [static]
```

Equality operator for [Vector2Compressed](#) struct.

Parameters

<i>left</i>	First Vector2Compressed instance.
<i>right</i>	Second Vector2Compressed instance.

Returns

True if the value of the first [Vector2Compressed](#) instance is equal to the value of the second [Vector2Compressed](#) instance, otherwise false.

6.349.3 Member Data Documentation

6.349.3.1 xEncoded

```
int xEncoded
```

Encoded value of the x component.

6.349.3.2 yEncoded

```
int yEncoded
```

Encoded value of the y component.

6.349.4 Property Documentation

6.349.4.1 X

```
float X [get], [set]
```

Gets or sets the x component.

6.349.4.2 Y

```
float Y [get], [set]
```

Gets or sets the y component.

6.350 Vector3Compressed Struct Reference

Represents a compressed Vector3 value for network transmission.

Inherits [INetworkStruct](#), and [IEquatable< Vector3Compressed >](#).

Public Member Functions

- `override bool Equals (object obj)`
Checks if the provided object is a `Vector3Compressed` instance and if it's equal to the current `Vector3Compressed` instance.
- `bool Equals (Vector3Compressed other)`
Checks if the current `Vector3Compressed` instance is equal to the other `Vector3Compressed` instance.
- `override int GetHashCode ()`
Returns the hash code for the current `Vector3Compressed` instance.

Static Public Member Functions

- `static implicit operator Vector2 (Vector3Compressed q)`
Implicit conversion from `Vector3Compressed` to `Vector2`.
- `static implicit operator Vector3 (Vector3Compressed q)`
Implicit conversion from `Vector3Compressed` to `Vector3`.
- `static implicit operator Vector3Compressed (Vector2 v)`
Implicit conversion from `Vector2` to `Vector3Compressed`.
- `static implicit operator Vector3Compressed (Vector3 v)`
Implicit conversion from `Vector3` to `Vector3Compressed`.
- `static bool operator!= (Vector3Compressed left, Vector3Compressed right)`
Inequality operator for `Vector3Compressed` struct.
- `static bool operator== (Vector3Compressed left, Vector3Compressed right)`
Equality operator for `Vector3Compressed` struct.

Public Attributes

- `int xEncoded`
Encoded value of the x component.
- `int yEncoded`
Encoded value of the y component.
- `int zEncoded`
Encoded value of the z component.

Properties

- float `X` [get, set]
Gets or sets the x component.
- float `Y` [get, set]
Gets or sets the y component.
- float `Z` [get, set]
Gets or sets the z component.

6.350.1 Detailed Description

Represents a compressed Vector3 value for network transmission.

6.350.2 Member Function Documentation

6.350.2.1 Equals() [1/2]

```
override bool Equals (
    object obj )
```

Checks if the provided object is a [Vector3Compressed](#) instance and if it's equal to the current [Vector3Compressed](#) instance.

Parameters

<code>obj</code>	The object to compare with the current Vector3Compressed instance.
------------------	--

Returns

True if the provided object is a [Vector3Compressed](#) instance and it's equal to the current [Vector3Compressed](#) instance, otherwise false.

6.350.2.2 Equals() [2/2]

```
bool Equals (
    Vector3Compressed other )
```

Checks if the current [Vector3Compressed](#) instance is equal to the other [Vector3Compressed](#) instance.

Parameters

<code>other</code>	The other Vector3Compressed instance to compare with the current Vector3Compressed instance.
--------------------	--

Returns

True if the values of both [Vector3Compressed](#) instances are equal, otherwise false.

6.350.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [Vector3Compressed](#) instance.

Returns

A hash code for the current [Vector3Compressed](#) instance.

6.350.2.4 operator Vector2()

```
static implicit operator Vector2 (
    Vector3Compressed q ) [static]
```

Implicit conversion from [Vector3Compressed](#) to Vector2.

Parameters

<i>q</i>	The Vector3Compressed instance to convert.
----------	--

Returns

The decompressed Vector2 value of the [Vector3Compressed](#) instance.

6.350.2.5 operator Vector3()

```
static implicit operator Vector3 (
    Vector3Compressed q ) [static]
```

Implicit conversion from [Vector3Compressed](#) to Vector3.

Parameters

<i>q</i>	The Vector3Compressed instance to convert.
----------	--

Returns

The decompressed Vector3 value of the [Vector3Compressed](#) instance.

6.350.2.6 operator [Vector3Compressed\(\)](#) [1/2]

```
static implicit operator Vector3Compressed (
    Vector2 v ) [static]
```

Implicit conversion from Vector2 to [Vector3Compressed](#).

Parameters

v	The Vector2 value to convert.
---	-------------------------------

Returns

A new [Vector3Compressed](#) instance with the compressed value of the Vector2.

6.350.2.7 operator [Vector3Compressed\(\)](#) [2/2]

```
static implicit operator Vector3Compressed (
    Vector3 v ) [static]
```

Implicit conversion from Vector3 to [Vector3Compressed](#).

Parameters

v	The Vector3 value to convert.
---	-------------------------------

Returns

A new [Vector3Compressed](#) instance with the compressed value of the Vector3.

6.350.2.8 operator"!=()

```
static bool operator!= (
    Vector3Compressed left,
    Vector3Compressed right ) [static]
```

Inequality operator for [Vector3Compressed](#) struct.

Parameters

<i>left</i>	First <code>Vector3Compressed</code> instance.
<i>right</i>	Second <code>Vector3Compressed</code> instance.

Returns

True if the value of the first `Vector3Compressed` instance is not equal to the value of the second `Vector3Compressed` instance, otherwise false.

6.350.2.9 operator==()

```
static bool operator== (
    Vector3Compressed left,
    Vector3Compressed right ) [static]
```

Equality operator for `Vector3Compressed` struct.

Parameters

<i>left</i>	First <code>Vector3Compressed</code> instance.
<i>right</i>	Second <code>Vector3Compressed</code> instance.

Returns

True if the value of the first `Vector3Compressed` instance is equal to the value of the second `Vector3Compressed` instance, otherwise false.

6.350.3 Member Data Documentation**6.350.3.1 xEncoded**

```
int xEncoded
```

Encoded value of the x component.

6.350.3.2 yEncoded

```
int yEncoded
```

Encoded value of the y component.

6.350.3.3 zEncoded

int zEncoded

Encoded value of the z component.

6.350.4 Property Documentation

6.350.4.1 X

float X [get], [set]

Gets or sets the x component.

6.350.4.2 Y

float Y [get], [set]

Gets or sets the y component.

6.350.4.3 Z

float Z [get], [set]

Gets or sets the z component.

6.351 Vector4Compressed Struct Reference

Represents a compressed Vector4 value for network transmission.

Inherits [INetworkStruct](#), and [IEquatable<Vector4Compressed>](#).

Public Member Functions

- override bool [Equals](#) (object obj)
Checks if the provided object is a [Vector4Compressed](#) instance and if it's equal to the current [Vector4Compressed](#) instance.
- bool [Equals](#) ([Vector4Compressed](#) other)
Checks if the current [Vector4Compressed](#) instance is equal to the other [Vector4Compressed](#) instance.
- override int [GetHashCode](#) ()
Returns the hash code for the current [Vector4Compressed](#) instance.

Static Public Member Functions

- static implicit `operator Vector4 (Vector4Compressed q)`
Implicit conversion from `Vector4Compressed` to `Vector4`.
- static implicit `operator Vector4Compressed (Vector4 v)`
Implicit conversion from `Vector4` to `Vector4Compressed`.
- static bool `operator!= (Vector4Compressed left, Vector4Compressed right)`
Inequality operator for `Vector4Compressed` struct.
- static bool `operator== (Vector4Compressed left, Vector4Compressed right)`
Equality operator for `Vector4Compressed` struct.

Public Attributes

- int `wEncoded`
Encoded value of the w component.
- int `xEncoded`
Encoded value of the x component.
- int `yEncoded`
Encoded value of the y component.
- int `zEncoded`
Encoded value of the z component.

Properties

- float `W` [get, set]
Gets or sets the w component.
- float `X` [get, set]
Gets or sets the x component.
- float `Y` [get, set]
Gets or sets the y component.
- float `Z` [get, set]
Gets or sets the z component.

6.351.1 Detailed Description

Represents a compressed Vector4 value for network transmission.

6.351.2 Member Function Documentation

6.351.2.1 Equals() [1/2]

```
override bool Equals (
    object obj )
```

Checks if the provided object is a `Vector4Compressed` instance and if it's equal to the current `Vector4Compressed` instance.

Parameters

<i>obj</i>	The object to compare with the current Vector4Compressed instance.
------------	--

Returns

True if the provided object is a [Vector4Compressed](#) instance and it's equal to the current [Vector4Compressed](#) instance, otherwise false.

6.351.2.2 Equals() [2/2]

```
bool Equals (   
    Vector4Compressed other )
```

Checks if the current [Vector4Compressed](#) instance is equal to the other [Vector4Compressed](#) instance.

Parameters

<i>other</i>	The other Vector4Compressed instance to compare with the current Vector4Compressed instance.
--------------	--

Returns

True if the values of both [Vector4Compressed](#) instances are equal, otherwise false.

6.351.2.3 GetHashCode()

```
override int GetHashCode ( )
```

Returns the hash code for the current [Vector4Compressed](#) instance.

Returns

A hash code for the current [Vector4Compressed](#) instance.

6.351.2.4 operator Vector4()

```
static implicit operator Vector4 (   
    Vector4Compressed q ) [static]
```

Implicit conversion from [Vector4Compressed](#) to [Vector4](#).

Parameters

<i>q</i>	The Vector4Compressed instance to convert.
----------	--

Returns

The decompressed Vector4 value of the [Vector4Compressed](#) instance.

6.351.2.5 operator [Vector4Compressed\(\)](#)

```
static implicit operator Vector4Compressed (
    Vector4 v ) [static]
```

Implicit conversion from Vector4 to [Vector4Compressed](#).

Parameters

<i>v</i>	The Vector4 value to convert.
----------	-------------------------------

Returns

A new [Vector4Compressed](#) instance with the compressed value of the Vector4.

6.351.2.6 operator"!=()

```
static bool operator!= (
    Vector4Compressed left,
    Vector4Compressed right ) [static]
```

Inequality operator for [Vector4Compressed](#) struct.

Parameters

<i>left</i>	First Vector4Compressed instance.
<i>right</i>	Second Vector4Compressed instance.

Returns

True if the value of the first [Vector4Compressed](#) instance is not equal to the value of the second [Vector4Compressed](#) instance, otherwise false.

6.351.2.7 operator==()

```
static bool operator== (
    Vector4Compressed left,
    Vector4Compressed right ) [static]
```

Equality operator for `Vector4Compressed` struct.

Parameters

<i>left</i>	First <code>Vector4Compressed</code> instance.
<i>right</i>	Second <code>Vector4Compressed</code> instance.

Returns

True if the value of the first `Vector4Compressed` instance is equal to the value of the second `Vector4Compressed` instance, otherwise false.

6.351.3 Member Data Documentation

6.351.3.1 wEncoded

```
int wEncoded
```

Encoded value of the w component.

6.351.3.2 xEncoded

```
int xEncoded
```

Encoded value of the x component.

6.351.3.3 yEncoded

```
int yEncoded
```

Encoded value of the y component.

6.351.3.4 zEncoded

```
int zEncoded
```

Encoded value of the z component.

6.351.4 Property Documentation

6.351.4.1 W

```
float W [get], [set]
```

Gets or sets the w component.

6.351.4.2 X

```
float X [get], [set]
```

Gets or sets the x component.

6.351.4.3 Y

```
float Y [get], [set]
```

Gets or sets the y component.

6.351.4.4 Z

```
float Z [get], [set]
```

Gets or sets the z component.

6.352 WarnIfAttribute Class Reference

Editor attribute for adding notices to fields if the condition member evaluates as `true`. Condition member can be a property, field or method (with a return value).

Inherits [DolfAttributeBase](#).

Public Member Functions

- `WarnIfAttribute` (string conditionMember, bool compareToValue, string message, `CompareOperator` compare=`CompareOperator.Equal`)
- `WarnIfAttribute` (string conditionMember, double compareToValue, string message, `CompareOperator` compare=`CompareOperator.Equal`)
- `WarnIfAttribute` (string conditionMember, long compareToValue, string message, `CompareOperator` compare=`CompareOperator.Equal`)
- `WarnIfAttribute` (string conditionMember, string message)

Initializes a new instance that will hide the field if the condition member is not equal to zero.

Public Attributes

- bool `AsBox`
Should the warning be shown as a box?
- string `Message`
The default warning text, when a warning is shown.

Additional Inherited Members

6.352.1 Detailed Description

Editor attribute for adding notices to fields if the condition member evaluates as `true`. Condition member can be a property, field or method (with a return value).

6.352.2 Constructor & Destructor Documentation

6.352.2.1 `WarnIfAttribute()`

```
WarnIfAttribute (
    string conditionMember,
    string message )
```

Initializes a new instance that will hide the field if the condition member is not equal to zero.

Parameters

<code>conditionMember</code>	
<code>message</code>	

6.352.3 Member Data Documentation

6.352.3.1 AsBox

bool AsBox

Should the warning be shown as a box?

6.352.3.2 Message

string Message

The default warning text, when a warning is shown.

6.353 WeaverGeneratedAttribute Class Reference

Weaver Generated Attribute

Inherits Attribute.

6.353.1 Detailed Description

Weaver Generated Attribute

Index

_128, [67](#)
 Data, [67](#)
 SIZE, [67](#)
_16, [68](#)
 Data, [68](#)
 SIZE, [68](#)
_2, [69](#)
 Data, [69](#)
 SIZE, [69](#)
_256, [69](#)
 Data, [70](#)
 SIZE, [70](#)
_32, [70](#)
 Data, [71](#)
 SIZE, [71](#)
_4, [71](#)
 Data, [72](#)
 SIZE, [72](#)
_512, [72](#)
 Data, [73](#)
 SIZE, [73](#)
_64, [73](#)
 Data, [73](#)
 SIZE, [74](#)
_8, [74](#)
 Data, [74](#)
 SIZE, [74](#)
_debruijnTable
 DynamicHeap, [114](#)
_doubleValue
 DolfAttributeBase, [107](#)
_isDouble
 DolfAttributeBase, [108](#)
_longValue
 DolfAttributeBase, [108](#)
_reserved
 NetworkObjectHeader, [473](#)
_value0
 NetworkObjectTypeId, [494](#)
_value1
 NetworkObjectTypeId, [494](#)

AABB, [246](#)
 AABB, [246](#), [247](#)
 Center, [247](#)
 Extents, [247](#)
 Max, [247](#)
 Min, [247](#)
Accept

NetworkRunnerCallbackArgs.ConnectRequest, [613](#)
ACCURACY
 ReadWriteUtils, [713](#)
Acquire
 INetworkAssetSource< T >, [202](#)
AcquirePrefabInstance
 INetworkObjectProvider, [206](#)
ActivePlayers
 NetworkRunner, [604](#)
 Simulation, [776](#)
ActorId
 NetAddress, [819](#)
Add
 INetworkDictionary, [204](#)
 INetworkLinkedList, [205](#)
 NetworkDictionary< K, V >, [398](#)
 NetworkLinkedList< T >, [429](#), [430](#)
 SerializableDictionary< TKey, TValue >, [751](#)
 SimulationInput.Buffer, [798](#)
AddBehaviour< T >
 Behaviour, [97](#)
AddCallbacks
 NetworkRunner, [558](#)
AddFirst
 NetBitBufferList, [821](#)
AddGlobal
 NetworkRunner, [558](#)
AddInstance
 NetworkPrefabTable, [520](#)
AdditionalJitter
 NetworkSimulationConfiguration, [638](#)
AdditionalLoss
 NetworkSimulationConfiguration, [639](#)
AddLast
 NetBitBufferList, [821](#)
AddMode
 NetworkRunnerUpdaterDefaultInvokeSettings, [619](#)
AddOnBufferTime
 LagCompensationStatisticsSnapshot, [848](#)
AddOnBVHTime
 LagCompensationStatisticsSnapshot, [848](#)
AddOnCompleted
 NetworkSceneAsyncOp, [621](#)
AddPlayerAreaOfInterest
 NetworkRunner, [558](#)
Address
 NetConfig, [827](#)
 Ptr, [697](#)

StartGameArgs, 833
AddSceneRef
 NetworkSceneInfo, 628
AddSource
 NetworkPrefabTable, 520
AddTicks
 TickAccumulator, 863
AddTime
 TickAccumulator, 863
AdvanceBufferTime
 LagCompensationStatisticsSnapshot, 849
After
 Fusion, 57
AfterAllTicks
 IAfterAllTicks, 184
AfterClientPredictionReset
 IAfterClientPredictionReset, 185
AfterHostMigration
 IAfterHostMigration, 186
AfterRender
 IAfterRender, 186
AfterSimulation
 Simulation, 769
AfterSpawned
 IAfterSpawned, 187
AfterTick
 IAfterTick, 187
AfterUpdate
 IAfterUpdate, 188
 Simulation, 769
ALIGNMENT
 NetworkId, 419
 NetworkObjectGuid, 466
 NetworkObjectNestingKey, 481
 NetworkObjectTypeid, 494
 NetworkPrefabId, 506
 NetworkPrefabRef, 516
 Tick, 860
ALL
 AuthorityMasks, 95
All
 Fusion, 53
Allocate
 DynamicHeapInstance, 115
 SimulationMessage, 805
AllocateArray< T >
 DynamicHeapInstance, 115
AllocateArrayPointers< T >
 DynamicHeapInstance, 116
AllocateTracked< T >
 DynamicHeapInstance, 116
AllocateTrackedArray< T >
 DynamicHeapInstance, 117
AllocateTrackedArrayPointers< T >
 DynamicHeapInstance, 117
Allocator, 75
 BUCKET_COUNT, 76
 BUCKET_INVALID, 76
 HEAP_ALIGNMENT, 76
 REPLICATE_WORD_ALIGN, 76
 REPLICATE_WORD_SHIFT, 77
 REPLICATE_WORD_SIZE, 77
Allocator.Config, 77
 BlockByteSize, 81
 BlockCount, 80
 BlockShift, 80
 BlockWordCount, 81
 Config, 78
 DEFAULT_BLOCK_COUNT, 80
 DEFAULT_BLOCK_SHIFT, 80
 Equals, 78, 79
 GetHashCode, 79
 GlobalsSize, 80
 HeapSizeAllocated, 81
 HeapSizeUsable, 81
 SIZE, 80
 ToString, 79
AllowEditMode
 FusionGlobalScriptableObjectSourceAttribute, 156
AllowFallback
 FusionGlobalScriptableObjectSourceAttribute, 156
AllowMultipleTargets
 EditorButtonAttribute, 122
 FieldEditorButtonAttribute, 130
AllowStateAuthorityOverride
 Fusion, 47, 48
Alpha
 NetworkBehaviourBufferInterpolator, 355
 Query, 263
 QueryParams, 265
 TickAccumulator, 864
AlreadyRunning
 Fusion, 55
Always
 Fusion, 45
AlwaysExpanded
 ExpandableEnumAttribute, 128
Angle, 81
 Clamp, 83
 Equals, 83, 84
 GetHashCode, 84
 Lerp, 84
 Max, 84
 Min, 85
 NetworkBehaviourBufferInterpolator, 347
 operator Angle, 85, 86
 operator double, 86
 operator float, 86
 operator!=, 87
 operator<, 88
 operator<=, 88
 operator>, 89
 operator>=, 90
 operator+, 87
 operator-, 87
 operator==, 89

SIZE, 90
ToString, 90
Animator
 NetworkMecanimAnimator, 444
Any
 NetAddress, 817
AnyIPv6
 NetAddress, 817
ApplyTiming
 NetworkMecanimAnimator, 444
AreaOfInterest
 Fusion, 48
AreaOfInterestEnabled
 SimulationConfig, 794
AreaOfInterestOverride
 NetworkTRSPData, 679
ArrayLengthAttribute, 90
 ArrayLengthAttribute, 91
 MaxLength, 92
 MinLength, 92
AsBox
 ErrorIfAttribute, 127
 WarnIfAttribute, 910
AsCustom
 NetworkObjectTypeld, 494
AsIndex
 NetworkPrefabId, 507
 PlayerRef, 689
 SceneRef, 748
AsInternalStructId
 NetworkObjectTypeld, 495
AsPathHash
 SceneRef, 748
AspectRatio
 NormalizedRectAttribute, 682
AsPrefabId
 NetworkObjectTypeld, 495
AsSceneObjectId
 NetworkObjectTypeld, 495
AssembliesToWeave
 NetworkProjectConfig, 532
AssemblyNameAttribute, 92
 RequiresUnsafeCode, 92
AssemblyQualifiedName
 SerializableType< BaseType >, 759
AssetGuid
 INetworkPrefabSource, 208
AssetObject, 93
AsShort
 SerializableType< BaseType >, 758
Assign
 NetworkString< TSize >, 648
AssignInputAuthority
 NetworkObject, 447
Attach
 NetworkRunner, 558, 559
AuthenticationTicketExpired
 Fusion, 55
AuthenticationValues
 NetworkRunner, 604
AuthorityMasks, 95
 ALL, 95
 INPUT, 95
 NONE, 96
 PROXY, 96
 STATE, 96
AuthValues
 StartGameArgs, 833
AutoHostOrClient
 Fusion, 46
AutoUpdateAreaOfInterestOverride
 NetworkTransform, 675
Available
 TickRate, 872
Awaiter
 NetworkSceneAsyncOp.Awaiter, 625
 NetworkSpawnOp.Awaiter, 643
Awake
 NetworkObject, 448
BackColor
 ScriptHelpAttribute, 749
BaseType
 SerializableTypeAttribute, 760
Before
 Fusion, 57
BeforeAllTicks
 IBeforeAllTicks, 190
BeforeClientPredictionReset
 IBeforeClientPredictionReset, 191
BeforeCopyPreviousState
 IBeforeCopyPreviousState, 191
BeforeFirstTick
 Simulation, 769
BeforeHitboxRegistration
 IBeforeHitboxRegistration, 192
BeforeSimulation
 IBeforeSimulation, 192
 Simulation, 769
BeforeTick
 IBeforeTick, 193
BeforeUpdate
 IBeforeUpdate, 194
 Simulation, 769
Behaviour, 96
 AddBehaviour< T >, 97
 DestroyBehaviour, 97
 GetBehaviour< T >, 97
 NetworkBehaviourBufferInterpolator, 355
 NetworkBehaviourId, 359
 RpcHeader, 731
 TryGetBehaviour< T >, 97
BehaviourCount
 NetworkObjectHeader, 473
BehaviourMeta
 NetworkProjectConfigAsset, 538
BehaviourStatisticsManager, 839

CompletedSnapshot, 840
BehaviourStatisticsSnapshot, 840
 FixedUpdateNetworkExecutionCount, 840
 FixedUpdateNetworkExecutionTime, 840
 RenderExecutionCount, 841
 RenderExecutionTime, 841
BinaryDataAttribute, 98
 BitCount
 BitSetAttribute, 99
 Bits
 NetworkButtons, 391
 BitSetAttribute, 98
 BitCount, 99
 BitSetAttribute, 98
BitwiseAndNotEqualZero
 Fusion, 44
BLOCK_SIZE
 NetworkId, 419
BlockByteSize
 Allocator.Config, 81
BlockCount
 Allocator.Config, 80
BlockShift
 Allocator.Config, 80
BlockWordCount
 Allocator.Config, 81
Bool
 NetworkBehaviourBufferInterpolator, 347, 348
Bounds
 BVHNodeDrawInfo, 253
Box
 Fusion, 46
Box2D
 Fusion.LagCompensation, 61
BoxExtents
 ColliderDrawInfo, 254
 Hitbox, 161
BoxOverlapQuery, 248
 BoxOverlapQuery, 248, 249
 Center, 249
 Check, 249
 Extents, 250
 Rotation, 250
BoxOverlapQueryParams, 250
 BoxOverlapQueryParams, 251
 Center, 251
 Extents, 251
 queryParams, 251
 Rotation, 252
 StaticHitsCapacity, 252
BroadRadius
 HitboxRoot, 181
BUCKET_COUNT
 Allocator, 76
 MemoryStatisticsSnapshot, 851
BUCKET_INVALID
 Allocator, 76
BucketFreeBlocksCount
MemoryStatisticsSnapshot, 851
BucketFullBlocksCount
 MemoryStatisticsSnapshot, 851
BucketUsedBlocksCount
 MemoryStatisticsSnapshot, 851
Buffer
 SimulationInput.Buffer, 797
BuildType
 NetworkRunner, 604
BuildTypes
 NetworkProjectConfig, 532
 NetworkRunner, 557
BVHDepth
 HitboxManager, 176
BVHDraw, 252
 GetEnumerator, 252
 LagCompensationDraw, 258
BVHMaxDeep
 LagCompensationStatisticsSnapshot, 849
BVHNodeDrawInfo, 253
 Bounds, 253
 Depth, 253
 MaxDepth, 253
BVHNodes
 HitboxManager, 177
BVHNodesCount
 LagCompensationStatisticsSnapshot, 849
ByteCount
 MaxStringByteCountAttribute, 285
 NetworkObjectHeader, 475
Bytes
 Fusion, 57
CachedStaticCollidersSize
 LagCompensationSettings, 277
CanAllocateUserPayload
 SimulationMessage, 805
CanReceiveRenderCallback
 SimulationBehaviour, 788
CanReceiveSimulationCallback
 SimulationBehaviour, 788
CanSpawn
 NetworkRunner, 604
Capacity
 FixedBufferPropertyAttribute, 143
 NetworkDictionary< K, V >, 402
 NetworkDictionaryReadOnly< K, V >, 406
 NetworkLinkedList< T >, 433
 NetworkLinkedListReadOnly< T >, 438
 NetworkString< TSize >, 670
 SimulationMessage, 810
CapacityAttribute, 99
 CapacityAttribute, 99
 Length, 100
Capsule
 Fusion, 46
CapsuleBottomCenter
 ColliderDrawInfo, 254
CapsuleExtents

ColliderDrawInfo, 254
Hitbox, 161
CapsuleRadius
 Hitbox, 162
CapsuleTopCenter
 ColliderDrawInfo, 255
CELL_SIZE
 Simulation.AreaOfInterest, 785
Center
 AABB, 247
 BoxOverlapQuery, 249
 BoxOverlapQueryParams, 251
 SphereOverlapQuery, 274
 SphereOverlapQueryParams, 276
Changed
 NetworkBehaviour.ChangeDetector.Enumerable, 332
ChangedTick
 NetworkBehaviour, 325
Channel
 RpcAttribute, 727
 RpcInfo, 734
Check
 BoxOverlapQuery, 249
 Query, 263
 RaycastQuery, 269
 SphereOverlapQuery, 274
CheckNetworkedPropertiesBeingEmpty
 NetworkProjectConfig, 532
CheckRpcAttributeUsage
 NetworkProjectConfig, 532
Clamp
 Angle, 83
ClampMax
 RangeExAttribute, 705
ClampMin
 RangeExAttribute, 705
ClampSelection
 TickRate, 868
Clear
 FixedSize< T >, 135
 Mask256, 281
 NetworkArray< T >, 295
 NetworkDictionary< K, V >, 399
 NetworkLinkedList< T >, 430
 NetworkPrefabTable, 520
 SerializableDictionary< TKey, TValue >, 751
 SimulationInput, 795
 SimulationInput.Buffer, 798
ClearMonitoredNetworkObjects
 NetworkObjectStatisticsManager, 852
ClearPlayerAreaOfInterest
 NetworkRunner, 559
Client
 Fusion, 46, 56
 TickRate, 872
 TickRate.Resolved, 874
 TickRate.Selection, 876
ClientSend
 TickRate.Resolved, 874
ClientSendDelta
 TickRate.Resolved, 875
ClientSendIndex
 TickRate.Selection, 876
ClientSendIndexOutOfRange
 TickRate, 867
ClientServer
 Fusion, 55, 56
ClientTickDelta
 TickRate.Resolved, 875
ClientTickStride
 TickRate.Resolved, 875
ClientToClientWithServerProxy
 NetworkConfiguration, 392
ClientToServer
 NetworkConfiguration, 392
Clone
 NetworkSimulationConfiguration, 638
 SimulationMessage, 805
CloneArray< T >
 NetworkBehaviourUtils, 362
CollectGarbage
 DynamicHeap, 112
CollectGarbageDelegate
 DynamicHeap, 113
Collider
 LagCompensatedHit, 244
Collider2D
 LagCompensatedHit, 244
ColliderDrawInfo, 254
 BoxExtents, 254
 CapsuleBottomCenter, 254
 CapsuleExtents, 254
 CapsuleTopCenter, 255
 LocalToWorldMatrix, 255
 Offset, 255
 Radius, 255
 Type, 255
ColliderIndex
 Hitbox, 163
Compare
 DofAttributeBase, 108
 NetworkObjectSortKeyComparer, 486
 NetworkString< TSize >, 648, 649
 Tick.RelationalComparer, 862
Compare< TOtherSize >
 NetworkString< TSize >, 649, 650
CompareOperator
 Fusion, 44
Comparer
 NetworkId, 420
 NetworkObjectTypeid, 495
 PlayerRef, 689
CompareTo
 NetworkId, 416
 NetworkObjectGuid, 462

NetworkPrefabId, 504
 NetworkPrefabRef, 512
 Tick, 856
 Completed
 IAsyncOperation, 189
 CompletedSnapshot
 BehaviourStatisticsManager, 840
 CompleteSnapshot
 FusionStatisticsManager, 841
 Compress
 FloatUtils, 148
 ComputeHash
 IDataEncryption, 124
 ConditionMember
 DofAttributeBase, 108
 Config
 Allocator.Config, 78
 HitboxRoot, 181
 NetworkProjectConfigAsset, 538
 NetworkRunner, 605
 Simulation, 776
 StartGameArgs, 833
 ConfigFlags
 HitboxRoot, 179
 ConnectAttempts
 NetConfig, 827
 NetworkConfiguration, 393
 ConnectInterval
 NetConfig, 827
 NetworkConfiguration, 393
 ConnectionDefaultRtt
 NetConfig, 827
 NetworkConfiguration, 394
 ConnectionGroups
 NetConfig, 827
 ConnectionPingInterval
 NetConfig, 828
 NetworkConfiguration, 394
 ConnectionRefused
 Fusion, 55
 ConnectionSendBuffers
 NetConfig, 828
 ConnectionShutdownTime
 NetConfig, 828
 NetworkConfiguration, 393
 ConnectionsPerGroup
 NetConfig, 830
 ConnectionTimeout
 Fusion, 55
 NetConfig, 828
 NetworkConfiguration, 393
 ConnectionToken
 StartGameArgs, 833
 ConnectionType
 Fusion, 44
 ConsumeTick
 TickAccumulator, 864
 Contains
 NetworkLinkedList< T >, 430
 NetworkLinkedListReadOnly< T >, 436
 NetworkPrefabTable, 521
 NetworkString< TSize >, 650, 651
 SimulationInput.Buffer, 798
 Contains< TOtherSize >
 NetworkString< TSize >, 652
 ContainsKey
 NetworkDictionary< K, V >, 399
 SerializableDictionary< TKey, TValue >, 752
 ContainsValue
 NetworkDictionary< K, V >, 399
 CounterMask
 Fusion, 49
 ContinueWhenAll
 TaskManager, 93
 Convert< T >
 NetworkInput, 422
 CopyBackingFieldsToState
 NetworkBehaviour, 308
 CopyFrom
 FixedArray< T >, 135
 NetworkArray< T >, 296
 SimulationInput, 796
 CopyFromNetworkArray< T >
 NetworkBehaviourUtils, 362
 CopyFromNetworkDictionary< D, K, V >
 NetworkBehaviourUtils, 363
 CopyFromNetworkList< T >
 NetworkBehaviourUtils, 363
 CopySortedTo
 SimulationInput.Buffer, 799
 CopyStateFrom
 NetworkBehaviour, 308
 NetworkObject, 448
 CopyStateToBackingFields
 NetworkBehaviour, 308
 CopyTo
 FixedArray< T >, 136
 NetworkArray< T >, 297
 Count
 Fusion, 57
 NetworkDictionary< K, V >, 402
 NetworkDictionaryReadOnly< K, V >, 406
 NetworkLinkedList< T >, 433
 NetworkLinkedListReadOnly< T >, 438
 SerializableDictionary< TKey, TValue >, 754
 SimulationInput.Buffer, 800
 TickRate, 872
 Create
 NetCommandHeader, 825
 NetworkSimulationConfiguration, 638
 RpcHeader, 729, 730
 Create< T >
 FixedArray< T >, 136
 Create< TActual, TAdapted >
 FixedArray< T >, 137
 Create< TKey, TValue >

SerializableDictionary< TKey, TValue >, 752
CreateFromFieldSequence< T >
 FixedSizeArray< T >, 137
CreateFromIpPort
 NetAddress, 818
CreateFromSeconds
 TickTimer, 878
CreateFromTicks
 TickTimer, 878
Current
 FixedSizeArray< T >.Enumerator, 141
 NetworkArray< T >.Enumerator, 301
 NetworkBehaviour.ChangeDetector.Enumerator,
 334
 NetworkDictionary< K, V >.Enumerator, 403
 NetworkLinkedList< T >.Enumerator, 434
CurrentConnectionType
 NetworkRunner, 605
CurrentTypeId
 NetworkProjectConfig, 532
CurrentVersion
 NetworkProjectConfig, 532
Custom
 Fusion, 50, 55
CustomAuthenticationFailed
 Fusion, 55
CustomCallbackInterfaces
 StartGameArgs, 833
CustomLobbyName
 StartGameArgs, 833
CustomPhotonAppSettings
 StartGameArgs, 834
CustomPublicAddress
 StartGameArgs, 834
CustomSTUNServer
 StartGameArgs, 834

Data
 _128, 67
 _16, 68
 _2, 69
 _256, 70
 _32, 71
 _4, 72
 _512, 73
 _64, 73
 _8, 74
 NetworkInput, 424
 NetworkPrefabAcquireContext, 501
 NetworkPrefabInfo, 509
 NetworkTRSP, 677
 SimulationInput, 796
DataConsistency
 SimulationConfig, 791
DataEncryptor, 123
 EncryptData, 123
DataProperty
 IUnityValueSurrogate< T >, 223
 UnityArraySurrogate< T, ReaderWriter >, 226
 UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >,
 228
 UnityLinkedListSurrogate< T, ReaderWriter >, 230
 UnityValueSurrogate< T, TReaderWriter >, 234
Debug
 NetworkRunner, 557
Decompress
 FloatUtils, 148
DecoratingPropertyAttribute, 100
 DecoratingPropertyAttribute, 100, 101
 DefaultOrder, 101
DecryptData
 IDataEncryption, 125
Default
 HitboxRoot, 179
 NetworkedAttribute, 407
 NetworkPrefabTableOptions, 527
DEFAULT_ACCURACY
 FloatUtils, 149
DEFAULT_BLOCK_COUNT
 Allocator.Config, 80
DEFAULT_BLOCK_SHIFT
 Allocator.Config, 80
DefaultContents
 FusionGlobalScriptableObjectAttribute, 153
DefaultContentsGeneratorMethod
 FusionGlobalScriptableObjectAttribute, 153
DefaultForPropertyAttribute, 101
 DefaultForPropertyAttribute, 102
 PropertyName, 102
 WordCount, 102
 WordOffset, 102
DefaultOrder
 DecoratingPropertyAttribute, 101
DefaultPath
 FusionGlobalScriptableObjectAttribute, 153
DefaultResourceName
 NetworkProjectConfig, 533
Defaults
 NetConfig, 830
Degrees
 Fusion, 57
DegreesPerSecond
 Fusion, 57
DelayMax
 NetworkSimulationConfiguration, 639
DelayMin
 NetworkSimulationConfiguration, 639
DelayPeriod
 NetworkSimulationConfiguration, 639
DelayShape
 NetworkSimulationConfiguration, 639
DelayThreshold
 NetworkSimulationConfiguration, 639
Delegate
 RpcInvokeData, 736
DeltaTime

NetworkRunner, 605
 Simulation, 776
 SimulationConfig, 792
Depth
 BVHNodeDrawInfo, 253
Description
 INetworkAssetSource< T >, 203
Deserialize
 NetworkProjectConfig, 530
Despawn
 NetworkRunner, 559
Despawned
 IDespawned, 194
 NetworkBehaviour, 308
DestroyBehaviour
 Behaviour, 97
DestroySingleton< T >
 NetworkRunner, 560
DestroyWhenStateAuthorityLeaves
 Fusion, 47
DetectChanges
 NetworkBehaviour.ChangeDetector, 330, 331
Direct
 Fusion, 45
Direction
 RaycastQuery, 270
 RaycastQueryParams, 271
DirtyObject
 EditorButtonAttribute, 122
DisableNATPunchthrough
 StartGameArgs, 834
DisableSharedModeInterpolation
 NetworkTransform, 674
Disconnect
 NetworkRunner, 560
DisconnectedByPluginLogic
 Fusion, 55
DisconnectReason
 Fusion.Protocol, 62
DisplayAsEnumAttribute, 103
 DisplayAsEnumAttribute, 103, 104
 EnumType, 104
 EnumTypeMemberName, 104
DisplayNameAttribute, 104
 DisplayNameAttribute, 105
 Name, 105
Dispose
 FixedArray< T >.Enumerator, 141
 NetworkArray< T >.Enumerator, 301
 NetworkDictionary< K, V >.Enumerator, 403
 NetworkLinkedList< T >.Enumerator, 434
 SimulationBehaviourListScope, 790
Distance
 LagCompensatedHit, 244
DolfAttributeBase, 105
 _doubleValue, 107
 _isDouble, 108
 _longValue, 108
Compare, 108
ConditionMember, 108
DolfAttributeBase, 106, 107
ErrorOnConditionMemberNotFound, 108
DontDestroyOnLoad
 Fusion, 48, 49
 NetworkPrefabAcquireContext, 501
DrawerPropertyAttribute, 108
DrawGizmos
 Hitbox, 161
 HitboxRoot, 179
DrawIfAttribute, 109
 DrawIfAttribute, 109
 Hide, 110
 Mode, 110
DrawIfMode
 Fusion, 45
DrawInfo
 HitboxManager, 177
DrawInlineAttribute, 110
DynamicHeap, 110
 _debruijnTable, 114
 CollectGarbage, 112
 CollectGarbageDelegate, 113
 Free, 113
 SetForcedAlive< T >, 113
DynamicHeap.Ignore, 114
DynamicHeapInstance, 114
 Allocate, 115
 AllocateArray< T >, 115
 AllocateArrayPointers< T >, 116
 AllocateTracked< T >, 116
 AllocateTrackedArray< T >, 117
 AllocateTrackedArrayPointers< T >, 117
 DynamicHeapInstance, 115
 Free, 119
DynamicWordCount
 NetworkBehaviour, 326
 NetworkMecanimAnimator, 445
EditMode
 Fusion, 45
EditorButtonAttribute, 119
 AllowMultipleTargets, 122
 DirtyObject, 122
 EditorButtonAttribute, 120
 Label, 122
 Priority, 122
 Visibility, 122
EditorButtonVisibility
 Fusion, 45
ELEMENT_WORDS
 NetworkLinkedList< T >, 432
 NetworkLinkedListReadOnly< T >, 437
Empty
 NetworkObjectGuid, 467
 NetworkPrefabRef, 517
EnableAutoUpdate
 HostMigrationConfig, 183

EnableClientSessionCreation
StartGameArgs, 834

Enabled
LagCompensationSettings, 277
NetworkSimulationConfiguration, 640

Encoding
MaxStringByteCountAttribute, 285

EncryptData
DataEncryptor, 123
IDataEncryption, 125

EncryptionConfig
NetworkProjectConfig, 533

EndsWith
NetworkString< TSize >, 653

EndsWith< TOtherSize >
NetworkString< TSize >, 653

EnqueueIncompleteSynchronousSpawns
NetworkProjectConfig, 533

EnsureRootComponentExists< T, TStopOn >
NestedComponentUtilities, 287

EnsureRunnerScenesActive
NetworkRunner, 560

EntryKeyPropertyPath
SerializableDictionary< TKey, TValue >, 754

Enumerator
FixedSize< T >.Enumerator, 140
NetworkArray< T >.Enumerator, 300

EnumType
DisplayAsEnumAttribute, 104

EnumTypeMemberName
DisplayAsEnumAttribute, 104

Equal
Fusion, 44

Equals
Allocator.Config, 78, 79
Angle, 83, 84
FloatCompressed, 145
Mask256, 281, 282
NetworkBehaviourId, 357
NetworkBool, 379
NetworkButtons, 382, 383
NetworkId, 416, 417
NetworkId.EqualityComparer, 421
NetworkLoadSceneParameters, 439, 440
NetworkObjectGuid, 463
NetworkObjectGuid.EqualityComparer, 468
NetworkObjectHeader, 470
NetworkObjectNestingKey, 479, 480
NetworkObjectNestingKey.EqualityComparer, 482
NetworkObjectTypeId, 490, 491
NetworkObjectTypeId.EqualityComparer, 497
NetworkPrefabId, 504
NetworkPrefabId.EqualityComparer, 508
NetworkPrefabRef, 513
NetworkPrefabRef.EqualityComparer, 518
NetworkRunnerUpdaterDefaultInvokeSettings,
617, 618
NetworkSceneInfo, 628

NetworkSceneLoadId, 632, 633
NetworkSceneObjectId, 634, 635
NetworkString< TSize >, 654, 655
PlayerRef, 685
Ptr, 693, 694
Ptr.EqualityComparer, 698
QuaternionCompressed, 700
SceneRef, 743
SerializableType< BaseType >, 758
Tick, 856
Tick.EqualityComparer, 861
Vector2Compressed, 895, 896
Vector3Compressed, 900
Vector4Compressed, 905, 906

Equals< TOtherSize >
NetworkString< TSize >, 655, 656

Error
Fusion, 55
Fusion.Protocol, 62
IAsyncOperation, 189
NetworkSceneAsyncOp, 624
TickRate, 867
ErrorIfAttribute, 127
AsBox, 127
Message, 127

ErrorMessage
StartGameResult, 839

ErrorOnConditionMemberNotFound
DolfAttributeBase, 108

Eventual
SimulationConfig, 792

ExecutionOrder
NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta,
540

Exists
NetworkRunner, 561

ExpandableEnumAttribute, 128
AlwaysExpanded, 128
ShowFlagsButtons, 128
ShowInlineHelp, 128

ExpansionFactor
LagCompensationSettings, 278

Expired
TickTimer, 879

ExpiredOrNotRunning
TickTimer, 879

Extents
AABB, 247
BoxOverlapQuery, 250
BoxOverlapQueryParams, 251

Failed
Fusion, 47

FailedClientCantSpawn
Fusion, 49

FailedLocalPlayerNotYetSet
Fusion, 49

FailedToCreateInstance
Fusion, 49

FailedToLoadPrefabSynchronously
 Fusion, 49

FieldEditorButtonAttribute, 129
 AllowMultipleTargets, 130
 FieldEditorButtonAttribute, 129
 Label, 130
 TargetMethod, 130

FieldsMask
 FieldsMask< T >, 131, 132

FieldsMask< T >, 130
 FieldsMask, 131, 132
 Mask, 133
 operator Mask256, 132

FindObject
 NetworkRunner, 561

FindObjectsOfTypeInOrder< T >
 NestedComponentUtilities, 287

FindObjectsOfTypeInOrder< T, TCast >
 NestedComponentUtilities, 289

FirstChild
 Fusion, 57

FixedArray
 FixedArray< T >, 134

FixedArray< T >, 133
 Clear, 135
 CopyFrom, 135
 CopyTo, 136
 Create< T >, 136
 Create< TActual, TAdapted >, 137
 CreateFromFieldSequence< T >, 137
 FixedArray, 134
 GetEnumerator, 138
 IndexOf< T >, 138
 Length, 139
 this[int index], 139
 ToArray, 139
 ToString, 139
 ToString, 139

FixedArray< T >.Enumerator, 140
 Current, 141
 Dispose, 141
 Enumerator, 140
 MoveNext, 141
 Reset, 141

FixedBufferPropertyAttribute, 142
 Capacity, 143
 FixedBufferPropertyAttribute, 142
 SurrogateType, 143
 Type, 143

FixedStorage, 143
 GetWordCount< T >, 143

FixedUpdateNetwork
 NetworkBehaviour, 309
 SimulationBehaviour, 787

FixedUpdateNetworkExecutionCount
 BehaviourStatisticsSnapshot, 840

FixedUpdateNetworkExecutionTime
 BehaviourStatisticsSnapshot, 840

FLAG_ADDRESSABLE
 SceneRef, 747

FLAG_DUMMY
 SimulationMessage, 810

FLAG_INTERNAL
 SimulationMessage, 810

FLAG_NOT_TICK_ALIGNED
 SimulationMessage, 811

FLAG_REMOTE
 SimulationMessage, 811

FLAG_STATIC
 SimulationMessage, 811

FLAG_TARGET_PLAYER
 SimulationMessage, 811

FLAG_TARGET_SERVER
 SimulationMessage, 811

FLAG_UNRELIABLE
 SimulationMessage, 811

FLAG_USER_FLAGS_START
 SimulationMessage, 812

FLAG_USER_MESSAGE
 SimulationMessage, 812

Flags
 NetworkObject, 452
 NetworkObjectHeader, 473
 SimulationMessage, 812

FLAGS_RESERVED
 SimulationMessage, 812

FLAGS_RESERVED_BITS
 SimulationMessage, 812

Float
 NetworkBehaviourBufferInterpolator, 348

FloatCompressed, 144
 Equals, 145
 GetHashCode, 145
 operator float, 145
 operator FloatCompressed, 146
 operator!=, 146
 operator==, 147
 valueEncoded, 147

FloatUtils, 147
 Compress, 148
 Decompress, 148
 DEFAULT_ACCURACY, 149

Forward
 Fusion, 56

ForwardTicks
 FusionStatisticsSnapshot, 843

Frames
 Fusion, 57

FramesPerSecond
 Fusion, 57

Free
 DynamicHeap, 113
 DynamicHeapInstance, 119

From
 Fusion, 50
 NetworkBehaviourBufferInterpolator, 355

FromActorId
 NetAddress, 818
FromAsyncOperation
 NetworkSceneAsyncOp, 621
FromCompleted
 NetworkSceneAsyncOp, 622
FromCoroutine
 NetworkSceneAsyncOp, 622
FromCustom
 NetworkObjectType, 491
FromEncoded
 PlayerRef, 685
FromError
 NetworkSceneAsyncOp, 623
FromIndex
 NetworkPrefabId, 504
 PlayerRef, 686
 SceneRef, 744
FromLocal
 RpcInfo, 733
FromMessage
 RpcInfo, 733
FromPath
 SceneRef, 744
FromPrefabId
 NetworkObjectType, 491
FromRaw
 NetworkPrefabId, 505
 SceneRef, 745
FromSceneRefAndObjectIndex
 NetworkObjectType, 492
FromStruct
 NetworkObjectType, 492
FromTask
 NetworkSceneAsyncOp, 623
Full
 SimulationConfig, 792
 SimulationInput.Buffer, 800
FullCone
 Fusion.Sockets.Stun, 64
Fusion, 31
 After, 57
 All, 53
 AllowStateAuthorityOverride, 47, 48
 AlreadyRunning, 55
 Always, 45
 AreaOfInterest, 48
 AuthenticationTicketExpired, 55
 AutoHostOrClient, 46
 Before, 57
 BitwiseAndNotEqualZero, 44
 Box, 46
 Bytes, 57
 Capsule, 46
 Client, 46, 56
 ClientServer, 55, 56
 CompareOperator, 44
 ConnectionRefused, 55
 ConnectionTimeout, 55
 ConnectionType, 44
 ConterMask, 49
 Count, 57
 Custom, 50, 55
 CustomAuthenticationFailed, 55
 Degrees, 57
 DegreesPerSecond, 57
 DestroyWhenStateAuthorityLeaves, 47
 Direct, 45
 DisconnectedByPluginLogic, 55
 DontDestroyOnLoad, 48, 49
 DrawIfMode, 45
 EditMode, 45
 EditorButtonVisibility, 45
 Equal, 44
 Error, 55
 Failed, 47
 FailedClientCantSpawn, 49
 FailedLocalPlayerNotYetSet, 49
 FailedToCreateInstance, 49
 FailedToLoadPrefabSynchronously, 49
 FirstChild, 57
 Forward, 56
 Frames, 57
 FramesPerSecond, 57
 From, 50
 FusionGlobalScriptableObjectUnloadDelegate, 57
 GameClosed, 55
 GameIdAlreadyExists, 55
 GamesFull, 55
 GameMode, 45
 GameNotFound, 55
 GlobalObjectInterest, 47
 Greater, 44
 GreaterOrEqual, 44
 HasMainNetworkTRSP, 48
 Hide, 45
 High, 50
 HitboxTypes, 46
 HitOptions, 46
 Host, 46, 56
 HostMigration, 55
 Ignore, 47
 IgnoreInputAuthority, 46
 IncludeBox2D, 46
 IncludePhysX, 46
 IncompatibleConfiguration, 55
 Initial, 48
 InProgress, 48
 InputAuthority, 53
 InsufficientSourceAuthority, 52
 InsufficientTargetAuthority, 52
 InternalStruct, 50
 Interpolated, 50
 Invalid, 50, 55
 InvalidArguments, 55
 InvalidAuthentication, 55

InvalidRegion, 55
Invoked, 52
IsZero, 44
Kilobytes, 57
LastChild, 57
Latest, 50
Less, 44
LessOrEqual, 44
LoadError, 48
Local, 51
Low, 50
Lowest, 50
MaskBroadcast, 53
MaskCulled, 53
MaskNotSent, 53
MaskSent, 53
MaskVersion, 47
MasterClientObject, 47
MaxCcuReached, 55
Medium, 50
Megabytes, 57
MilliSecs, 57
Multiplier, 57
NetworkObjectAcquireResult, 46
NetworkObjectFlags, 47
NetworkObjectHeaderFlags, 47
NetworkObjectSpawnDelegate, 58
NetworkPrefabTableGetPrefabResult, 48
NetworkSceneInfoChangeSource, 48
NetworkSceneInfoDefaultFlags, 48
NetworkSpawnFlags, 49
NetworkSpawnStatus, 49
NetworkTypeKind, 49
NoActiveConnections, 52
None, 45–48, 52, 57
Normalized, 57
NormalizedPercentage, 57
NotCulled, 52
NotEqual, 44
NotFound, 48
NotInvokableDuringResim, 52
NotInvokableLocally, 52
NotSentBroadcastNoActiveConnections, 53
NotSentBroadcastNoConfirmedNorInterested-Clients, 53
NotSentTargetClientNotAvailable, 53
NotSentTargetObjectNotConfirmed, 53
NotSentTargetObjectNotInPlayerInterest, 53
NotZero, 44
Ok, 55
OperationCanceled, 55
OperationTimeout, 55
Packets, 57
PageSizes, 50
PayloadSizeExceeded, 52
Percentage, 57
PerSecond, 57
Photon, 54
PhotonCloudTimeout, 55
Player, 50
PlayMode, 45
Prefab, 50
PriorityLevel, 50
Proxies, 53
Queued, 49
Radians, 57
RadiansPerSecond, 57
ReadOnly, 45
Relayed, 45
Reliable, 51
Remote, 48, 51, 54
RenderSource, 50
RenderTimeframe, 51
Resimulate, 56
Retry, 47
RpcChannel, 51
RpcHostMode, 51
RpcInvokeDelegate, 58
RpcLocalInvokeResult, 51
RpcSendCullResult, 52
RpcSendMessageResult, 52
RpcSources, 53
RpcStaticInvokeDelegate, 58
RpcTargets, 53
RpcTargetStatus, 54
SceneCountMask, 49
SceneObject, 50
ScriptHeaderBackColor, 54
ScriptHeaderIcon, 54
ScriptHeaderStyle, 54
Seconds, 57
Self, 54
SentBroadcast, 53
SentToServerForForwarding, 53
SentToTargetClient, 53
Server, 46, 56
ServerInRoom, 55
SessionLobby, 54
Shared, 46, 55, 56
SharedModeStateAuthLocalPlayer, 49
SharedModeStateAuthMasterClient, 49
ShutdownReason, 55
SimulationModes, 56
SimulationStages, 56
Single, 46
SourcesHostPlayer, 51
SourcesServer, 51
Spawned, 49
SpawnedByClient, 47
Sphere, 46
SquareMagnitude, 57
StateAuthority, 53
Struct, 48
StructArray, 48
SubtickAccuracy, 46
Success, 47, 48

TagetPlayerIsNotLocal, 52
TargetPlayerIsLocalButRpcIsNotInvokableLocally, 52
TargetPlayerIsNotLocal, 52
TargetPlayerUnreachable, 52
Ticks, 57
TicksPerSecond, 57
To, 50
Topologies, 56
Units, 56, 57
Unity, 54
UnityPlayerLoopSystemAddMode, 57
Unreachable, 54
Unreliable, 51
V1, 47
Fusion.Analyzer, 59
Fusion.Async, 59
Fusion.Encryption, 59
Fusion.Internal, 59
Fusion.LagCompensation, 60
 Box2D, 61
 Hitbox, 61
 HitType, 61
 None, 61
 PhysX, 61
 PreProcessingDelegate, 61
Fusion.Protocol, 62
 DisconnectReason, 62
 Error, 62
 IncompatibleConfiguration, 62
 InvalidEventCode, 62
 InvalidJoinGameMode, 62
 InvalidJoinMsgType, 62
 ServerAlreadyInRoom, 62
 ServerLogic, 62
Fusion.Runtime, 62
Fusion.Runtime.Unity, 62
Fusion.Sockets, 62
 NetCommands, 63
 NetConnectFailedReason, 63
 NetDisconnectReason, 64
 NetPacketType, 64
 ServerFull, 64
 ServerRefused, 64
 Timeout, 64
Fusion.Sockets.Stun, 64
 FullCone, 64
 Invalid, 64
 NATType, 64
 OpenInternet, 64
 Symmetric, 64
 UdpBlocked, 64
Fusion.Statistics, 65
FusionGlobalScriptableObject< T >, 149
 GlobalInternal, 151
 IsGlobal, 151
 IsGlobalLoadedInternal, 151
 OnDisable, 150
 OnLoadedAsGlobal, 150
 OnUnloadedAsGlobal, 150
 TryGetGlobalInternal, 150
 UnloadGlobalInternal, 151
 FusionGlobalScriptableObjectAttribute, 152
 DefaultContents, 153
 DefaultContentsGeneratorMethod, 153
 DefaultPath, 153
 FusionGlobalScriptableObjectAttribute, 152
 FusionGlobalScriptableObjectLoadResult, 153
 FusionGlobalScriptableObjectLoadResult, 154
 Object, 154
 operator FusionGlobalScriptableObjectLoadResult, 154
 Unloader, 154
 FusionGlobalScriptableObjectSourceAttribute, 155
 AllowEditMode, 156
 AllowFallback, 156
 Load, 156
 ObjectType, 156
 Order, 156
 FusionGlobalScriptableObjectUnloadDelegate
 Fusion, 57
 FusionMonoBehaviour, 157
 FusionScriptableObject, 157
 FusionStatisticsManager, 841
 CompleteSnapshot, 841
 ObjectStatisticsManager, 842
 FusionStatisticsSnapshot, 842
 ForwardTicks, 843
 GeneralAllocMemoryFreeInBytes, 843
 GeneralAllocMemoryUsedInBytes, 843
 InBandwidth, 844
 InObjectUpdates, 844
 InPackets, 844
 InputInBandwidth, 844
 InputOutBandwidth, 844
 InputReceiveDelta, 844
 InterpolationOffset, 845
 InterpolationSpeed, 845
 ObjectsAllocMemoryFreeInBytes, 845
 ObjectsAllocMemoryUsedInBytes, 845
 OutBandwidth, 845
 OutObjectUpdates, 845
 OutPackets, 846
 Resimulations, 846
 RoundTripTime, 846
 SimulationSpeed, 846
 SimulationTimeOffset, 846
 StateReceiveDelta, 846
 TimeResets, 847
 WordsReadCount, 847
 WordsReadSize, 847
 WordsWrittenCount, 847
 WordsWrittenSize, 847
 GameClosed
 Fusion, 55
 GameIdAlreadyExists

Fusion, 55
GameIsFull
 Fusion, 55
GameMode
 Fusion, 45
 HostMigrationToken, 184
 NetworkRunner, 605
 StartGameArgs, 834
GameNotFound
 Fusion, 55
GameObject
 LagCompensatedHit, 245
GeneralAllocMemoryFreeInBytes
 FusionStatisticsSnapshot, 843
GeneralAllocMemoryUsedInBytes
 FusionStatisticsSnapshot, 843
GenerateKey
 IDataEncryption, 126
Get
 NetworkArray< T >, 298
 NetworkDictionary< K, V >, 399
 NetworkDictionaryReadOnly< K, V >, 406
 NetworkLinkedList< T >, 430
 NetworkLinkedListReadOnly< T >, 436
 NetworkString< TSize >, 656
 SimulationInput.Buffer, 799
 TickRate, 868
Get< T >
 NetworkInput, 422
GetAllBehaviours
 NetworkRunner, 561
GetAllBehaviours< T >
 NetworkRunner, 562
GetAllNetworkBehaviourTypes
 ReflectionUtils, 720
GetAllNetworkObjects
 NetworkRunner, 563
GetAllSimulationBehaviourTypes
 ReflectionUtils, 720
GetAllWeavedAssemblies
 ReflectionUtils, 720
GetAllWeavedNetworkBehaviourTypes
 ReflectionUtils, 720
GetAllWeavedSimulationBehaviourTypes
 ReflectionUtils, 721
GetAllWeaverGeneratedTypes
 ReflectionUtils, 721
GetAreaOfInterestGizmoData
 NetworkRunner, 563
 Simulation, 769
GetArrayReader< T >
 NetworkBehaviour, 309
GetAvailableRegions
 NetworkRunner, 563
GetAwaiter
 NetworkSceneAsyncOp, 623
 NetworkSpawnOp, 642
GetBehaviour< T >
 Behaviour, 97
 GetBehaviourChangedTickArray
 NetworkObjectHeader, 470
GetBehaviourReader< T >
 NetworkBehaviour, 310, 311
GetBehaviourReader< TBehaviour, TProperty >
 NetworkBehaviour, 311
GetBit
 Mask256, 282
GetByteArrayHashCode
 ReadWriteUtilsForWeaver, 714
GetByteCountUtf8NoHash
 ReadWriteUtilsForWeaver, 714
GetCapacity< TSize >
 NetworkString< TSize >, 657
GetCellSize
 Simulation.AreaOfInterest, 783
GetChangeDetector
 NetworkBehaviour, 312
GetCharCount
 NetworkString< TSize >, 657
GetCurrentVersion
 Versioning, 692
GetCustomAttributeOrThrow< T >
 ReflectionUtils, 721
GetData
 SimulationMessage, 806
GetDataPointer
 NetworkObjectHeader, 470
GetDataWordCount
 NetworkObjectHeader, 471
GetDictionaryReader< K, V >
 NetworkBehaviour, 312, 313
GetDivisor
 TickRate, 868
GetElementHashCode
 IElementReaderWriter< T >, 195
GetElementWordCount
 IElementReaderWriter< T >, 196
GetEntries
 NetworkPrefabTable, 521
GetEnumerator
 BVHDraw, 252
 FixedArray< T >, 138
 HitboxColliderContainerDraw, 256
 NetworkArray< T >, 298
 NetworkBehaviour.ChangeDetector.Enumerable,
 332
 NetworkDictionary< K, V >, 399
 NetworkLinkedList< T >, 431
 NetworkString< TSize >, 657
 SerializableDictionary< TKey, TValue >, 752
 SnapshotHistoryDraw, 272
GetExecutionOrder
 NetworkProjectConfig, 531
GetFlag
 SimulationMessage, 806
GetGuid

NetworkPrefabTable, 521
GetHashCode
 Allocator.Config, 79
 Angle, 84
 FloatCompressed, 145
 Mask256, 282
 NetworkBehaviourId, 357
 NetworkBool, 380
 NetworkButtons, 383
 NetworkId, 417
 NetworkId.EqualityComparer, 421
 NetworkLoadSceneParameters, 440
 NetworkObjectGuid, 463
 NetworkObjectGuid.EqualityComparer, 468
 NetworkObjectHeader, 471
 NetworkObjectNestingKey, 480
 NetworkObjectNestingKey.EqualityComparer, 482
 NetworkObjectTypeId, 493
 NetworkObjectTypeId.EqualityComparer, 498
 NetworkPrefabId, 505
 NetworkPrefabId.EqualityComparer, 508
 NetworkPrefabRef, 513
 NetworkPrefabRef.EqualityComparer, 518
 NetworkRunnerUpdaterDefaultInvokeSettings, 618
 NetworkSceneInfo, 629
 NetworkSceneLoadId, 633
 NetworkSceneObjectId, 635
 NetworkString< TSize >, 657
 PlayerRef, 686
 Ptr, 694
 Ptr.EqualityComparer, 699
 QuaternionCompressed, 701
 SceneRef, 745
 SerializableType< BaseType >, 758
 Tick, 857
 Tick.EqualityComparer, 861
 Vector2Compressed, 896
 Vector3Compressed, 901
 Vector4Compressed, 906
GetId
 NetworkPrefabTable, 521
GetInput< T >
 NetworkBehaviour, 313
GetInputAuthority
 Simulation, 770
GetInputForPlayer
 Simulation, 770
GetInputForPlayer< T >
 NetworkRunner, 564
GetInsertTime
 SimulationInput.Buffer, 799
GetInstanceCount
 NetworkPrefabTable, 522
GetInstanceEnumerator
 NetworkRunner, 564
GetInterfaceListHead
 NetworkRunner, 564
GetInterfaceListNext

 NetworkRunner, 565
GetInterfaceListPrev
 NetworkRunner, 565
GetInterfaceListsCount
 NetworkRunner, 565
GetLastUsedInputHeader
 SimulationInput.Buffer, 800
GetLinkListReader< T >
 NetworkBehaviour, 314
GetLocalAuthorityMask
 NetworkBehaviour, 315
 NetworkObject, 448
GetMainNetworkTRSPData
 NetworkObjectHeader, 471
GetMaxWordCount
 NetworkInputUtils, 425
GetMemorySnapshot
 NetworkRunner, 566
GetMetaData
 NetworkBehaviourUtils, 364
GetNestedComponentInChildren< T, TStopOn >
 NestedComponentUtilities, 290
GetNestedComponentInParent< T, TStopOn >
 NestedComponentUtilities, 291
GetNestedComponentInParents< T, TStopOn >
 NestedComponentUtilities, 291
GetNestedComponentsInChildren< T >
 NestedComponentUtilities, 291
GetNestedComponentsInChildren< T, TSearch, TStop >
 NestedComponentUtilities, 292
GetNestedComponentsInChildren< T, TStopOn >
 NestedComponentUtilities, 292
GetNestedComponentsInParents< T >
 NestedComponentUtilities, 292
GetNestedComponentsInParents< T, TStop >
 NestedComponentUtilities, 293
GetNetworkObjectStatistics
 NetworkObjectStatisticsManager, 852
GetObjectAndPlayersInAreaOfInterestCell
 Simulation, 770
GetObjectInAreaOfInterestForPlayer
 NetworkRunner, 566
 Simulation, 770
GetParentComponent< T >
 NestedComponentUtilities, 293
GetPhysicsScene
 NetworkRunner, 566
GetPhysicsScene2D
 NetworkRunner, 566
GetPlayerActorId
 NetworkRunner, 567
GetPlayerConnectionToken
 NetworkRunner, 567
GetPlayerConnectionType
 NetworkRunner, 567
GetPlayerObject
 NetworkRunner, 568

GetPlayerRtt
 NetworkRunner, 568

GetPlayerTickAndAlpha
 HitboxManager, 166

GetPlayerUserId
 NetworkRunner, 568

GetPressed
 NetworkButtons, 383

GetProductVersion
 Versioning, 692

GetPropertyReader< T >
 NetworkBehaviour, 315

GetPropertyReader< TBehaviour, TProperty >
 NetworkBehaviour, 316

GetRawInputForPlayer
 NetworkRunner, 569

GetRef< T >
 NetworkArrayExtensions, 302

GetReleased
 NetworkButtons, 384

GetRenderBuffers
 RenderTimeline, 724

GetResult
 NetworkSceneAsyncOp.Awaiter, 626
 NetworkSpawnOp.Awaiter, 644

GetResumeSnapshotNetworkObjects
 NetworkRunner, 569

GetResumeSnapshotNetworkSceneObjects
 NetworkRunner, 569

GetRpcStaticIndexOrThrow
 NetworkBehaviourUtils, 364

GetRpcTargetStatus
 NetworkRunner, 569

GetRunnerForGameObject
 NetworkRunner, 570

GetRunnerForScene
 NetworkRunner, 570

GetSceneRef
 INetworkSceneManager, 216, 217

GetShortAssemblyQualifiedName
 SerializableType< BaseType >, 758

GetSingleton< T >
 NetworkRunner, 570

GetSource
 NetworkPrefabTable, 522, 523

GetStateAuthority
 Simulation, 771

GetStaticWordCount
 NetworkBehaviourUtils, 365

GetStatisticsSnapshot
 HitboxManager, 166

GetStringHashCode
 ReadWriteUtilsForWeaver, 714

GetTickRate
 TickRate, 869

GetType
 NetworkInputUtils, 425

GetTypeKey

NetworkInputUtils, 425

GetVersion
 NetworkObjectFlagsExtensions, 457

GetWeavedAttributeOrThrow
 ReflectionUtils, 722

GetWordCount
 NetworkBehaviourUtils, 365
 NetworkInputUtils, 426
 NetworkObject, 448

GetWordCount< T >
 FixedStorage, 143
 NetworkStructUtils, 671

GetWordCountString
 ReadWriteUtilsForWeaver, 715

GizmosColor
 Hitbox, 162
 HitboxRoot, 181

GizmosDrawWireCapsule
 LagCompensationDraw, 258

Global
 NetworkProjectConfig, 536
 NetworkProjectConfigAsset, 539

GlobalInternal
 FusionGlobalScriptableObject< T >, 151

GlobalObjectInterest
 Fusion, 47

GlobalsSize
 Allocator.Config, 80
 HeapConfiguration, 158

Greater
 Fusion, 44

GreaterOrEqual
 Fusion, 44

GroupTypesByNamespace
 SerializeReferenceTypePickerAttribute, 762

Guid
 NetworkObjectPrefabData, 483

HasAddress
 NetAddress, 820

HasAnyActiveConnections
 Simulation, 771

HasHeader
 NetworkPrefabAcquireContext, 502
 NetworkPrefabInfo, 509

HasInputAuthority
 NetworkBehaviour, 326
 NetworkObject, 454

HasMainNetworkTRSP
 Fusion, 48
 NetworkObjectHeader, 472

HasSingleton< T >
 NetworkRunner, 571

HasStateAuthority
 NetworkBehaviour, 326
 NetworkObject, 454

HasStaticWordCount
 NetworkBehaviourUtils, 365

Header

NetworkPrefabInfo, 509
SimulationInput, 796
Heap
 NetworkProjectConfig, 533
HEAP_ALIGNMENT
 Allocator, 76
HeapConfiguration, 157
 GlobalsSize, 158
 Init, 158
 PageCount, 158
 PageShift, 158
 ToString, 158
HeapSizeAllocated
 Allocator.Config, 81
HeapSizeUsable
 Allocator.Config, 81
Hide
 DrawIfAttribute, 110
 Fusion, 45
 ScriptHelpAttribute, 749
HideArrayElementLabelAttribute, 159
 HideArrayElementLabelAttribute, 159
HideNetworkObjectInactivityGuard
 NetworkProjectConfig, 533
High
 Fusion, 50
Hitbox, 159
 BoxExtents, 161
 CapsuleExtents, 161
 CapsuleRadius, 162
 ColliderIndex, 163
 DrawGizmos, 161
 Fusion.LagCompensation, 61
 GizmosColor, 162
 HitboxActive, 163
 HitboxIndex, 163
 LagCompensatedHit, 245
 Offset, 162
 OnDrawGizmos, 161
 Position, 163
 PositionRotationQueryParams, 260
 Root, 162
 SetLayer, 161
 SphereRadius, 162
 Type, 162
HitboxActive
 Hitbox, 163
HitboxBufferLengthInMs
 LagCompensationSettings, 277
HitboxColliderContainerDraw, 256
 GetEnumerator, 256
HitboxColliderPosition
 LagCompensatedHit, 245
HitboxColliderRotation
 LagCompensatedHit, 245
HitboxDefaultCapacity
 LagCompensationSettings, 277
Hitboxes
 HitboxRoot, 182
 HitboxesCount
 LagCompensationStatisticsSnapshot, 849
 HitboxIndex
 Hitbox, 163
 HitboxManager, 163
 BVHDepth, 176
 BVHNodes, 177
 DrawInfo, 177
 GetPlayerTickAndAlpha, 166
 GetStatisticsSnapshot, 166
 OverlapBox, 166–168
 OverlapSphere, 168–170
 PositionRotation, 171
 Raycast, 172, 173
 RaycastAll, 174–176
 TotalHitboxes, 177
 HitboxRoot, 177
 BroadRadius, 181
 Config, 181
 ConfigFlags, 179
 Default, 179
 DrawGizmos, 179
 GizmosColor, 181
 Hitboxes, 182
 HitboxRootActive, 182
 IncludeInactiveHitboxes, 179
 InInterest, 182
 InitHitboxes, 179
 IsHitboxActive, 180
 Legacy, 179
 Manager, 183
 MAX_HITBOXES, 182
 Offset, 182
 OnDrawGizmos, 180
 ReinitializeHitboxesBeforeRegistration, 179
 SetHitboxActive, 180
 SetMinBoundingRadius, 181
 HitboxRootActive
 HitboxRoot, 182
 HitboxTypes
 Fusion, 46
 HitOptions
 Fusion, 46
 HitType
 Fusion.LagCompensation, 61
Host
 Fusion, 46, 56
HostMigration
 Fusion, 55
 NetworkProjectConfig, 533
 SimulationConfig, 792
HostMigrationConfig, 183
 EnableAutoUpdate, 183
 UpdateDelay, 183
HostMigrationResume
 StartGameArgs, 835
HostMigrationToken, 184

GameMode, 184
 StartGameArgs, 835

HostMode
 RpcAttribute, 727

HostPlayer
 SimulationRuntimeConfig, 815

IAfterAllTicks, 184
 AfterAllTicks, 184

IAfterClientPredictionReset, 185
 AfterClientPredictionReset, 185

IAfterHostMigration, 185
 AfterHostMigration, 186

IAfterRender, 186
 AfterRender, 186

IAfterSpawned, 186
 AfterSpawned, 187

IAfterTick, 187
 AfterTick, 187

IAfterUpdate, 188
 AfterUpdate, 188

IAsyncOperation, 188
 Completed, 189
 Error, 189
 IsDone, 189

IBeforeAllTicks, 189
 BeforeAllTicks, 190

IBeforeClientPredictionReset, 190
 BeforeClientPredictionReset, 191

IBeforeCopyPreviousState, 191
 BeforeCopyPreviousState, 191

IBeforeHitboxRegistration, 191
 BeforeHitboxRegistration, 192

IBeforeSimulation, 192
 BeforeSimulation, 192

IBeforeTick, 193
 BeforeTick, 193

IBeforeUpdate, 193
 BeforeUpdate, 194

ICoroutine, 194

Id
 NetworkBehaviour, 326
 NetworkObject, 454
 NetworkObjectHeader, 473
 NetworkObjectHeaderPtr, 476
 NetworkObjectMeta, 477

IDataEncryption, 124
 ComputeHash, 124
 DecryptData, 125
 EncryptData, 125
 GenerateKey, 126
 Setup, 126
 VerifyHash, 126

IDespawned, 194
 Despawned, 194

IElementReaderWriter< T >, 195
 GetElementHashCode, 195
 GetElementWordCount, 196
 Read, 196

 ReadRef, 196
 Write, 197

IFixedStorage, 197

Ignore
 Fusion, 47

IgnoreInputAuthority
 Fusion, 46

IInputAuthorityGained, 197
 InputAuthorityGained, 198

IInputAuthorityLost, 198
 InputAuthorityLost, 198

IInterestEnter, 198
 InterestEnter, 199

IInterestExit, 199
 InterestExit, 199

ILocalPrefabCreated, 200
 LocalPrefabCreated, 200

IMessage, 691

InBandwidth
 FusionStatisticsSnapshot, 844
 NetworkObjectStatisticsSnapshot, 854

IncludeBox2D
 Fusion, 46

IncludeInactiveHitboxes
 HitboxRoot, 179

IncludePhysX
 Fusion, 46

IncompatibleConfiguration
 Fusion, 55
 Fusion.Protocol, 62

IndexOf
 NetworkLinkedList< T >, 431
 NetworkLinkedListReadOnly< T >, 436, 437
 NetworkSceneInfo, 629
 NetworkString< TSize >, 658–660

IndexOf< T >
 FixedSize< T >, 138
 NetworkArrayExtensions, 302

IndexOf< TOtherSize >
 NetworkString< TSize >, 661–663

InEditMode
 ReadOnlyAttribute, 707

INetworkArray, 200
 this[int index], 201

INetworkAssetSource< T >, 202
 Acquire, 202
 Description, 203
 IsCompleted, 203
 Release, 203
 WaitForResult, 203

INetworkDictionary, 203
 Add, 204

INetworkInput, 204

INetworkLinkedList, 204
 Add, 205

INetworkObjectInitializer, 205
 InitializeNetworkState, 205

INetworkObjectProvider, 206

AcquirePrefabInstance, 206
ReleaseInstance, 207
INetworkPrefabSource, 207
AssetGuid, 208
INetworkRunnerCallbacks, 208
OnConnectedToServer, 209
OnConnectFailed, 209
OnConnectRequest, 209
OnCustomAuthenticationResponse, 210
OnDisconnectedFromServer, 210
OnHostMigration, 210
OnInput, 210
OnInputMissing, 211
OnObjectEnterAOI, 211
OnObjectExitAOI, 211
OnPlayerJoined, 212
OnPlayerLeft, 212
OnReliableDataProgress, 212
OnReliableDataReceived, 213
OnSceneLoadDone, 213
OnSceneLoadStart, 213
OnSessionListUpdated, 213
OnShutdown, 214
OnUserSimulationMessage, 214
INetworkRunnerUpdater, 214
Initialize, 215
Shutdown, 215
INetworkSceneManager, 216
GetSceneRef, 216, 217
Initialize, 217
IsBusy, 219
IsRunnerScene, 217
LoadScene, 217
MainRunnerScene, 219
MakeDontDestroyOnLoad, 217
MoveGameObjectToScene, 218
OnSceneInfoChanged, 218
Shutdown, 218
TryGetPhysicsScene2D, 218
TryGetPhysicsScene3D, 219
UnloadScene, 219
INetworkStruct, 220
INetworkTRSPTeleport, 220
Teleport, 220
InInterest
 HitboxRoot, 182
Init
 HeapConfiguration, 158
 NetworkBehaviour.ChangeDetector, 331
 NetworkConfiguration, 392
 TickRate, 869
 UnityArraySurrogate< T, ReaderWriter >, 224
 UnityDictionarySurrogate< TKeyType, TValueReaderWriter, TValueType, TValueReaderWriter >, 227
 UnityLinkedListSurrogate< T, ReaderWriter >, 229
 UnitySurrogateBase, 231
 UnityValueSurrogate< T, TReaderWriter >, 233
InitHitboxes
 HitboxRoot, 179
Initial
 Fusion, 48
Initialize
 INetworkRunnerUpdater, 215
 INetworkSceneManager, 217
InitializeNetworkArray< T >
 NetworkBehaviourUtils, 366
InitializeNetworkDictionary< D, K, V >
 NetworkBehaviourUtils, 366
InitializeNetworkList< T >
 NetworkBehaviourUtils, 367
InitializeNetworkState
 INetworkObjectInitializer, 205
 NetworkObjectInitializerUnity, 476
InlineHelpAttribute, 221
 InlineHelpAttribute, 221
 ShowTypeHelp, 221
InObjectUpdates
 FusionStatisticsSnapshot, 844
InPackets
 FusionStatisticsSnapshot, 844
 NetworkObjectStatisticsSnapshot, 854
InPlayMode
 ReadOnlyAttribute, 707
InProgress
 Fusion, 48
INPUT
 AuthorityMasks, 95
InputAuthority
 Fusion, 53
 NetworkObject, 454
 NetworkObjectHeader, 473
 NetworkObjectMeta, 477
InputAuthorityGained
 IInputAuthorityGained, 198
InputAuthorityLost
 IInputAuthorityLost, 198
InputCount
 Simulation, 776
InputDataWordCount
 SimulationConfig, 792
InputInBandwidth
 FusionStatisticsSnapshot, 844
InputOutBandwidth
 FusionStatisticsSnapshot, 844
InputReceiveDelta
 FusionStatisticsSnapshot, 844
InputTotalWordCount
 SimulationConfig, 794
InputTransferMode
 SimulationConfig, 793
InputTransferModes
 SimulationConfig, 792
Instance
 NetworkObjectSortKeyComparer, 487
Instances

NetworkRunner, 605
InstantiateInRunnerScene
 NetworkRunner, 571
InstantiateInRunnerScene< T >
 NetworkRunner, 571
InsufficientSourceAuthority
 Fusion, 52
InsufficientTargetAuthority
 Fusion, 52
Int
 NetworkBehaviourBufferInterpolator, 350
InterestEnter
 IInterestEnter, 199
InterestExit
 IInterestExit, 199
InternalOnDestroy
 NetworkBehaviourUtils, 368
InternalOnDisable
 NetworkBehaviourUtils, 368
InternalOnEnable
 NetworkBehaviourUtils, 368
InternalStruct
 Fusion, 50
InterpAlpha
 SimulationInputHeader, 801
InterpFrom
 SimulationInputHeader, 801
Interpolated
 Fusion, 50
InterpolatedErrorCorrectionSettings, 234
 MaxRate, 234
 MinRate, 235
 PosBlendEnd, 235
 PosBlendStart, 235
 PosMinCorrection, 235
 PosTeleportDistance, 236
 RotBlendEnd, 236
 RotBlendStart, 236
 RotTeleportRadians, 236
InterpolationOffset
 FusionStatisticsSnapshot, 845
InterpolationSpeed
 FusionStatisticsSnapshot, 845
InterpTo
 SimulationInputHeader, 802
Invalid
 Fusion, 50, 55
 Fusion.Sockets.Stun, 64
InvalidArguments
 Fusion, 55
InvalidAuthentication
 Fusion, 55
InvalidEventCode
 Fusion.Protocol, 62
InvalidJoinGameMode
 Fusion.Protocol, 62
InvalidJoinMsgType
 Fusion.Protocol, 62
 InvalidRegion
 Fusion, 55
 InvalidTickRate
 TickRate, 867
 InvertY
 NormalizedRectAttribute, 682
 Invoked
 Fusion, 52
 InvokeLocal
 RpcAttribute, 728
 InvokeRenderInBatchMode
 NetworkProjectConfig, 534
 InvokeRpc
 NetworkBehaviourUtils, 374
 InvokeSceneLoadDone
 NetworkRunner, 572
 InvokeSceneLoadStart
 NetworkRunner, 572
 IPlayerJoined, 237
 PlayerJoined, 237
 IPlayerLeft, 237
 PlayerLeft, 238
 IRemotePrefabCreated, 238
 RemotePrefabCreated, 238
 Is< T >
 NetworkInput, 422
 IsAcquired
 NetworkPrefabTable, 523
 IsActiveOnLoad
 NetworkLoadSceneParameters, 441
 IsBeingDestroyed
 NetworkObjectReleaseContext, 485
 IsBusy
 INetworkSceneManager, 219
 ISceneLoadDone, 239
 SceneLoadDone, 239
 ISceneLoadStart, 239
 SceneLoadStart, 240
 IsClient
 NetworkRunner, 605
 Simulation, 777
 IsCloudReady
 NetworkRunner, 606
 IsCompleted
 INetworkAssetSource< T >, 203
 NetworkSceneAsyncOp.Awaiter, 626
 NetworkSpawnOp.Awaiter, 644
 IsConnectedToServer
 NetworkRunner, 606
 IsCustom
 NetworkObjectTypeid, 496
 IsDone
 IAsyncOperation, 189
 NetworkSceneAsyncOp, 624
 IsFailed
 NetworkSpawnOp, 642
 IsFirstTick
 NetworkRunner, 606

Simulation, 777
IsForward
NetworkRunner, 606
Simulation, 777
IsGlobal
FusionGlobalScriptableObject< T >, 151
IsGlobalLoaded
NetworkProjectConfigAsset, 539
IsGlobalLoadedInternal
FusionGlobalScriptableObject< T >, 151
IsHitboxActive
HitboxRoot, 180
IsIgnored
NetworkObjectFlagsExtensions, 457
ISimulationEnter, 240
SimulationEnter, 240
ISimulationExit, 241
SimulationExit, 241
IsIndex
SceneRef, 748
IsInList
NetBitBufferList, 822
IsInputAuthority
Simulation, 771
IsInSimulation
NetworkObject, 454
IsInterestedIn
NetworkRunner, 572
Simulation, 772
IsInvokeLocal
RpcInfo, 734
IsIPv4
NetAddress, 820
IsIPv6
NetAddress, 820
IsLastTick
NetworkRunner, 606
Simulation, 777
IsLocalPhysics2D
NetworkLoadSceneParameters, 441
IsLocalPhysics3D
NetworkLoadSceneParameters, 442
IsLocalPlayerFirstExecution
Simulation, 777
IsLocalSimulationInputAuthority
Simulation, 772
IsLocalSimulationStateAuthority
Simulation, 772, 773
IsLocalSimulationStateOrInputSource
Simulation, 773
IsMainTRSP
NetworkTRSP, 678
IsMasterClient
PlayerRef, 689
Simulation, 778
IsNestedObject
NetworkObjectReleaseContext, 485
IsNone
NetworkObjectNestingKey, 481
NetworkObjectTypeId, 496
NetworkPrefabId, 507
PlayerRef, 689
IsNothing
Mask256, 282
IsOpen
SessionInfo, 764
StartGameArgs, 835
IsPath
SceneRef, 745
ISpawned, 241
Spawned, 242
IsPlayer
NetworkRunner, 606
Simulation, 778
IsPlayerValid
NetworkRunner, 572
IsPrefab
NetworkObjectTypeId, 496
IsProxy
NetworkBehaviour, 326
NetworkObject, 454
IsQueued
NetworkSpawnOp, 642
IsReadOnly
SerializableDictionary< TKey, TValue >, 755
IsRealPlayer
PlayerRef, 690
IsRelayAddr
NetAddress, 820
IsReserved
NetworkId, 420
IsResimulation
NetworkRunner, 607
Simulation, 778
IsResume
NetworkObject, 452
NetworkRunner, 607
IsRunnerScene
INetworkSceneManager, 217
IsRunning
NetworkRunner, 607
Simulation, 778
TickTimer, 880
IsSceneAuthority
NetworkRunner, 607
IsSceneManagerBusy
NetworkRunner, 607
IsSceneObject
NetworkObjectTypeId, 496
IsServer
NetworkRunner, 607
Simulation, 778
IsSet
NetworkButtons, 384
IsSet< T >
NetworkButtons, 384

IsSharedModeMasterClient
 NetworkRunner, 608

IsShutdown
 NetworkRunner, 608
 Simulation, 778

IsSingleLoad
 NetworkLoadSceneParameters, 442

IsSinglePlayer
 NetworkRunner, 608
 Simulation, 779

IsSpawnable
 NetworkObject, 455

IsSpawned
 NetworkSpawnOp, 642

IsStarting
 NetworkRunner, 608

IsStateAuthority
 Simulation, 774

IsStruct
 NetworkObjectTypeld, 496

IsSynchronous
 NetworkPrefabAcquireContext, 501
 NetworkPrefabInfo, 509

IsTargeted
 SimulationMessage, 806

IStateAuthorityChanged, 242
 StateAuthorityChanged, 242

IsUnreliable
 SimulationMessage, 814

IsValid
 LobbyInfo, 279
 NetAddress, 820
 NetworkBehaviourId, 359
 NetworkId, 420
 NetworkInput, 424
 NetworkObject, 455
 NetworkObjectGuid, 467
 NetworkObjectNestingKey, 481
 NetworkObjectTypeld, 496
 NetworkPrefabId, 507
 NetworkPrefabRef, 517
 NetworkSceneAsyncOp, 624
 NetworkSceneObjectld, 636
 SceneRef, 748
 SerializableType< BaseType >, 759
 SessionInfo, 764
 TickRate, 869, 870

IsVersionCurrent
 NetworkObjectFlagsExtensions, 457

IsVisible
 SessionInfo, 764
 StartGameArgs, 835

IsZero
 Fusion, 44

ItemsPropertyPath
 SerializableDictionary< TKey, TValue >, 754

IUnitySurrogate, 222
 Read, 222

 Write, 222

IUnityValueSurrogate< T >, 223
 DataProperty, 223

JoinSessionLobby
 NetworkRunner, 573

Key
 NetworkRpcStaticWeavedInvokerAttribute, 546
 NetworkRpcWeavedInvokerAttribute, 547
 RpcInvokeData, 736

Keys
 SerializableDictionary< TKey, TValue >, 755

Kilobytes
 Fusion, 57

Kind
 NetworkObjectTypeld, 497

Label
 EditorButtonAttribute, 122
 FieldEditorButtonAttribute, 130

LagCompensatedExt, 256
 SortDistance, 256
 SortReference, 257

LagCompensatedHit, 242
 Collider, 244
 Collider2D, 244
 Distance, 244
 GameObject, 245
 Hitbox, 245
 HitboxColliderPosition, 245
 HitboxColliderRotation, 245
 Normal, 245
 operator LagCompensatedHit, 243, 244
 Point, 245
 Type, 246

LagCompensation
 NetworkProjectConfig, 534
 NetworkRunner, 608

LagCompensationDraw, 257
 BVHDraw, 258
 GizmosDrawWireCapsule, 258
 SnapshotHistoryDraw, 258

LagCompensationSettings, 276
 CachedStaticCollidersSize, 277
 Enabled, 277
 ExpansionFactor, 278
 HitboxBufferLengthInMs, 277
 HitboxDefaultCapacity, 277
 Optimize, 278

LagCompensationStatisticsSnapshot, 848
 AddOnBufferTime, 848
 AddOnBVHTime, 848
 AdvanceBufferTime, 849
 BVHMaxDeep, 849
 BVHNodesCount, 849
 HitboxesCount, 849
 RefitBVHTime, 849
 TotalElapsedTIme, 849

UpdateBufferTime, 850
UpdateBVHTime, 850
LagCompensationUtils.ContactData, 258
 Normal, 259
 Penetration, 259
 Point, 259
LastChild
 Fusion, 57
LastReceiveTick
 NetworkObject, 455
Latest
 Fusion, 50
LatestServerTick
 NetworkRunner, 608
 Simulation, 779
LatestState
 SimulationConfig, 792
LayerAttribute, 278
LayerMask
 Query, 263
 QueryParams, 265
LayerMatrixAttribute, 278
Legacy
 HitboxRoot, 179
Length
 CapacityAttribute, 100
 FixedArray< T >, 139
 NetworkArray< T >, 299
 NetworkArrayReadOnly< T >, 304
 NetworkBehaviourBuffer, 344
 NetworkString< TSize >, 670
 RaycastQuery, 270
 RaycastQueryParams, 271
Lerp
 Angle, 84
Less
 Fusion, 44
LessOrEqual
 Fusion, 44
Load
 FusionGlobalScriptableObjectSourceAttribute, 156
 NetworkPrefabTable, 523
LoadError
 Fusion, 48
LoadId
 NetworkLoadSceneParameters, 441
LoadScene
 INetworkSceneManager, 217
 NetworkRunner, 573, 574
LoadSceneMode
 NetworkLoadSceneParameters, 442
LoadSceneParameters
 NetworkLoadSceneParameters, 442
LobbyInfo, 279
 IsValid, 279
 Name, 279
 NetworkRunner, 609
 Region, 279
Local
 Fusion, 51
LocalAddress
 Simulation, 779
LocalAlpha
 NetworkRunner, 609
 Simulation, 779
LocalhostIPv4
 NetAddress, 819
LocalhostIPv6
 NetAddress, 819
LocalInvokeResult
 RpcInvokeInfo, 737
LocalPhysicsMode
 NetworkLoadSceneParameters, 442
LocalPlayer
 NetworkRunner, 609
 Simulation, 779
LocalPrefabCreated
 ILocalPrefabCreated, 200
LocalRenderTime
 NetworkRunner, 609
LocalToWorldMatrix
 ColliderDrawInfo, 255
LogSimpleUnity, 280
LossChanceMax
 NetworkSimulationConfiguration, 640
LossChanceMin
 NetworkSimulationConfiguration, 640
LossChancePeriod
 NetworkSimulationConfiguration, 640
LossChanceShape
 NetworkSimulationConfiguration, 640
LossChanceThreshold
 NetworkSimulationConfiguration, 640
Low
 Fusion, 50
Lowest
 Fusion, 50
MainRunnerScene
 INetworkSceneManager, 219
MakeDontDestroyOnLoad
 INetworkSceneManager, 217
 NetworkRunner, 575
MakeInitializer< K, V >
 NetworkBehaviour, 317
MakeInitializer< T >
 NetworkBehaviour, 317
MakePtr< T >
 NetworkBehaviour, 317
MakeRef< T >
 NetworkBehaviour, 318, 319
MakeSerializableDictionary< K, V >
 NetworkBehaviourUtils, 369
Manager
 HitboxRoot, 183
Mask
 FieldsMask< T >, 133

Mask256, 280
 Clear, 281
 Equals, 281, 282
 GetBit, 282
 GetHashCode, 282
 IsNothing, 282
 Mask256, 281
 operator long, 282
 operator Mask256, 283
 operator~, 283
 operator&, 283
 operator |, 283
 SetBit, 283
 this[int i], 284
 ToString, 284
MaskBroadcast
 Fusion, 53
MaskCulled
 Fusion, 53
MaskNotSent
 Fusion, 53
MaskSent
 Fusion, 53
MaskVersion
 Fusion, 47
MASTER_CLIENT_RAW
 PlayerRef, 688
MasterClient
 PlayerRef, 690
 SimulationRuntimeConfig, 815
MasterClientObject
 Fusion, 47
MatchmakingMode
 StartGameArgs, 835
MAX
 NetworkRNG, 545
Max
 AABB, 247
 Angle, 84
 RangeExAttribute, 706
MAX_HITBOXES
 HitboxRoot, 182
MAX_INDEX
 NetworkPrefabId, 506
MAX_PAYLOAD_SIZE
 SimulationMessage, 812
MAX_SCENE_OBJECT_INDEX
 NetworkObjectTypeid, 494
MaxCcuReached
 Fusion, 55
MaxConnections
 NetConfig, 828
MaxDepth
 BVHNodeDrawInfo, 253
MaxLateInputs
 TimeSyncConfiguration, 881
MaxLateSnapshots
 TimeSyncConfiguration, 882
 MaxLength
 ArrayLengthAttribute, 92
 MaxPayloadSize
 RpcAttribute, 727
 MaxPlayers
 SessionInfo, 764
 MaxRate
 InterpolatedErrorCorrectionSettings, 234
 MaxScenes
 NetworkSceneInfo, 630
 MaxSize
 NetworkSerializeMethodAttribute, 637
 MaxStringByteCountAttribute, 284
 ByteCount, 285
 Encoding, 285
 MaxStringByteCountAttribute, 285
 Medium
 Fusion, 50
 Megabytes
 Fusion, 57
MemoryStatisticsSnapshot, 850
 BUCKET_COUNT, 851
 BucketFreeBlocksCount, 851
 BucketFullBlocksCount, 851
 BucketUsedBlocksCount, 851
 TargetAllocator, 851
 TotalFreeBlocks, 852
Message
 ErrorIfAttribute, 127
 NetworkObjectSpawnException, 488
 SimulationMessagePtr, 814
 WarnIfAttribute, 911
MessageSize
 RpcSendResult, 738
Meta
 NetworkPrefabAcquireContext, 501
META_WORD_COUNT
 NetworkDictionary< K, V >, 401
META_WORDS
 NetworkLinkedList< T >, 432
 NetworkLinkedListReadOnly< T >, 437
Method
 RenderAttribute, 723
 RpcHeader, 732
MethodName
 OnChangedRenderAttribute, 683
MilliSecs
 Fusion, 57
Min
 AABB, 247
 Angle, 85
 RangeExAttribute, 706
MinLength
 ArrayLengthAttribute, 92
MinRate
 InterpolatedErrorCorrectionSettings, 235
Mode
 DrawIfAttribute, 110

NetworkRunner, 609
Simulation, 779
Modes
 SimulationBehaviourAttribute, 789
MonitorNetworkObjectStatistics
 NetworkObjectStatisticsManager, 853
MoveGameObjectToSameScene
 NetworkRunner, 575
MoveGameObjectToScene
 INetworkSceneManager, 218
 NetworkRunner, 575
MoveNext
 FixedArray< T >.Enumerator, 141
 NetworkArray< T >.Enumerator, 301
 NetworkBehaviour.ChangeDetector.Enumerator,
 333
 NetworkDictionary< K, V >.Enumerator, 403
 NetworkLinkedList< T >.Enumerator, 434
MoveToRunnerScene
 NetworkRunner, 576
MoveToRunnerScene< T >
 NetworkRunner, 576
Multiple
 NetworkProjectConfig, 530
Multiplier
 Fusion, 57
Name
 DisplayNameAttribute, 105
 LobbyInfo, 279
 NetworkObject, 455
 SessionInfo, 765
NATTType
 Fusion.Sockets.Stun, 64
 NetworkRunner, 609
NestedComponentUtilities, 285
 EnsureRootComponentExists< T, TStopOn >, 287
 FindObjectsOfTypeInOrder< T >, 287
 FindObjectsOfTypeInOrder< T, TCast >, 289
 GetNestedComponentInChildren< T, TStopOn >,
 290
 GetNestedComponentInParent< T, TStopOn >,
 291
 GetNestedComponentInParents< T, TStopOn >,
 291
 GetNestedComponentsInChildren< T >, 291
 GetNestedComponentsInChildren< T, TSearch,
 TStop >, 292
 GetNestedComponentsInChildren< T, TStopOn >,
 292
 GetNestedComponentsInParents< T >, 292
 GetNestedComponentsInParents< T, TStop >,
 293
 GetParentComponent< T >, 293
NestedObjects
 NetworkObject, 452
NestingKey
 NetworkObjectHeader, 473
NestingRoot
 NetworkObjectHeader, 474
NetAddress, 816
 ActorId, 819
 Any, 817
 AnyIPv6, 817
 CreateFromIpPort, 818
 FromActorId, 818
 HasAddress, 820
 IsIPv4, 820
 IsIPv6, 820
 IsRelayAddr, 820
 IsValid, 820
 LocalhostIPv4, 819
 LocalhostIPv6, 819
NetBitBufferList, 821
 AddFirst, 821
 AddLast, 821
 IsInList, 822
 Remove, 822
 RemoveHead, 822
NetCommandAccepted, 823
NetCommandConnect, 823
NetCommandDisconnect, 824
NetCommandHeader, 824
 Create, 825
NetCommandRefused, 825
NetCommands
 Fusion.Sockets, 63
NetConfig, 826
 Address, 827
 ConnectAttempts, 827
 ConnectInterval, 827
 ConnectionDefaultRtt, 827
 ConnectionGroups, 827
 ConnectionPingInterval, 828
 ConnectionSendBuffers, 828
 ConnectionShutdownTime, 828
 ConnectionsPerGroup, 830
 ConnectionTimeout, 828
 Defaults, 830
 MaxConnections, 828
 Notify, 828
 OperationExpireTime, 829
 PacketSize, 829
 PacketSizeInBits, 830
 Simulation, 829
 SocketRecvBuffer, 829
 SocketSendBuffer, 829
NetConfigPointer
 Simulation, 780
NetConnectFailedReason
 Fusion.Sockets, 63
NetDisconnectReason
 Fusion.Sockets, 64
NetPacketType
 Fusion.Sockets, 64
Network
 NetworkProjectConfig, 534

NetworkArray
 NetworkArray< T >, 295
 NetworkArray< T >, 293
 Clear, 295
 CopyFrom, 296
 CopyTo, 297
 Get, 298
 GetEnumerator, 298
 Length, 299
 NetworkArray, 295
 operator NetworkArrayReadOnly< T >, 298
 Set, 298
 this[int index], 299
 ToArray, 298
 ToStringString, 299
 ToReadOnly, 299
 ToString, 299
 NetworkArray< T >.Enumerator, 300
 Current, 301
 Dispose, 301
 Enumerator, 300
 MoveNext, 301
 Reset, 301
 NetworkArrayExtensions, 302
 GetRef< T >, 302
 IndexOf< T >, 302
 NetworkArrayReadOnly< T >, 303
 Length, 304
 this[int index], 304
 NetworkAssemblyIgnoreAttribute, 304
 NetworkAssemblyWeavedAttribute, 304
 NetworkBehaviour, 304
 ChangedTick, 325
 CopyBackingFieldsToState, 308
 CopyStateFrom, 308
 CopyStateToBackingFields, 308
 Despawned, 308
 DynamicWordCount, 326
 FixedUpdateNetwork, 309
 GetArrayReader< T >, 309
 GetBehaviourReader< T >, 310, 311
 GetBehaviourReader< TBehaviour, TProperty >, 311
 GetChangeDetector, 312
 GetDictionaryReader< K, V >, 312, 313
 GetInput< T >, 313
 GetLinkListReader< T >, 314
 GetLocalAuthorityMask, 315
 GetPropertyReader< T >, 315
 GetPropertyReader< TBehaviour, TProperty >, 316
 HasInputAuthority, 326
 HasStateAuthority, 326
 Id, 326
 IsProxy, 326
 MakeInitializer< K, V >, 317
 MakeInitializer< T >, 317
 MakePtr< T >, 317
 MakeRef< T >, 318, 319
 NetworkDeserialize, 319
 NetworkSerialize, 320
 NetworkUnwrap, 321
 NetworkWrap, 321
 offset, 325
 operator NetworkBehaviourId, 321
 ReinterpretState< T >, 323
 ReplicateTo, 323, 324
 ReplicateToAll, 324
 ResetState, 324
 Spawned, 324
 StateBuffer, 326
 StateBufferIsValid, 327
 TryGetSnapshotsBuffers, 325
 NetworkBehaviour.ArrayReader< T >, 327
 Read, 327
 NetworkBehaviour.BehaviourReader< T >, 328
 Read, 328
 T, 329
 NetworkBehaviour.ChangeDetector, 329
 DetectChanges, 330, 331
 Init, 331
 SimulationState, 330
 SnapshotFrom, 330
 SnapshotTo, 330
 Source, 330
 NetworkBehaviour.ChangeDetector.Enumerable, 332
 Changed, 332
 GetEnumerator, 332
 NetworkBehaviour.ChangeDetector.Enumerator, 333
 Current, 334
 MoveNext, 333
 Reset, 333
 NetworkBehaviour.DictionaryReader< K, V >, 334
 Read, 334
 NetworkBehaviour.LinkListReader< T >, 335
 Read, 335
 NetworkBehaviour.PropertyReader< T >, 336
 PropertyReader, 336
 Read, 337
 T, 337
 NetworkBehaviourBuffer, 337
 Length, 344
 operator bool, 338
 Read, 339, 341
 Read< T >, 341, 343
 ReinterpretState< T >, 343
 this[int index], 344
 Tick, 344
 Valid, 345
 NetworkBehaviourBufferInterpolator, 345
 Alpha, 355
 Angle, 347
 Behaviour, 355
 Bool, 347, 348
 Float, 348
 From, 355

Int, 350
NetworkBehaviourBufferInterpolator, 346
operator bool, 350
Quaternion, 351
Select< T >, 351, 352
To, 355
Valid, 355
Vector2, 352, 353
Vector3, 353, 354
Vector4, 354
NetworkBehaviourId, 356
Behaviour, 359
Equals, 357
GetHashCode, 357
IsValid, 359
None, 359
Object, 359
operator!=, 358
operator==, 358
SIZE, 359
ToString, 358
NetworkBehaviourUtils, 360
CloneArray< T >, 362
CopyFromNetworkArray< T >, 362
CopyFromNetworkDictionary< D, K, V >, 363
CopyFromNetworkList< T >, 363
GetMetaData, 364
GetRpcStaticIndexOrThrow, 364
GetStaticWordCount, 365
GetWordCount, 365
HasStaticWordCount, 365
InitializeNetworkArray< T >, 366
InitializeNetworkDictionary< D, K, V >, 366
InitializeNetworkList< T >, 367
InternalOnDestroy, 368
InternalOnDisable, 368
InternalOnEnable, 368
InvokeRpc, 374
MakeSerializableDictionary< K, V >, 369
NotifyLocalSimulationNotAllowedToSendRpc, 369
NotifyLocalTargetedRpcCulled, 370
NotifyNetworkUnwrapFailed< T >, 370
NotifyNetworkWrapFailed< T >, 370, 371
NotifyRpcPayloadSizeExceeded, 371
NotifyRpcTargetUnreachable, 371
RegisterMetaData, 372
RegisterRpcInvokeDelegates, 372
ShouldRegisterRpcInvokeDelegates, 372
ThrowIfBehaviourNotInitialized, 373
TryGetRpcInvokeDelegateArray, 373
TryGetRpcStaticInvokeDelegate, 373
NetworkBehaviourUtils.ArrayInitializer< T >, 374
operator NetworkArray< T >, 375
operator NetworkLinkedList< T >, 375
NetworkBehaviourUtils.DictionaryInitializer< K, V >, 376
operator NetworkDictionary< K, V >, 376
NetworkBehaviourUtils.MetaData, 377
NetworkBehaviourWeavedAttribute, 377
NetworkBehaviourWeavedAttribute, 377
WordCount, 378
NetworkBool, 378
Equals, 379
GetHashCode, 380
NetworkBool, 379
operator bool, 380
operator NetworkBool, 380
ToString, 380
NetworkButtons, 381
Bits, 391
Equals, 382, 383
GetHashCode, 383
GetPressed, 383
GetReleased, 384
IsSet, 384
IsSet< T >, 384
NetworkButtons, 382, 390
Set, 386
Set< T >, 386
SetAllDown, 387
SetAllUp, 387
SetDown, 387
SetDown< T >, 387
SetUp, 388
SetUp< T >, 388
WasPressed, 389
WasPressed< T >, 389
WasReleased, 389
WasReleased< T >, 390
NetworkConditions
NetworkProjectConfig, 534
NetworkConfiguration, 391
ClientToClientWithServerProxy, 392
ClientToServer, 392
ConnectAttempts, 393
ConnectInterval, 393
ConnectionDefaultRtt, 394
ConnectionPingInterval, 394
ConnectionShutdownTime, 393
ConnectionTimeout, 393
Init, 392
ReliableDataTransferModes, 393
ReliableDataTransfers, 392
SocketRecvBufferSize, 394
SocketSendBufferSize, 394
NetworkConnected
Simulation, 774
NetworkDelegates, 394
NetworkDeserialize
NetworkBehaviour, 319
NetworkDeserializeMethodAttribute, 395
NetworkDictionary
NetworkDictionary< K, V >, 398
NetworkDictionary< K, V >, 395
Add, 398
Capacity, 402

Clear, 399
 ContainsKey, 399
 ContainsValue, 399
 Count, 402
 Get, 399
 GetEnumerator, 399
 META_WORD_COUNT, 401
 NetworkDictionary, 398
 operator NetworkDictionaryReadOnly< K, V >, 399
 Remove, 400
 Set, 401
 this[K key], 402
 ToReadOnly, 401
 TryGet, 401
 NetworkDictionary< K, V >.Enumerator, 402
 Current, 403
 Dispose, 403
 MoveNext, 403
 Reset, 403
 NetworkDictionaryReadOnly< K, V >, 404
 Capacity, 406
 Count, 406
 Get, 406
 TryGet, 406
 NetworkDisconnected
 Simulation, 775
 NetworkedAttribute, 407
 Default, 407
 NetworkedAttribute, 407
 NetworkedBehaviours
 NetworkObject, 453
 NetworkedWeavedAttribute, 408
 NetworkedWeavedAttribute, 408
 WordCount, 409
 WordOffset, 409
 NetworkEvents, 409
 NetworkEvents.ConnectFailedEvent, 410
 NetworkEvents.ConnectRequestEvent, 411
 NetworkEvents.CustomAuthenticationResponse, 411
 NetworkEvents.DisconnectFromServerEvent, 411
 NetworkEvents.HostMigrationEvent, 411
 NetworkEvents.InputEvent, 412
 NetworkEvents.InputPlayerEvent, 412
 NetworkEvents.ObjectEvent, 412
 NetworkEvents.ObjectPlayerEvent, 412
 NetworkEvents.PlayerEvent, 413
 NetworkEvents.ReliableDataEvent, 413
 NetworkEvents.ReliableProgressEvent, 413
 NetworkEvents.RunnerEvent, 413
 NetworkEvents.SessionListUpdateEvent, 414
 NetworkEvents.ShutdownEvent, 414
 NetworkEvents.SimulationMessageEvent, 414
 NetworkId, 414
 ALIGNMENT, 419
 BLOCK_SIZE, 419
 Comparer, 420
 CompareTo, 416
 Equals, 416, 417
 GetHashCode, 417
 IsReserved, 420
 IsValid, 420
 operator bool, 417
 operator!=, 417
 operator==, 417
 Raw, 419
 Read, 418
 SIZE, 419
 ToNamePrefixString, 418
 ToString, 418
 Write, 418, 419
 NetworkId.EqualityComparer, 420
 Equals, 421
 GetHashCode, 421
 NetworkIdsObjectName
 NetworkProjectConfig, 534
 NetworkInput, 421
 Convert< T >, 422
 Data, 424
 Get< T >, 422
 Is< T >, 422
 IsValid, 424
 Set< T >, 423
 TryGet< T >, 423
 TrySet< T >, 423
 Type, 424
 WordCount, 424
 NetworkInputUtils, 424
 GetMaxWordCount, 425
 GetType, 425
 GetTypeKey, 425
 GetWordCount, 426
 NetworkInputWeavedAttribute, 426
 NetworkInputWeavedAttribute, 426
 WordCount, 427
 NetworkLinkedList
 NetworkLinkedList< T >, 429
 NetworkLinkedList< T >, 427
 Add, 429, 430
 Capacity, 433
 Clear, 430
 Contains, 430
 Count, 433
 ELEMENT_WORDS, 432
 Get, 430
 GetEnumerator, 431
 IndexOf, 431
 META_WORDS, 432
 NetworkLinkedList, 429
 Remap, 431
 Remove, 432
 Set, 432
 this[int index], 433
 NetworkLinkedList< T >.Enumerator, 433
 Current, 434
 Dispose, 434

MoveNext, 434
Reset, 434
NetworkLinkedListReadOnly< T >, 435
Capacity, 438
Contains, 436
Count, 438
ELEMENT_WORDS, 437
Get, 436
IndexOf, 436, 437
META_WORDS, 437
this[int index], 438
NetworkLoadSceneParameters, 438
Equals, 439, 440
GetHashCode, 440
IsActiveOnLoad, 441
IsLocalPhysics2D, 441
IsLocalPhysics3D, 442
IsSingleLoad, 442
LoadId, 441
LoadSceneMode, 442
LoadSceneParameters, 442
LocalPhysicsMode, 442
operator!=, 440
operator==, 440
ToString, 441
NetworkMecanimAnimator, 443
Animator, 444
ApplyTiming, 444
DynamicWordCount, 445
SetTrigger, 443, 444
NetworkObject, 445
AssignInputAuthority, 447
Awake, 448
CopyStateFrom, 448
Flags, 452
GetLocalAuthorityMask, 448
GetWordCount, 448
HasInputAuthority, 454
HasStateAuthority, 454
Id, 454
InputAuthority, 454
IsInSimulation, 454
IsProxy, 454
IsResume, 452
IsSpawnable, 455
IsValid, 455
LastReceiveTick, 455
Name, 455
NestedObjects, 452
NetworkedBehaviours, 453
NetworkTypeId, 453
NetworkUnwrap, 449
NetworkWrap, 449, 450
OnDestroy, 450
operator NetworkId, 450
PriorityCallback, 453
PriorityLevelDelegate, 450
ReleaseStateAuthority, 451
RemoveInputAuthority, 451
RenderSource, 455
RenderTime, 455
RenderTimeframe, 456
ReplicateTo, 453
ReplicateToDelegate, 451
RequestStateAuthority, 452
Runner, 456
SetPlayerAlwaysInterested, 452
SortKey, 453
StateAuthority, 456
NetworkObjectAcquireResult
Fusion, 46
NetworkObjectFlags
Fusion, 47
NetworkObjectFlagsExtensions, 456
GetVersion, 457
IsIgnored, 457
IsVersionCurrent, 457
SetCurrentVersion, 458
SetIgnored, 458
NetworkObjectGuid, 458
ALIGNMENT, 466
CompareTo, 462
Empty, 467
Equals, 463
GetHashCode, 463
IsValid, 467
NetworkObjectGuid, 460, 462
operator Guid, 463
operator NetworkObjectGuid, 464
operator NetworkPrefabRef, 464
operator!=, 465
operator==, 465
Parse, 465
RawGuidValue, 467
SIZE, 467
ToString, 465, 466
ToUnityGuidString, 466
TryParse, 466
NetworkObjectGuid.EqualityComparer, 467
Equals, 468
GetHashCode, 468
NetworkObjectHeader, 468
_reserved, 473
BehaviourCount, 473
ByteCount, 475
Equals, 470
Flags, 473
GetBehaviourChangedTickArray, 470
GetDataPointer, 470
GetDataWordCount, 471
GetHashCode, 471
GetMainNetworkTRSPData, 471
HasMainNetworkTRSP, 472
Id, 473
InputAuthority, 473
NestingKey, 473

NestingRoot, 474
operator!=, 472
operator==, 472
PLAYER_DATA_WORD, 474
SIZE, 474
StateAuthority, 474
ToString, 472
Type, 474
WordCount, 474
WORDS, 475
NetworkObjectHeaderFlags
 Fusion, 47
NetworkObjectHeaderPtr, 475
 Id, 476
 Ptr, 476
 Type, 476
NetworkObjectInitializerUnity, 476
 InitializeNetworkState, 476
NetworkObjectMeta, 477
 Id, 477
 InputAuthority, 477
 StateAuthority, 477
 Type, 478
NetworkObjectNestingKey, 478
 ALIGNMENT, 481
 Equals, 479, 480
 GetHashCode, 480
 IsNone, 481
 IsValid, 481
 NetworkObjectNestingKey, 479
 SIZE, 481
 ToString, 480
 Value, 481
NetworkObjectNestingKey.EqualityComparer, 482
 Equals, 482
 GetHashCode, 482
NetworkObjectPrefabData, 483
 Guid, 483
NetworkObjectProviderDummy, 483
NetworkObjectReleaseContext, 484
 IsBeingDestroyed, 485
 IsNestedObject, 485
 NetworkObjectReleaseContext, 484
 Object, 485
 ToString, 485
 TypeId, 485
NetworkObjectSortKeyComparer, 486
 Compare, 486
 Instance, 487
NetworkObjectSpawnDelegate
 Fusion, 58
NetworkObjectSpawnException, 487
 Message, 488
 NetworkObjectSpawnException, 487
 Status, 488
 TypeId, 488
NetworkObjectStatisticsManager, 852
 ClearMonitoredNetworkObjects, 852
GetNetworkObjectStatistics, 852
MonitorNetworkObjectStatistics, 853
NetworkObjectStatisticsSnapshot, 853
 InBandwidth, 854
 InPackets, 854
 OutBandwidth, 854
 OutPackets, 854
NetworkObjectTypeId, 488
 _value0, 494
 _value1, 494
 ALIGNMENT, 494
 AsCustom, 494
 AsInternalStructId, 495
 AsPrefabId, 495
 AsSceneObjectId, 495
 Comparer, 495
 Equals, 490, 491
 FromCustom, 491
 FromPrefabId, 491
 FromSceneRefAndObjectIndex, 492
 FromStruct, 492
 GetHashCode, 493
 IsCustom, 496
 IsNone, 496
 IsPrefab, 496
 IsSceneObject, 496
 IsStruct, 496
 IsValid, 496
 Kind, 497
 MAX_SCENE_OBJECT_INDEX, 494
 operator NetworkObjectTypeId, 493
 operator!=, 493
 operator==, 493
 PlayerData, 497
 SIZE, 494
 ToString, 493
NetworkObjectTypeId.EqualityComparer, 497
 Equals, 497
 GetHashCode, 498
NetworkPhysicsInfo, 499
 SIZE, 499
 TimeScale, 499
 WORD_COUNT, 499
NetworkPrefabAcquireContext, 500
 Data, 501
 DontDestroyOnLoad, 501
 HasHeader, 502
 IsSynchronous, 501
 Meta, 501
 NetworkPrefabAcquireContext, 500
 PrefabId, 501
NetworkPrefabAttribute, 502
NetworkPrefabId, 502
 ALIGNMENT, 506
 AsIndex, 507
 CompareTo, 504
 Equals, 504
 FromIndex, 504

FromRaw, 505
GetHashCode, 505
IsNone, 507
IsValid, 507
MAX_INDEX, 506
operator!=, 505
operator==, 505
RawValue, 506
SIZE, 506
ToString, 505, 506
NetworkPrefabId.EqualityComparer, 507
Equals, 508
GetHashCode, 508
NetworkPrefabInfo, 508
Data, 509
HasHeader, 509
Header, 509
IsSynchronous, 509
Prefab, 509
NetworkPrefabRef, 510
ALIGNMENT, 516
CompareTo, 512
Empty, 517
Equals, 513
GetHashCode, 513
IsValid, 517
NetworkPrefabRef, 511, 512
operator Guid, 514
operator NetworkObjectGuid, 514
operator NetworkPrefabRef, 514
operator!=, 515
operator==, 515
Parse, 515
RawGuidValue, 517
SIZE, 517
ToString, 515, 516
ToUnityGuidString, 516
TryParse, 516
NetworkPrefabRef.EqualityComparer, 517
Equals, 518
GetHashCode, 518
NetworkPrefabTable, 518
AddInstance, 520
AddSource, 520
Clear, 520
Contains, 521
GetEntries, 521
GetGuid, 521
GetId, 521
GetInstanceCount, 522
GetSource, 522, 523
IsAcquired, 523
Load, 523
Options, 526
Prefabs, 526
RemoveInstance, 524
TryAddSource, 524
Unload, 525
UnloadAll, 525
UnloadUnreferenced, 525
Version, 526
NetworkPrefabTableGetPrefabResult
Fusion, 48
NetworkPrefabTableOptions, 526
Default, 527
UnloadPrefabOnReleasingLastInstance, 527
UnloadUnusedPrefabsOnShutdown, 527
NetworkProjectConfig, 527
AssembliesToWeave, 532
BuildTypes, 532
CheckNetworkedPropertiesBeingEmpty, 532
CheckRpcAttributeUsage, 532
CurrentTypeld, 532
CurrentVersion, 532
DefaultResourceName, 533
Deserialize, 530
EncryptionConfig, 533
EnqueueIncompleteSynchronousSpawns, 533
GetExecutionOrder, 531
Global, 536
Heap, 533
HideNetworkObjectInactivityGuard, 533
HostMigration, 533
InvokeRenderInBatchMode, 534
LagCompensation, 534
Multiple, 530
Network, 534
NetworkConditions, 534
NetworkIdsObjectName, 534
None, 530
NullChecksForNetworkedProperties, 534
PeerMode, 535
PeerModes, 529
PrefabTable, 535
ReplicationFeatures, 530
Scheduling, 530
SchedulingAndInterestManagement, 530
Serialize, 531
Simulation, 535
Single, 530
TimeSynchronizationOverride, 535
ToString, 531
Typeld, 535
UnloadGlobal, 531
UseSerializableDictionary, 535
Version, 536
NetworkProjectConfigAsset, 536
BehaviourMeta, 538
Config, 538
Global, 539
IsGlobalLoaded, 539
OnDisable, 537
PrefabOptions, 538
Prefabs, 538
TryGetGlobal, 537
UnloadGlobal, 538

NetworkProjectConfigAsset.SerializableSimulationBehaviourMethod
GetInterfaceListNext, 565
GetInterfaceListPrev, 565
GetInterfaceListsCount, 565
GetMemorySnapshot, 566
GetObjectsInAreaOfInterestForPlayer, 566
GetPhysicsScene, 566
GetPhysicsScene2D, 566
GetPlayerActorId, 567
GetPlayerConnectionToken, 567
GetPlayerConnectionType, 567
GetPlayerObject, 568
GetPlayerRtt, 568
GetPlayerUserId, 568
GetRawInputForPlayer, 569
GetResumeSnapshotNetworkObjects, 569
GetResumeSnapshotNetworkSceneObjects, 569
GetRpcTargetStatus, 569
GetRunnerForGameObject, 570
GetRunnerForScene, 570
GetSingleton< T >, 570
HasSingleton< T >, 571
Instances, 605
InstantiateInRunnerScene, 571
InstantiateInRunnerScene< T >, 571
InvokeSceneLoadDone, 572
InvokeSceneLoadStart, 572
IsClient, 605
IsCloudReady, 606
IsConnectedToServer, 606
IsFirstTick, 606
IsForward, 606
IsInterestedIn, 572
IsLastTick, 606
IsPlayer, 606
IsPlayerValid, 572
IsResimulation, 607
IsResume, 607
IsRunning, 607
IsSceneAuthority, 607
IsSceneManagerBusy, 607
IsServer, 607
IsSharedModeMasterClient, 608
IsShutdown, 608
IsSinglePlayer, 608
IsStarting, 608
JoinSessionLobby, 573
LagCompensation, 608
LatestServerTick, 608
LoadScene, 573, 574
LobbyInfo, 609
LocalAlpha, 609
LocalPlayer, 609
LocalRenderTime, 609
MakeDontDestroyOnLoad, 575
Mode, 609
MoveGameObjectToSameScene, 575
MoveGameObjectToScene, 575
MoveToRunnerScene, 576

NetworkReceiveDone
Simulation, 775

NetworkRNG, 540
MAX, 545
NetworkRNG, 541
Next, 542
NextExclusive, 542
NextInt32, 542
NextSingle, 542
NextSingleExclusive, 542
NextUInt32, 543
Peek, 545
RangeExclusive, 543
RangeInclusive, 543, 544
SIZE, 545
ToString, 544

NetworkRpcStaticWeavedInvokerAttribute, 545
Key, 546

NetworkRpcStaticWeavedInvokerAttribute, 546

NetworkRpcWeavedInvokerAttribute, 546
Key, 547

NetworkRpcWeavedInvokerAttribute, 547
Sources, 547
Targets, 548

NetworkRunner, 548
ActivePlayers, 604
AddCallbacks, 558
AddGlobal, 558
AddPlayerAreaOfInterest, 558
Attach, 558, 559
AuthenticationValues, 604
BuildType, 604
BuildTypes, 557
CanSpawn, 604
ClearPlayerAreaOfInterest, 559
Config, 605
CurrentConnectionType, 605
Debug, 557
DeltaTime, 605
Despawn, 559
DestroySingleton< T >, 560
Disconnect, 560
EnsureRunnerScenesActive, 560
Exists, 561
FindObject, 561
GameMode, 605
GetAllBehaviours, 561
GetAllBehaviours< T >, 562
GetAllNetworkObjects, 563
GetAreaOfInterestGizmoData, 563
GetAvailableRegions, 563
GetInputForPlayer< T >, 564
GetInstanceEnumerator, 564
GetInterfaceListHead, 564

MoveToRunnerScene< T >, 576
NATTType, 609
ObjectAcquired, 612
ObjectDelegate, 576
ObjectProvider, 610
OnBeforeSpawned, 576
Prefabs, 610
ProvideInput, 610
PushHostMigrationSnapshot, 577
RegisterSceneObjects, 577
Release, 557
RemoteRenderTime, 610
RemoveCallbacks, 577
RemoveGlobal, 578
RenderInternal, 578
Running, 557
SceneManager, 610
SendReliableDataToPlayer, 578
SendReliableDataToServer, 578
SendRpc, 579
SessionInfo, 610
SetAreaOfInterestCellSize, 579
SetAreaOfInterestGrid, 581
SetBehaviourReplicateTo, 581
SetBehaviourReplicateToAll, 581
SetIsSimulated, 582
SetMasterClient, 582
SetPlayerAlwaysInterested, 582
SetPlayerObject, 583
SetSimulateMultiPeerPhysics, 583
Shutdown, 557, 583
SimulationTime, 611
SimulationUnityScene, 611
SinglePlayerContinue, 584
SinglePlayerPause, 584
Spawn, 584–587
Spawn< T >, 587
SpawnAsync, 588–590
SpawnAsync< T >, 591
Stage, 611
StartGame, 592
Starting, 557
State, 611
States, 557
Tick, 611
TickRate, 611
TicksExecuted, 612
Topology, 612
TryFindBehaviour, 592
TryFindBehaviour< T >, 593
TryFindObject, 593
TryGetBehaviourStatistics, 594
TryGetFusionStatistics, 594
TryGetInputForPlayer< T >, 594
TryGetNetworkedBehaviourFromNetworkedObjectRef< T >, 595
TryGetNetworkedBehaviourId, 595
TryGetObjectRefFromNetworkedBehaviour, 596
TryGetPhysicsInfo, 596
TryGetPlayerObject, 596
TryGetSceneInfo, 597
TrySetPhysicsInfo, 597
TrySpawn, 598, 600, 601
TrySpawn< T >, 602
UnloadScene, 602
UpdateInternal, 604
UserId, 612
NetworkRunnerCallbackArgs, 612
NetworkRunnerCallbackArgs.ConnectRequest, 613
 Accept, 613
 Refuse, 613
 RemoteAddress, 614
 Waiting, 614
NetworkRunnerUpdaterDefault, 614
 RegisterInPlayerLoop, 615
 RenderSettings, 616
 UnregisterFromPlayerLoop, 615
 UpdateSettings, 616
NetworkRunnerUpdaterDefault.NetworkRunnerRender, 616
NetworkRunnerUpdaterDefault.NetworkRunnerUpdate, 616
NetworkRunnerUpdaterDefault.InvokeSettings, 617
 AddMode, 619
 Equals, 617, 618
 GetHashCode, 618
 operator!=, 618
 operator==, 619
 ReferencePlayerLoopSystem, 620
 ToString, 619
NetworkSceneAsyncOp, 620
 AddOnCompleted, 621
 Error, 624
 FromAsyncOperation, 621
 FromCompleted, 622
 FromCoroutine, 622
 FromError, 623
 FromTask, 623
 GetAwaiter, 623
 IsDone, 624
 IsValid, 624
 SceneRef, 624
NetworkSceneAsyncOp.Awaiter, 625
 Awaiter, 625
 GetResult, 626
 IsCompleted, 626
 OnCompleted, 626
NetworkSceneInfo, 626
 AddSceneRef, 628
 Equals, 628
 GetHashCode, 629
 IndexOf, 629
 MaxScenes, 630
 operator NetworkSceneInfo, 629
 RemoveSceneRef, 630
 SceneCount, 631

SceneParams, [631](#)
Scenes, [631](#)
SIZE, [630](#)
ToString, [630](#)
Version, [631](#)
WORD_COUNT, [631](#)
NetworkSceneInfoChangeSource
 Fusion, [48](#)
NetworkSceneInfoDefaultFlags
 Fusion, [48](#)
NetworkSceneLoadId, [632](#)
 Equals, [632](#), [633](#)
 GetHashCode, [633](#)
 NetworkSceneLoadId, [632](#)
 Value, [633](#)
NetworkSceneObjectId, [634](#)
 Equals, [634](#), [635](#)
 GetHashCode, [635](#)
 IsValid, [636](#)
 Objectld, [635](#)
 Scene, [636](#)
 SceneLoadId, [636](#)
 ToString, [635](#)
NetworkSerialize
 NetworkBehaviour, [320](#)
NetworkSerializeMethodAttribute, [636](#)
 MaxSize, [637](#)
NetworkSimulationConfiguration, [637](#)
 AdditionalJitter, [638](#)
 AdditionalLoss, [639](#)
 Clone, [638](#)
 Create, [638](#)
 DelayMax, [639](#)
 DelayMin, [639](#)
 DelayPeriod, [639](#)
 DelayShape, [639](#)
 DelayThreshold, [639](#)
 Enabled, [640](#)
 LossChanceMax, [640](#)
 LossChanceMin, [640](#)
 LossChancePeriod, [640](#)
 LossChanceShape, [640](#)
 LossChanceThreshold, [640](#)
NetworkSpawnFlags
 Fusion, [49](#)
NetworkSpawnOp, [641](#)
 GetAwaiter, [642](#)
 IsFailed, [642](#)
 IsQueued, [642](#)
 IsSpawned, [642](#)
 Object, [642](#)
 Runner, [642](#)
 Status, [643](#)
NetworkSpawnOp.Awaiter, [643](#)
 Awaiter, [643](#)
 GetResult, [644](#)
 IsCompleted, [644](#)
 OnCompleted, [644](#)
NetworkSpawnStatus
 Fusion, [49](#)
NetworkString
 NetworkString< TSize >, [648](#)
NetworkString< TSize >, [645](#)
 Assign, [648](#)
 Capacity, [670](#)
 Compare, [648](#), [649](#)
 Compare< TOtherSize >, [649](#), [650](#)
 Contains, [650](#), [651](#)
 Contains< TOtherSize >, [652](#)
 EndsWith, [653](#)
 EndsWith< TOtherSize >, [653](#)
 Equals, [654](#), [655](#)
 Equals< TOtherSize >, [655](#), [656](#)
 Get, [656](#)
 GetCapacity< TSize >, [657](#)
 GetCharCount, [657](#)
 GetEnumerator, [657](#)
 GetHashCode, [657](#)
 IndexOf, [658](#)–[660](#)
 IndexOf< TOtherSize >, [661](#)–[663](#)
 Length, [670](#)
 NetworkString, [648](#)
 operator NetworkString< TSize >, [663](#)
 operator string, [664](#)
 operator!=, [664](#), [665](#)
 operator==, [665](#), [666](#)
 Set, [666](#)
 StartsWith, [667](#)
 StartsWith< TOtherSize >, [667](#)
 Substring, [668](#)
 this[int index], [670](#)
 ToLower, [669](#)
 ToString, [669](#)
 ToUpper, [669](#)
 Value, [670](#)
NetworkStructUtils, [670](#)
 GetWordCount< T >, [671](#)
NetworkStructWeavedAttribute, [671](#)
 NetworkStructWeavedAttribute, [672](#)
 WordCount, [672](#)
NetworkTransform, [672](#)
 AutoUpdateAreaOfInterestOverride, [675](#)
 DisableSharedModelInterpolation, [674](#)
 SetAreaOfInterestOverride, [673](#)
 SyncParent, [674](#)
 SyncScale, [674](#)
 Teleport, [674](#)
NetworkTRSP, [675](#)
 Data, [677](#)
 IsMainTRSP, [678](#)
 Render, [676](#)
 ResolveAOIOVERRIDE, [676](#)
 SetAreaOfInterestOverride, [677](#)
 SetParentTransform, [677](#)
 State, [678](#)
 Teleport, [677](#)

NetworkTRSPData, 678
AreaOfInterestOverride, 679
NonNetworkedParent, 681
Parent, 679
Position, 679
POSITION_OFFSET, 679
Rotation, 680
Scale, 680
SIZE, 680
TeleportKey, 680
WORDS, 680
NetworkTypeid
 NetworkObject, 453
NetworkTypeidKind
 Fusion, 49
NetworkUnwrap
 NetworkBehaviour, 321
 NetworkObject, 449
NetworkWrap
 NetworkBehaviour, 321
 NetworkObject, 449, 450
Next
 NetworkRNG, 542
 Tick, 857
NextExclusive
 NetworkRNG, 542
NextInt32
 NetworkRNG, 542
NextSingle
 NetworkRNG, 542
NextSingleExclusive
 NetworkRNG, 542
NextUInt32
 NetworkRNG, 543
NoActiveConnections
 Fusion, 52
NONE
 AuthorityMasks, 96
None
 Fusion, 45–48, 52, 57
 Fusion.LagCompensation, 61
 NetworkBehaviourId, 359
 NetworkProjectConfig, 530
 PlayerRef, 690
 SceneRef, 748
 TickTimer, 881
NonNetworkedParent
 NetworkTRSPData, 681
Normal
 LagCompensatedHit, 245
 LagCompensationUtils.ContactData, 259
Normalized
 Fusion, 57
NormalizedPercentage
 Fusion, 57
NormalizedRectAttribute, 681
 AspectRatio, 682
 InvertY, 682
 NormalizedRectAttribute, 681
NoSimulation
 Simulation, 775
NotCulled
 Fusion, 52
NotEqual
 Fusion, 44
NotFound
 Fusion, 48
 TickRate, 867
Notify
 NetConfig, 828
NotifyLocalSimulationNotAllowedToSendRpc
 NetworkBehaviourUtils, 369
NotifyLocalTargetedRpcCulled
 NetworkBehaviourUtils, 370
NotifyNetworkUnwrapFailed< T >
 NetworkBehaviourUtils, 370
NotifyNetworkWrapFailed< T >
 NetworkBehaviourUtils, 370, 371
NotifyRpcPayloadSizeExceeded
 NetworkBehaviourUtils, 371
NotifyRpcTargetUnreachable
 NetworkBehaviourUtils, 371
NotInvokableDuringResim
 Fusion, 52
NotInvokableLocally
 Fusion, 52
NotSentBroadcastNoActiveConnections
 Fusion, 53
NotSentBroadcastNoConfirmedNorInterestedClients
 Fusion, 53
NotSentTargetClientNotAvailable
 Fusion, 53
NotSentTargetObjectNotConfirmed
 Fusion, 53
NotSentTargetObjectNotInPlayerInterest
 Fusion, 53
NotZero
 Fusion, 44
Null
 Ptr, 697
NullChecksForNetworkedProperties
 NetworkProjectConfig, 534
Object
 FusionGlobalScriptableObjectLoadResult, 154
 NetworkBehaviourId, 359
 NetworkObjectReleaseContext, 485
 NetworkSpawnOp, 642
 RpcHeader, 732
 SimulationBehaviour, 788
ObjectAcquired
 NetworkRunner, 612
ObjectCount
 Simulation, 780
ObjectDataConsistency
 SimulationConfig, 793
ObjectDelegate

NetworkRunner, 576
 ObjectId
 NetworkSceneObjectId, 635
 ObjectInitializer
 StartGameArgs, 835
 ObjectProvider
 NetworkRunner, 610
 StartGameArgs, 836
 Objects
 Simulation, 780
 ObjectsAllocMemoryFreeInBytes
 FusionStatisticsSnapshot, 845
 ObjectsAllocMemoryUsedInBytes
 FusionStatisticsSnapshot, 845
 ObjectStatisticsManager
 FusionStatisticsManager, 842
 ObjectType
 FusionGlobalScriptableObjectSourceAttribute, 156
Offset
 ColliderDrawInfo, 255
 Hitbox, 162
 HitboxRoot, 182
 SimulationMessage, 813
offset
 NetworkBehaviour, 325
Ok
 Fusion, 55
 StartGameResult, 839
 TickRate, 867
OnBeforeSpawned
 NetworkRunner, 576
OnChangedRenderAttribute, 682
 MethodName, 683
 OnChangedRenderAttribute, 683
OnCompleted
 NetworkSceneAsyncOp.Awaiter, 626
 NetworkSpawnOp.Awaiter, 644
OnConnectedToServer
 INetworkRunnerCallbacks, 209
OnConnectFailed
 INetworkRunnerCallbacks, 209
OnConnectRequest
 INetworkRunnerCallbacks, 209
OnCustomAuthenticationResponse
 INetworkRunnerCallbacks, 210
OnDestroy
 NetworkObject, 450
OnDisable
 FusionGlobalScriptableObject< T >, 150
 NetworkProjectConfigAsset, 537
OnDisconnectedFromServer
 INetworkRunnerCallbacks, 210
OnDrawGizmos
 Hitbox, 161
 HitboxRoot, 180
OnGameStarted
 StartGameArgs, 836
OnHostMigration
 INetworkRunnerCallbacks, 210
OnInput
 INetworkRunnerCallbacks, 210
OnInputMissing
 INetworkRunnerCallbacks, 211
OnLoadedAsGlobal
 FusionGlobalScriptableObject< T >, 150
OnObjectEnterAOI
 INetworkRunnerCallbacks, 211
OnObjectExitAOI
 INetworkRunnerCallbacks, 211
OnPlayerJoined
 INetworkRunnerCallbacks, 212
OnPlayerLeft
 INetworkRunnerCallbacks, 212
OnReliableDataProgress
 INetworkRunnerCallbacks, 212
OnReliableDataReceived
 INetworkRunnerCallbacks, 213
OnSceneInfoChanged
 INetworkSceneManager, 218
OnSceneLoadDone
 INetworkRunnerCallbacks, 213
OnSceneLoadStart
 INetworkRunnerCallbacks, 213
OnSessionListUpdated
 INetworkRunnerCallbacks, 213
OnShutdown
 INetworkRunnerCallbacks, 214
OnUnloadedAsGlobal
 FusionGlobalScriptableObject< T >, 150
OnUserSimulationMessage
 INetworkRunnerCallbacks, 214
OpenInternet
 Fusion.Sockets.Stun, 64
OperationCanceled
 Fusion, 55
OperationExpireTime
 NetConfig, 829
OperationTimeout
 Fusion, 55
operator Angle
 Angle, 85, 86
operator bool
 NetworkBehaviourBuffer, 338
 NetworkBehaviourBufferInterpolator, 350
 NetworkBool, 380
 NetworkId, 417
 Ptr, 694
 SessionInfo, 763
 Tick, 857
operator double
 Angle, 86
operator float
 Angle, 86
 FloatCompressed, 145
operator FloatCompressed
 FloatCompressed, 146

operator FusionGlobalScriptableObjectLoadResult
 FusionGlobalScriptableObjectLoadResult, 154

operator Guid
 NetworkObjectGuid, 463
 NetworkPrefabRef, 514

operator int
 Tick, 858

operator LagCompensatedHit
 LagCompensatedHit, 243, 244

operator long
 Mask256, 282

operator Mask256
 FieldsMask< T >, 132
 Mask256, 283

operator NetworkArray< T >
 NetworkBehaviourUtils.ArrayInitializer< T >, 375

operator NetworkArrayReadOnly< T >
 NetworkArray< T >, 298

operator NetworkBehaviourId
 NetworkBehaviour, 321

operator NetworkBool
 NetworkBool, 380

operator NetworkDictionary< K, V >
 NetworkBehaviourUtils.DictionaryInitializer< K, V >, 376

operator NetworkDictionaryReadOnly< K, V >
 NetworkDictionary< K, V >, 399

operator NetworkId
 NetworkObject, 450

operator NetworkLinkedList< T >
 NetworkBehaviourUtils.ArrayInitializer< T >, 375

operator NetworkObjectGuid
 NetworkObjectGuid, 464
 NetworkPrefabRef, 514

operator NetworkObjectTypeId
 NetworkObjectTypeId, 493

operator NetworkPrefabRef
 NetworkObjectGuid, 464
 NetworkPrefabRef, 514

operator NetworkSceneInfo
 NetworkSceneInfo, 629

operator NetworkString< TSize >
 NetworkString< TSize >, 663

operator Quaternion
 QuaternionCompressed, 701

operator QuaternionCompressed
 QuaternionCompressed, 701

operator SerializableType
 SerializableType< BaseType >, 758

operator string
 NetworkString< TSize >, 664

operator Tick
 Tick, 858

operator Type
 SerializableType< BaseType >, 759

operator Vector2
 Vector2Compressed, 896
 Vector3Compressed, 901

operator Vector2Compressed
 Vector2Compressed, 897

operator Vector3
 Vector3Compressed, 901

operator Vector3Compressed
 Vector3Compressed, 902

operator Vector4
 Vector4Compressed, 906

operator Vector4Compressed
 Vector4Compressed, 907

operator!=
 Angle, 87
 FloatCompressed, 146
 NetworkBehaviourId, 358
 NetworkId, 417
 NetworkLoadSceneParameters, 440
 NetworkObjectGuid, 465
 NetworkObjectHeader, 472
 NetworkObjectType, 493
 NetworkPrefabId, 505
 NetworkPrefabRef, 515
 NetworkRunnerUpdaterDefaultInvokeSettings, 618
 NetworkString< TSize >, 664, 665
 PlayerRef, 686
 Ptr, 695
 QuaternionCompressed, 702
 SceneRef, 746
 Tick, 858
 Vector2Compressed, 897
 Vector3Compressed, 902
 Vector4Compressed, 907

operator<
 Angle, 88
 Tick, 859

operator<=

operator>
 Angle, 89
 Tick, 859

operator>=

operator~
 Mask256, 283

operator+
 Angle, 87
 Ptr, 695

operator-
 Angle, 87
 Ptr, 695

operator==
 Angle, 89
 FloatCompressed, 147
 NetworkBehaviourId, 358
 NetworkId, 417
 NetworkLoadSceneParameters, 440
 NetworkObjectGuid, 465

NetworkObjectHeader, 472
 NetworkObjectTypeid, 493
 NetworkPrefabId, 505
 NetworkPrefabRef, 515
 NetworkRunnerUpdaterDefaultInvokeSettings, 619
 NetworkString< TSize >, 665, 666
 PlayerRef, 687
 Ptr, 696
 QuaternionCompressed, 702
 SceneRef, 746
 Tick, 859
 Vector2Compressed, 897
 Vector3Compressed, 903
 Vector4Compressed, 907
 operator&
 Mask256, 283
 operator |
 Mask256, 283
 Optimize
 LagCompensationSettings, 278
 Options
 NetworkPrefabTable, 526
 Query, 263
 QueryParams, 266
 Order
 FusionGlobalScriptableObjectSourceAttribute, 156
 order
 UnityContextMenuAttribute, 885
 UnityDelayedAttribute, 886
 UnityHeaderAttribute, 887
 UnityMinAttribute, 888
 UnityMultilineAttribute, 889
 UnityNonReorderableAttribute, 889
 UnityRangeAttribute, 891
 UnitySpaceAttribute, 893
 UnityTooltipAttribute, 894
 Origin
 RaycastQuery, 270
 RaycastQueryParams, 271
 OutBandwidth
 FusionStatisticsSnapshot, 845
 NetworkObjectStatisticsSnapshot, 854
 OutObjectUpdates
 FusionStatisticsSnapshot, 845
 OutPackets
 FusionStatisticsSnapshot, 846
 NetworkObjectStatisticsSnapshot, 854
 OverlapBox
 HitboxManager, 166–168
 OverlapSphere
 HitboxManager, 168–170
 Packets
 Fusion, 57
 PacketSize
 NetConfig, 829
 PacketSizeInBits
 NetConfig, 830
 PageCount
 HeapConfiguration, 158
 PageShift
 HeapConfiguration, 158
 PageSizes
 Fusion, 50
 Parent
 NetworkTRSPData, 679
 Parse
 NetworkObjectGuid, 465
 NetworkPrefabRef, 515
 PayloadSizeExceeded
 Fusion, 52
 Peek
 NetworkRNG, 545
 PeerMode
 NetworkProjectConfig, 535
 PeerModes
 NetworkProjectConfig, 529
 Pending
 TickAccumulator, 865
 Penetration
 LagCompensationUtils.ContactData, 259
 Percentage
 Fusion, 57
 PerSecond
 Fusion, 57
 Photon
 Fusion, 54
 PhotonCloudTimeout
 Fusion, 55
 PhysX
 Fusion.LagCompensation, 61
 Player
 Fusion, 50
 Query, 264
 QueryParams, 266
 SimulationInput, 796
 PLAYER_DATA_WORD
 NetworkObjectHeader, 474
 PlayerCount
 SessionInfo, 765
 SimulationConfig, 793
 StartGameArgs, 836
 PlayerData
 NetworkObjectTypeid, 497
 PlayerId
 PlayerRef, 690
 PlayerJoined
 IPlayerJoined, 237
 PlayerLeft
 IPlayerLeft, 238
 PlayerMaxCount
 SimulationRuntimeConfig, 815
 PlayerRef, 683
 AsIndex, 689
 Comparer, 689
 Equals, 685
 FromEncoded, 685

FromIndex, 686
GetHashCode, 686
IsMasterClient, 689
IsNone, 689
IsRealPlayer, 690
MASTER_CLIENT_RAW, 688
MasterClient, 690
None, 690
operator!=, 686
operator==, 687
PlayerId, 690
RawEncoded, 690
Read, 687
SIZE, 689
ToString, 687
Write, 688
Write< T >, 688
PlayMode
 Fusion, 45
Point
 LagCompensatedHit, 245
 LagCompensationUtils.ContactData, 259
PosBlendEnd
 InterpolatedErrorCorrectionSettings, 235
PosBlendStart
 InterpolatedErrorCorrectionSettings, 235
Position
 Hitbox, 163
 NetworkTRSPData, 679
POSITION_OFFSET
 NetworkTRSPData, 679
PositionRotation
 HitboxManager, 171
PositionRotationQueryParams, 259
 Hitbox, 260
 PositionRotationQueryParams, 260
 QueryParams, 260
PosMinCorrection
 InterpolatedErrorCorrectionSettings, 235
PosTeleportDistance
 InterpolatedErrorCorrectionSettings, 236
Prefab
 Fusion, 50
 NetworkPrefabInfo, 509
PrefabId
 NetworkPrefabAcquireContext, 501
PrefabOptions
 NetworkProjectConfigAsset, 538
Prefabs
 NetworkPrefabTable, 526
 NetworkProjectConfigAsset, 538
 NetworkRunner, 610
PrefabTable
 NetworkProjectConfig, 535
PreProcessingDelegate
 Fusion.LagCompensation, 61
Query, 264
 QueryParams, 266
PreserveInPluginAttribute, 691
 PreserveInPluginAttribute, 691
Priority
 EditorButtonAttribute, 122
PriorityCallback
 NetworkObject, 453
PriorityLevel
 Fusion, 50
PriorityLevelDelegate
 NetworkObject, 450
ProjectConfig
 Simulation, 780
Properties
 SessionInfo, 765
PropertyName
 DefaultForPropertyAttribute, 102
PropertyReader
 NetworkBehaviour.PropertyReader< T >, 336
ProvideInput
 NetworkRunner, 610
Proxies
 Fusion, 53
PROXY
 AuthorityMasks, 96
Ptr, 692
 Address, 697
 Equals, 693, 694
 GetHashCode, 694
 NetworkObjectHeaderPtr, 476
 Null, 697
 operator bool, 694
 operator!=, 695
 operator+, 695
 operator-, 695
 operator==, 696
 SIZE, 697
 ToString, 696
Ptr.EqualityComparer, 697
 Equals, 698
 GetHashCode, 699
PushHostMigrationSnapshot
 NetworkRunner, 577
Quaternion
 NetworkBehaviourBufferInterpolator, 351
QuaternionCompressed, 699
 Equals, 700
 GetHashCode, 701
 operator Quaternion, 701
 operator QuaternionCompressed, 701
 operator!=, 702
 operator==, 702
 W, 703
 wEncoded, 703
 X, 704
 xEncoded, 703
 Y, 704
 yEncoded, 703
 Z, 704

zEncoded, 703
Query, 261
 Alpha, 263
 Check, 263
 LayerMask, 263
 Options, 263
 Player, 264
 PreProcessingDelegate, 264
 Query, 262
 Tick, 264
 TickTo, 264
 TriggerInteraction, 264
 UserArgs, 264
QueryParams, 265
 Alpha, 265
 BoxOverlapQueryParams, 251
 LayerMask, 265
 Options, 266
 Player, 266
 PositionRotationQueryParams, 260
 PreProcessingDelegate, 266
 RaycastQueryParams, 272
 SphereOverlapQueryParams, 276
 Tick, 266
 TickTo, 266
 TriggerInteraction, 266
 UserArgs, 267
Queued
 Fusion, 49
Radians
 Fusion, 57
RadiansPerSecond
 Fusion, 57
Radius
 ColliderDrawInfo, 255
 SphereOverlapQuery, 275
 SphereOverlapQueryParams, 276
RangeExAttribute, 704
 ClampMax, 705
 ClampMin, 705
 Max, 706
 Min, 706
 RangeExAttribute, 705
 UseSlider, 705
RangeExclusive
 NetworkRNG, 543
RangeInclusive
 NetworkRNG, 543, 544
Raw
 NetworkId, 419
 Tick, 860
RawEncoded
 PlayerRef, 690
RawGuidValue
 NetworkObjectGuid, 467
 NetworkPrefabRef, 517
RawValue
 NetworkPrefabId, 506
 SceneRef, 747
Raycast
 HitboxManager, 172, 173
RaycastAll
 HitboxManager, 174–176
RaycastAllQuery, 267
 RaycastAllQuery, 267, 268
RaycastQuery, 268
 Check, 269
 Direction, 270
 Length, 270
 Origin, 270
 RaycastQuery, 269
RaycastQueryParams, 270
 Direction, 271
 Length, 271
 Origin, 271
 QueryParams, 272
 RaycastQueryParams, 271
 StaticHitsCapacity, 272
Read
 IElementReaderWriter< T >, 196
 IUnitySurrogate, 222
 NetworkBehaviour.ArrayReader< T >, 327
 NetworkBehaviour.BehaviourReader< T >, 328
 NetworkBehaviour.DictionaryReader< K, V >, 334
 NetworkBehaviour.LinkedListReader< T >, 335
 NetworkBehaviour.PropertyReader< T >, 337
 NetworkBehaviourBuffer, 339, 341
 NetworkId, 418
 PlayerRef, 687
 RpcHeader, 730
 UnityArraySurrogate< T, ReaderWriter >, 225
 UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >, 227
 UnityLinkedListSurrogate< T, ReaderWriter >, 229
 UnitySurrogateBase, 231
 UnityValueSurrogate< T, TReaderWriter >, 233
Read< T >
 NetworkBehaviourBuffer, 341, 343
ReadBoolean
 ReadWriteUtilsForWeaver, 715
ReadFloat
 ReadWriteUtils, 708
ReadInt
 SimulationMessage, 807
ReadNetworkBehaviourRef
 ReadWriteUtils, 708
ReadNetworkedObjectRef
 SimulationMessage, 807
ReadOnly
 Fusion, 45
ReadOnlyAttribute, 706
 InEditMode, 707
 InPlayMode, 707
ReadQuaternion
 ReadWriteUtils, 709

ReadRef
 IElementReaderWriter< T >, 196

ReadSize
 RpcHeader, 730

ReadStringUtf32NoHash
 ReadWriteUtilsForWeaver, 715

ReadStringUtf32WithHash
 ReadWriteUtilsForWeaver, 716

ReadStringUtf8NoHash
 ReadWriteUtilsForWeaver, 716

ReadVector2
 ReadWriteUtils, 709

ReadVector3
 ReadWriteUtils, 709
 SimulationMessage, 807

ReadVector4
 ReadWriteUtils, 710

ReadWriteUtils, 707
 ACCURACY, 713
 ReadFloat, 708
 ReadNetworkBehaviourRef, 708
 ReadQuaternion, 709
 ReadVector2, 709
 ReadVector3, 709
 ReadVector4, 710
 WriteEmptyNetworkBehaviourRef, 710
 WriteFloat, 710
 WriteNetworkBehaviourRef, 711
 WriteNullBehaviourRef, 711
 WriteQuaternion, 711
 WriteVector2, 712
 WriteVector3, 712
 WriteVector4, 712

ReadWriteUtilsForWeaver, 713
 GetByteArrayHashCode, 714
 GetByteCountUtf8NoHash, 714
 GetStringHashCode, 714
 GetWordCountString, 715
 ReadBoolean, 715
 ReadStringUtf32NoHash, 715
 ReadStringUtf32WithHash, 716
 ReadStringUtf8NoHash, 716
 VerifyRawNetworkUnwrap< T >, 717
 VerifyRawNetworkWrap< T >, 717
 WriteBoolean, 718
 WriteStringUtf32NoHash, 718
 WriteStringUtf32WithHash, 718
 WriteStringUtf8NoHash, 719

Redundancy
 SimulationConfig, 792

RedundancyUncompressed
 SimulationConfig, 792

RedundantInputs
 TimeSyncConfiguration, 882

RedundantSnapshots
 TimeSyncConfiguration, 882

ReferenceCountAdd
 SimulationMessage, 808

ReferenceCountSub
 SimulationMessage, 808

ReferencePlayerLoopSystem
 NetworkRunnerUpdaterDefaultInvokeSettings, 620

References
 SimulationMessage, 813

RefitBVHTime
 LagCompensationStatisticsSnapshot, 849

ReflectionUtils, 719
 GetAllNetworkBehaviourTypes, 720
 GetAllSimulationBehaviourTypes, 720
 GetAllWeavedAssemblies, 720
 GetAllWeavedNetworkBehaviourTypes, 720
 GetAllWeavedSimulationBehaviourTypes, 721
 GetAllWeaverGeneratedTypes, 721
 GetCustomAttributeOrThrow< T >, 721
 GetWeavedAttributeOrThrow, 722

Refuse
 NetworkRunnerCallbackArgs.ConnectRequest, 613

Region
 LobbyInfo, 279
 SessionInfo, 765

RegisterInPlayerLoop
 NetworkRunnerUpdaterDefault, 615

RegisterMetaData
 NetworkBehaviourUtils, 372

RegisterRpcInvokeDelegates
 NetworkBehaviourUtils, 372

RegisterSceneObjects
 NetworkRunner, 577

ReinitializeHitboxesBeforeRegistration
 HitboxRoot, 179

ReinterpretState< T >
 NetworkBehaviour, 323
 NetworkBehaviourBuffer, 343

Relayed
 Fusion, 45

Release
 INetworkAssetSource< T >, 203
 NetworkRunner, 557

ReleaseInstance
 INetworkObjectProvider, 207

ReleaseStateAuthority
 NetworkObject, 451

Reliable
 Fusion, 51

ReliableDataTransferModes
 NetworkConfiguration, 393

ReliableDataTransfers
 NetworkConfiguration, 392

Remainder
 TickAccumulator, 865

RemainingTicks
 TickTimer, 879

RemainingTime
 TickTimer, 880

Remap

NetworkLinkedList< T >, 431
Remote
 Fusion, 48, 51, 54
RemoteAddress
 NetworkRunnerCallbackArgs.ConnectRequest, 614
RemoteAlpha
 Simulation, 780
RemotePrefabCreated
 IRemotePrefabCreated, 238
RemoteRenderTime
 NetworkRunner, 610
RemoteTick
 Simulation, 780
RemoteTickPrevious
 Simulation, 781
Remove
 NetBitBufferList, 822
 NetworkDictionary< K, V >, 400
 NetworkLinkedList< T >, 432
 SerializableDictionary< TKey, TValue >, 752
 SimulationInput.Buffer, 800
RemoveCallbacks
 NetworkRunner, 577
RemoveGlobal
 NetworkRunner, 578
RemoveHead
 NetBitBufferList, 822
RemoveInputAuthority
 NetworkObject, 451
RemoveInstance
 NetworkPrefabTable, 524
RemoveSceneRef
 NetworkSceneInfo, 630
Render
 NetworkTRSP, 676
 SimulationBehaviour, 787
RenderAttribute, 722
 Method, 723
 RenderAttribute, 723
 Source, 724
 Timeframe, 724
RenderExecutionCount
 BehaviourStatisticsSnapshot, 841
RenderExecutionTime
 BehaviourStatisticsSnapshot, 841
RenderInternal
 NetworkRunner, 578
RenderSettings
 NetworkRunnerUpdaterDefault, 616
RenderSource
 Fusion, 50
 NetworkObject, 455
RenderTime
 NetworkObject, 455
RenderTimeframe
 Fusion, 51
 NetworkObject, 456
RenderTimeline, 724
 GetRenderBuffers, 724
RenderWeavedAttribute, 725
 RenderWeavedAttribute, 725
REPLICATE_WORD_ALIGN
 Allocator, 76
REPLICATE_WORD_SHIFT
 Allocator, 77
REPLICATE_WORD_SIZE
 Allocator, 77
ReplicateTo
 NetworkBehaviour, 323, 324
 NetworkObject, 453
ReplicateToAll
 NetworkBehaviour, 324
ReplicateToDelegate
 NetworkObject, 451
ReplicationFeatures
 NetworkProjectConfig, 530
 SimulationConfig, 793
RequestStateAuthority
 NetworkObject, 452
RequiresUnsafeCode
 AssemblyNameAttribute, 92
Reset
 FixedArray< T >.Enumerator, 141
 NetworkArray< T >.Enumerator, 301
 NetworkBehaviour.ChangeDetector.Enumerator, 333
 NetworkDictionary< K, V >.Enumerator, 403
 NetworkLinkedList< T >.Enumerator, 434
 SerializableDictionary< TKey, TValue >, 753
ResetState
 NetworkBehaviour, 324
Resimulate
 Fusion, 56
Resimulations
 FusionStatisticsSnapshot, 846
Resolve
 TickRate, 870
ResolveAOIOVERRIDE
 NetworkTRSP, 676
ResolveNetworkPrefabSourceAttribute, 725
ResourceType
 UnityResourcePathAttribute, 892
Result
 RpcSendResult, 739
Retry
 Fusion, 47
Root
 Hitbox, 162
RootGameObjects
 SceneLoadDoneArgs, 741
Rotation
 BoxOverlapQuery, 250
 BoxOverlapQueryParams, 252
 NetworkTRSPData, 680
RotBlendEnd

InterpolatedErrorCorrectionSettings, 236
RotBlendStart
 InterpolatedErrorCorrectionSettings, 236
RotTeleportRadians
 InterpolatedErrorCorrectionSettings, 236
RoundTripTime
 FusionStatisticsSnapshot, 846
RpcAttribute, 726
 Channel, 727
 HostMode, 727
 InvokeLocal, 728
 MaxPayloadSize, 727
 RpcAttribute, 727
 Sources, 728
 Targets, 728
 TickAligned, 728
RpcChannel
 Fusion, 51
RpcHeader, 728
 Behaviour, 731
 Create, 729, 730
 Method, 732
 Object, 732
 Read, 730
 ReadSize, 730
 SIZE, 732
 ToString, 731
 Write, 731
RpcHostMode
 Fusion, 51
RpclInfo, 732
 Channel, 734
 FromLocal, 733
 FromMessage, 733
 IsInvokeLocal, 734
 Source, 734
 Tick, 734
 ToString, 734
RpclInvokeData, 735
 Delegate, 736
 Key, 736
 Sources, 736
 Targets, 736
 ToString, 735
RpclInvokeDelegate
 Fusion, 58
RpclInvokeInfo, 736
 LocallInvokeResult, 737
 SendCullResult, 737
 SendResult, 737
 ToString, 737
RpclLocallInvokeResult
 Fusion, 51
RpclSendCullResult
 Fusion, 52
RpclSendMessageResult
 Fusion, 52
RpclSendResult, 738
 MessageSize, 738
 Result, 739
 ToString, 738
RpcSources
 Fusion, 53
RpcStaticInvokeDelegate
 Fusion, 58
RpcTargetAttribute, 739
 RpcTargetAttribute, 739
RpcTargets
 Fusion, 53
RpcTargetStatus
 Fusion, 54
Run
 TaskManager, 94
Runner
 NetworkObject, 456
 NetworkSpawnOp, 642
 SimulationBehaviour, 788
Running
 NetworkRunner, 557
 TickAccumulator, 865
SampleWindowSeconds
 TimeSyncConfiguration, 882
Scale
 NetworkTRSPData, 680
Scene
 NetworkSceneObjectId, 636
 SceneLoadDoneArgs, 741
 StartGameArgs, 836
SceneCount
 NetworkSceneInfo, 631
SceneCountMask
 Fusion, 49
SceneLoadDone
 ISceneLoadDone, 239
SceneLoadDoneArgs, 740
 RootGameObjects, 741
 Scene, 741
 SceneLoadDoneArgs, 740
 SceneObjects, 741
 SceneRef, 741
SceneLoadId
 NetworkSceneObjectId, 636
SceneLoadStart
 ISceneLoadStart, 240
SceneManager
 NetworkRunner, 610
 StartGameArgs, 836
SceneObject
 Fusion, 50
SceneObjects
 SceneLoadDoneArgs, 741
SceneParams
 NetworkSceneInfo, 631
ScenePathAttribute, 741
SceneRef, 742
 AsIndex, 748

AsPathHash, 748
 Equals, 743
 FLAG_ADDRESSABLE, 747
 FromIndex, 744
 FromPath, 744
 FromRaw, 745
 GetHashCode, 745
 IsIndex, 748
 IsPath, 745
 IsValid, 748
 NetworkSceneAsyncOp, 624
 None, 748
 operator!=, 746
 operator==, 746
 RawValue, 747
 SceneLoadDoneArgs, 741
 SIZE, 747
 ToString, 746, 747
Scenes
 NetworkSceneInfo, 631
Scheduling
 NetworkProjectConfig, 530
SchedulingAndInterestManagement
 NetworkProjectConfig, 530
SchedulingEnabled
 SimulationConfig, 794
SchedulingWithoutAOI
 SimulationConfig, 794
ScriptHeaderBackColor
 Fusion, 54
ScriptHeaderIcon
 Fusion, 54
ScriptHeaderStyle
 Fusion, 54
ScriptHelpAttribute, 749
 BackColor, 749
 Hide, 749
 Style, 749
 Url, 749
Seconds
 Fusion, 57
Select< T >
 NetworkBehaviourBufferInterpolator, 351, 352
Self
 Fusion, 54
SendCullResult
 RpcInvokeInfo, 737
SendDelta
 Simulation, 781
SendRate
 Simulation, 781
SendReliableDataToPlayer
 NetworkRunner, 578
SendReliableDataToServer
 NetworkRunner, 578
SendResult
 RpcInvokeInfo, 737
SendRpc

NetworkRunner, 579
Sent
 SimulationInput, 796
SentBroadcast
 Fusion, 53
SentToServerForForwarding
 Fusion, 53
SentToTargetClient
 Fusion, 53
SerializableDictionary< TKey, TValue >, 750
 Add, 751
 Clear, 751
 ContainsKey, 752
 Count, 754
 Create< TKey, TValue >, 752
 EntryKeyPropertyName, 754
 GetEnumerator, 752
 IsReadOnly, 755
 ItemsPropertyName, 754
 Keys, 755
 Remove, 752
 Reset, 753
 Store, 753
 this[TKey key], 755
 TryGetValue, 753
 Values, 755
 Wrap, 754
SerializableType
 SerializableType< BaseType >, 757
SerializableType< BaseType >, 756
 AssemblyQualifiedName, 759
 AsShort, 758
 Equals, 758
 GetHashCode, 758
 GetShortAssemblyQualifiedName, 758
 IsValid, 759
 operator SerializableType, 758
 operator Type, 759
 SerializableType, 757
 Value, 759
SerializableTypeAttribute, 760
 BaseType, 760
 UseFullAssemblyQualifiedName, 760
 WarnIfNoPreserveAttribute, 760
Serialize
 NetworkProjectConfig, 531
SerializeReferenceTypePickerAttribute, 761
 GroupTypesByNamespace, 762
 SerializeReferenceTypePickerAttribute, 761
 ShowFullName, 762
 Types, 762
Server
 Fusion, 46, 56
 TickRate.Resolved, 874
ServerAlreadyInRoom
 Fusion.Protocol, 62
ServerFull
 Fusion.Sockets, 64

ServerIndex
 TickRate.Selection, 876

ServerIndexOutOfRange
 TickRate, 867

ServerInRoom
 Fusion, 55

ServerLogic
 Fusion.Protocol, 62

ServerMode
 SimulationRuntimeConfig, 815

ServerRefused
 Fusion.Sockets, 64

ServerSend
 TickRate.Resolved, 874

ServerSendDelta
 TickRate.Resolved, 875

ServerSendIndex
 TickRate.Selection, 877

ServerSendIndexOutOfRange
 TickRate, 867

ServerSendRateLargerThanTickRate
 TickRate, 867

ServerTickDelta
 TickRate.Resolved, 875

ServerTickStride
 TickRate.Resolved, 875

Service
 TaskManager, 94

SessionInfo, 762
 IsOpen, 764
 IsValid, 764
 IsVisible, 764
 MaxPlayers, 764
 Name, 765
 NetworkRunner, 610
 operator bool, 763
 PlayerCount, 765
 Properties, 765
 Region, 765
 ToString, 763
 UpdateCustomProperties, 764

SessionLobby
 Fusion, 54

SessionName
 StartGameArgs, 836

SessionNameGenerator
 StartGameArgs, 837

SessionProperties
 StartGameArgs, 837

Set
 NetworkArray< T >, 298
 NetworkButtons, 386
 NetworkDictionary< K, V >, 401
 NetworkLinkedList< T >, 432
 NetworkString< TSize >, 666

Set< T >
 NetworkButtons, 386
 NetworkInput, 423

SetAllDown
 NetworkButtons, 387

SetAllUp
 NetworkButtons, 387

SetAreaOfInterestCellSize
 NetworkRunner, 579

SetAreaOfInterestGrid
 NetworkRunner, 581

SetAreaOfInterestOverride
 NetworkTransform, 673
 NetworkTRSP, 677

SetBehaviourReplicateTo
 NetworkRunner, 581

SetBehaviourReplicateToAll
 NetworkRunner, 581

SetBit
 Mask256, 283

SetCurrentVersion
 NetworkObjectFlagsExtensions, 458

SetDown
 NetworkButtons, 387

SetDown< T >
 NetworkButtons, 387

SetDummy
 SimulationMessage, 808

SetForcedAlive< T >
 DynamicHeap, 113

SetHitboxActive
 HitboxRoot, 180

SetIgnored
 NetworkObjectFlagsExtensions, 458

SetIsSimulated
 NetworkRunner, 582

SetLayer
 Hitbox, 161

SetMasterClient
 NetworkRunner, 582

SetMinBoundingRadius
 HitboxRoot, 181

SetNotTickAligned
 SimulationMessage, 808

SetParentTransform
 NetworkTRSP, 677

SetPlayerAlwaysInterested
 NetworkObject, 452
 NetworkRunner, 582

SetPlayerObject
 NetworkRunner, 583

SetSimulateMultiPeerPhysics
 NetworkRunner, 583

SetStatic
 SimulationMessage, 808

SetTarget
 SimulationMessage, 808

SetTrigger
 NetworkMecanimAnimator, 443, 444

SetUnreliable
 SimulationMessage, 809

SetUp
 NetworkButtons, 388
Setup
 IDataEncryption, 126
 TaskManager, 95
SetUp< T >
 NetworkButtons, 388
Shared
 Fusion, 46, 55, 56
SharedModeStateAuthLocalPlayer
 Fusion, 49
SharedModeStateAuthMasterClient
 Fusion, 49
ShouldRegisterRpcInvokeDelegates
 NetworkBehaviourUtils, 372
ShowFlagsButtons
 ExpandableEnumAttribute, 128
ShowFullName
 SerializeReferenceTypePickerAttribute, 762
ShowInlineHelp
 ExpandableEnumAttribute, 128
ShowTypeHelp
 InlineHelpAttribute, 221
Shutdown
 INetworkRunnerUpdater, 215
 INetworkSceneManager, 218
 NetworkRunner, 557, 583
ShutdownReason
 Fusion, 55
 StartGameResult, 839
Simulation, 765
 ActivePlayers, 776
 AfterSimulation, 769
 AfterUpdate, 769
 BeforeFirstTick, 769
 BeforeSimulation, 769
 BeforeUpdate, 769
 Config, 776
 DeltaTime, 776
 GetAreaOfInterestGizmoData, 769
 GetInputAuthority, 770
 GetInputForPlayer, 770
 GetObjectsAndPlayersInAreaOfInterestCell, 770
 GetObjectsInAreaOfInterestForPlayer, 770
 GetStateAuthority, 771
 HasAnyActiveConnections, 771
 InputCount, 776
 IsClient, 777
 IsFirstTick, 777
 IsForward, 777
 IsInputAuthority, 771
 IsInterestedIn, 772
 IsLastTick, 777
 IsLocalPlayerFirstExecution, 777
 IsLocalSimulationInputAuthority, 772
 IsLocalSimulationStateAuthority, 772, 773
 IsLocalSimulationStateOrInputSource, 773
 IsMasterClient, 778
 IsPlayer, 778
 IsResimulation, 778
 IsRunning, 778
 IsServer, 778
 IsShutdown, 778
 IsSinglePlayer, 779
 IsStateAuthority, 774
 LatestServerTick, 779
 LocalAddress, 779
 LocalAlpha, 779
 LocalPlayer, 779
 Mode, 779
 NetConfig, 829
 NetConfigPointer, 780
 NetworkConnected, 774
 NetworkDisconnected, 775
 NetworkProjectConfig, 535
 NetworkReceiveDone, 775
 NoSimulation, 775
 ObjectCount, 780
 Objects, 780
 ProjectConfig, 780
 RemoteAlpha, 780
 RemoteTick, 780
 RemoteTickPrevious, 781
 SendDelta, 781
 SendRate, 781
 Stage, 781
 Tick, 781
 TickDeltaDouble, 781
 TickDeltaFloat, 782
 TickPrevious, 782
 TickRate, 782
 TickStride, 782
 Time, 782
 Topology, 782
 TryGetHostPlayer, 775
 Update, 776
Simulation.AreaOfInterest, 783
 CELL_SIZE, 785
 GetCellSize, 783
 SphereToCells, 784
 ToCell, 784
 ToCellCenter, 785
 x, 785
SimulationBehaviour, 786
 CanReceiveRenderCallback, 788
 CanReceiveSimulationCallback, 788
 FixedUpdateNetwork, 787
 Object, 788
 Render, 787
 Runner, 788
SimulationBehaviourAttribute, 788
 Modes, 789
 Stages, 789
 Topologies, 789
SimulationBehaviourListScope, 790
 Dispose, 790

SimulationConfig, 790
AreaOfInterestEnabled, 794
DataConsistency, 791
DeltaTime, 792
Eventual, 792
Full, 792
HostMigration, 792
InputDataWordCount, 792
InputTotalWordCount, 794
InputTransferMode, 793
InputTransferModes, 792
LatestState, 792
ObjectDataConsistency, 793
PlayerCount, 793
Redundancy, 792
RedundancyUncompressed, 792
ReplicationFeatures, 793
SchedulingEnabled, 794
SchedulingWithoutAOI, 794
SimulationTimeMode, 792
SimulationUpdateTimeMode, 793
TickRateSelection, 793
Topology, 794
UnscaledDeltaTime, 792
SimulationEnter
 ISimulationEnter, 240
SimulationExit
 ISimulationExit, 241
SimulationInput, 795
 Clear, 795
 CopyFrom, 796
 Data, 796
 Header, 796
 Player, 796
 Sent, 796
SimulationInput.Buffer, 797
 Add, 798
 Buffer, 797
 Clear, 798
 Contains, 798
 CopySortedTo, 799
 Count, 800
 Full, 800
 Get, 799
 GetInsertTime, 799
 GetLastUsedInputHeader, 800
 Remove, 800
SimulationInputHeader, 801
 InterpAlpha, 801
 InterpFrom, 801
 InterpTo, 802
 SIZE, 802
 Tick, 802
 WORD_COUNT, 802
SimulationMessage, 802
 Allocate, 805
 CanAllocateUserPayload, 805
 Capacity, 810
Clone, 805
FLAG_DUMMY, 810
FLAG_INTERNAL, 810
FLAG_NOT_TICK_ALIGNED, 811
FLAG_REMOTE, 811
FLAG_STATIC, 811
FLAG_TARGET_PLAYER, 811
FLAG_TARGET_SERVER, 811
FLAG_UNRELIABLE, 811
FLAG_USER_FLAGS_START, 812
FLAG_USER_MESSAGE, 812
Flags, 812
FLAGS_RESERVED, 812
FLAGS_RESERVED_BITS, 812
GetData, 806
GetFlag, 806
IsTargeted, 806
IsUnreliable, 814
MAX_PAYLOAD_SIZE, 812
Offset, 813
ReadInt, 807
ReadNetworkedObjectRef, 807
ReadVector3, 807
ReferenceCountAdd, 808
ReferenceCountSub, 808
References, 813
SetDummy, 808
SetNotTickAligned, 808
SetStatic, 808
SetTarget, 808
SetUnreliable, 809
SIZE, 813
Source, 813
Target, 813
Tick, 813
ToString, 809
WriteInt, 809
WriteNetworkedObjectRef, 809
WriteVector3, 810
SimulationMessagePtr, 814
 Message, 814
SimulationModes
 Fusion, 56
SimulationRuntimeConfig, 814
 HostPlayer, 815
 MasterClient, 815
 PlayerMaxCount, 815
 ServerMode, 815
 TickRate, 816
 Topology, 816
SimulationSpeed
 FusionStatisticsSnapshot, 846
SimulationStages
 Fusion, 56
SimulationState
 NetworkBehaviour.ChangeDetector, 330
SimulationTime
 NetworkRunner, 611

SimulationTimeMode
 SimulationConfig, 792

SimulationTimeOffset
 FusionStatisticsSnapshot, 846

SimulationUnityScene
 NetworkRunner, 611

SimulationUpdateTimeMode
 SimulationConfig, 793

Single
 Fusion, 46

SinglePlayerContinue
 NetworkRunner, 584

SinglePlayerPause
 NetworkRunner, 584

SIZE
 _128, 67
 _16, 68
 _2, 69
 _256, 70
 _32, 71
 _4, 72
 _512, 73
 _64, 74
 _8, 74

Allocator.Config, 80

Angle, 90

NetworkBehaviourId, 359

NetworkId, 419

NetworkObjectGuid, 467

NetworkObjectHeader, 474

NetworkObjectNestingKey, 481

NetworkObjectTypeId, 494

NetworkPhysicsInfo, 499

NetworkPrefabId, 506

NetworkPrefabRef, 517

NetworkRNG, 545

NetworkSceneInfo, 630

NetworkTRSPData, 680

PlayerRef, 689

Ptr, 697

RpcHeader, 732

SceneRef, 747

SimulationInputHeader, 802

SimulationMessage, 813

Tick, 860

TickRate.Resolved, 874

SnapshotFrom
 NetworkBehaviour.ChangeDetector, 330

SnapshotHistoryDraw, 272
 GetEnumerator, 272
 LagCompensationDraw, 258

SnapshotTo
 NetworkBehaviour.ChangeDetector, 330

SocketRecvBuffer
 NetConfig, 829

SocketRecvBufferSize
 NetworkConfiguration, 394

SocketSendBuffer
 NetConfig, 829

SocketSendBufferSize
 NetworkConfiguration, 394

SortDistance
 LagCompensatedExt, 256

SortKey
 NetworkObject, 453

SortReference
 LagCompensatedExt, 257

Source
 NetworkBehaviour.ChangeDetector, 330
 RenderAttribute, 724
 RpcInfo, 734
 SimulationMessage, 813

SourcesHostPlayer
 Fusion, 51

SourcesServer
 Fusion, 51

Sources
 NetworkRpcWeavedInvokerAttribute, 547
 RpcAttribute, 728
 RpcInvokeData, 736

Spawn
 NetworkRunner, 584–587

Spawn< T >
 NetworkRunner, 587

SpawnAsync
 NetworkRunner, 588–590

SpawnAsync< T >
 NetworkRunner, 591

Spawned
 Fusion, 49
 ISpawned, 242
 NetworkBehaviour, 324

SpawnedByClient
 Fusion, 47

Sphere
 Fusion, 46

SphereOverlapQuery, 273
 Center, 274
 Check, 274
 Radius, 275
 SphereOverlapQuery, 273, 274

SphereOverlapQueryParams, 275
 Center, 276
 queryParams, 276
 Radius, 276
 SphereOverlapQueryParams, 275
 StaticHitsCapacity, 276

SphereRadius
 Hitbox, 162

SphereToCells
 Simulation.AreaOfInterest, 784

SquareMagnitude
 Fusion, 57

StackTrace
 StartGameResult, 839

Stage
 NetworkRunner, 611
 Simulation, 781
Stages
 SimulationBehaviourAttribute, 789
Start
 TickAccumulator, 864
StartGame
 NetworkRunner, 592
StartGameArgs, 831
 Address, 833
 AuthValues, 833
 Config, 833
 ConnectionToken, 833
 CustomCallbackInterfaces, 833
 CustomLobbyName, 833
 CustomPhotonAppSettings, 834
 CustomPublicAddress, 834
 CustomSTUNServer, 834
 DisableNATPunchthrough, 834
 EnableClientSessionCreation, 834
 GameMode, 834
 HostMigrationResume, 835
 HostMigrationToken, 835
 IsOpen, 835
 IsVisible, 835
 MatchmakingMode, 835
 ObjectInitializer, 835
 ObjectProvider, 836
 OnGameStarted, 836
 PlayerCount, 836
 Scene, 836
 SceneManager, 836
 SessionName, 836
 SessionNameGenerator, 837
 SessionProperties, 837
 StartGameCancellationToken, 837
 ToString, 832
 Updater, 837
 UseCachedRegions, 837
 UseDefaultPhotonCloudPorts, 837
StartGameCancellationToken
 StartGameArgs, 837
StartGameResult, 838
 ErrorMessage, 839
 Ok, 839
 ShutdownReason, 839
 StackTrace, 839
 ToString, 838
Starting
 NetworkRunner, 557
StartNew
 TickAccumulator, 865
StartsWith
 NetworkString< TSize >, 667
StartsWith< TOtherSize >
 NetworkString< TSize >, 667
STATE
 AuthorityMasks, 96
State
 NetworkRunner, 611
 NetworkTRSP, 678
StateAuthority
 Fusion, 53
 NetworkObject, 456
 NetworkObjectHeader, 474
 NetworkObjectMeta, 477
StateAuthorityChanged
 IStateAuthorityChanged, 242
StateBuffer
 NetworkBehaviour, 326
StateBufferIsValid
 NetworkBehaviour, 327
StateReceiveDelta
 FusionStatisticsSnapshot, 846
States
 NetworkRunner, 557
StaticHitsCapacity
 BoxOverlapQueryParams, 252
 RaycastQueryParams, 272
 SphereOverlapQueryParams, 276
Status
 NetworkObjectSpawnException, 488
 NetworkSpawnOp, 643
Stop
 TickAccumulator, 865
Store
 SerializableDictionary< TKey, TValue >, 753
Struct
 Fusion, 48
StructArray
 Fusion, 48
StunServers.StunServer, 830
Style
 ScriptHelpAttribute, 749
Substring
 NetworkString< TSize >, 668
SubtickAccuracy
 Fusion, 46
Success
 Fusion, 47, 48
SurrogateType
 FixedBufferPropertyAttribute, 143
Symmetric
 Fusion.Sockets.Stun, 64
SyncParent
 NetworkTransform, 674
SyncScale
 NetworkTransform, 674

T
 NetworkBehaviour.BehaviourReader< T >, 329
 NetworkBehaviour.PropertyReader< T >, 337
TagetPlayerIsNotLocal
 Fusion, 52
Target
 SimulationMessage, 813

TargetAllocator
 MemoryStatisticsSnapshot, 851

TargetMethod
 FieldEditorButtonAttribute, 130

TargetPlayerIsLocalButRpcIsNotInvokableLocally
 Fusion, 52

TargetPlayerIsNotLocal
 Fusion, 52

TargetPlayerUnreachable
 Fusion, 52

Targets
 NetworkRpcWeavedInvokerAttribute, 548

 RpcAttribute, 728

 RpcInvokeData, 736

TargetTick
 TickTimer, 881

TaskManager, 93
 ContinueWhenAll, 93

 Run, 94

 Service, 94

 Setup, 95

Teleport
 INetworkTRSPTeleport, 220

 NetworkTransform, 674

 NetworkTRSP, 677

TeleportKey
 NetworkTRSPData, 680

this[int i]
 Mask256, 284

this[int index]
 FixedArray< T >, 139

 INetworkArray, 201

 NetworkArray< T >, 299

 NetworkArrayReadOnly< T >, 304

 NetworkBehaviourBuffer, 344

 NetworkLinkedList< T >, 433

 NetworkLinkedListReadOnly< T >, 438

 NetworkString< TSize >, 670

 TickRate, 872

this[K key]
 NetworkDictionary< K, V >, 402

this[TKey key]
 SerializableDictionary< TKey, TValue >, 755

ThrowIfBehaviourNotInitialized
 NetworkBehaviourUtils, 373

Tick, 854
 ALIGNMENT, 860

 CompareTo, 856

 Equals, 856

 GetHashCode, 857

 NetworkBehaviourBuffer, 344

 NetworkRunner, 611

 Next, 857

 operator bool, 857

 operator int, 858

 operator Tick, 858

 operator!=, 858

 operator<, 859

 operator<=, 859

 operator>, 859

 operator>=, 859

 operator==, 859

 Query, 264

 QueryParams, 266

 Raw, 860

 RpcInfo, 734

 Simulation, 781

 SimulationInputHeader, 802

 SimulationMessage, 813

 SIZE, 860

 ToString, 859

 Tick.EqualityComparer, 860

 Equals, 861

 GetHashCode, 861

 Tick.RelationalComparer, 861

 Compare, 862

 TickAccumulator, 862

 AddTicks, 863

 AddTime, 863

 Alpha, 864

 ConsumeTick, 864

 Pending, 865

 Remainder, 865

 Running, 865

 Start, 864

 StartNew, 865

 Stop, 865

 TimeScale, 865

 TickAligned
 RpcAttribute, 728

 TickDeltaDouble
 Simulation, 781

 TickDeltaFloat
 Simulation, 782

 TickPrevious
 Simulation, 782

 TickRate, 866
 Available, 872

 ClampSelection, 868

 Client, 872

 ClientSendIndexOutOfRange, 867

 Count, 872

 Error, 867

 Get, 868

 GetDivisor, 868

 GetTickRate, 869

 Init, 869

 InvalidTickRate, 867

 IsValid, 869, 870

 NetworkRunner, 611

 NotFound, 867

 Ok, 867

 Resolve, 870

 ServerIndexOutOfRange, 867

 ServerSendIndexOutOfRange, 867

 ServerSendRateLargerThanTickRate, 867

Simulation, 782
SimulationRuntimeConfig, 816
this[int index], 872
ToArray, 871
Validate, 871
ValidateResult, 867
ValidateSelection, 871
TickRate.Resolved, 873
Client, 874
ClientSend, 874
ClientSendDelta, 875
ClientTickDelta, 875
ClientTickStride, 875
Server, 874
ServerSend, 874
ServerSendDelta, 875
ServerTickDelta, 875
ServerTickStride, 875
SIZE, 874
WORDS, 874
TickRate.Selection, 876
Client, 876
ClientSendIndex, 876
ServerIndex, 876
ServerSendIndex, 877
TickRateSelection
 SimulationConfig, 793
Ticks
 Fusion, 57
TicksExecuted
 NetworkRunner, 612
TicksPerSecond
 Fusion, 57
TickStride
 Simulation, 782
TickTimer, 877
 CreateFromSeconds, 878
 CreateFromTicks, 878
 Expired, 879
 ExpiredOrNotRunning, 879
 IsRunning, 880
 None, 881
 RemainingTicks, 879
 RemainingTime, 880
 TargetTick, 881
 ToString, 880
TickTo
 Query, 264
 QueryParams, 266
Time
 Simulation, 782
Timeframe
 RenderAttribute, 724
Timeout
 Fusion.Sockets, 64
TimeResets
 FusionStatisticsSnapshot, 847
TimeScale
 NetworkPhysicsInfo, 499
 TickAccumulator, 865
 TimeSyncConfiguration, 881
 MaxLateInputs, 881
 MaxLateSnapshots, 882
 RedundantInputs, 882
 RedundantSnapshots, 882
 SampleWindowSeconds, 882
 TimeSynchronizationOverride
 NetworkProjectConfig, 535
To
 Fusion, 50
 NetworkBehaviourBufferInterpolator, 355
ToArray
 FixedArray< T >, 139
 NetworkArray< T >, 298
 TickRate, 871
ToCell
 Simulation.AreaOfInterest, 784
ToCellCenter
 Simulation.AreaOfInterest, 785
ToggleLeftAttribute, 882
ToListString
 FixedArray< T >, 139
 NetworkArray< T >, 299
ToLower
 NetworkString< TSize >, 669
ToNamePrefixString
 NetworkId, 418
Topologies
 Fusion, 56
 SimulationBehaviourAttribute, 789
Topology
 NetworkRunner, 612
 Simulation, 782
 SimulationConfig, 794
 SimulationRuntimeConfig, 816
ToReadOnly
 NetworkArray< T >, 299
 NetworkDictionary< K, V >, 401
ToString
 Allocator.Config, 79
 Angle, 90
 FixedArray< T >, 139
 HeapConfiguration, 158
 Mask256, 284
 NetworkArray< T >, 299
 NetworkBehaviourId, 358
 NetworkBool, 380
 NetworkId, 418
 NetworkLoadSceneParameters, 441
 NetworkObjectGuid, 465, 466
 NetworkObjectHeader, 472
 NetworkObjectNestingKey, 480
 NetworkObjectReleaseContext, 485
 NetworkObjectTypeid, 493
 NetworkPrefabId, 505, 506
 NetworkPrefabRef, 515, 516

NetworkProjectConfig, 531
 NetworkRNG, 544
 NetworkRunnerUpdaterDefaultInvokeSettings, 619
 NetworkSceneInfo, 630
 NetworkSceneObjectID, 635
 NetworkString< TSize >, 669
 PlayerRef, 687
 Ptr, 696
 RpcHeader, 731
 RpcInfo, 734
 RpcInvokeData, 735
 RpcInvokeInfo, 737
 RpcSendResult, 738
 SceneRef, 746, 747
 SessionInfo, 763
 SimulationMessage, 809
 StartGameArgs, 832
 StartGameResult, 838
 Tick, 859
 TickTimer, 880
 TotalElapsedTime
 LagCompensationStatisticsSnapshot, 849
 TotalFreeBlocks
 MemoryStatisticsSnapshot, 852
 TotalHitboxes
 HitboxManager, 177
 ToUnityGuidString
 NetworkObjectGuid, 466
 NetworkPrefabRef, 516
 ToUpper
 NetworkString< TSize >, 669
 TriggerInteraction
 Query, 264
 QueryParams, 266
 TryAddSource
 NetworkPrefabTable, 524
 TryFindBehaviour
 NetworkRunner, 592
 TryFindBehaviour< T >
 NetworkRunner, 593
 TryFindObject
 NetworkRunner, 593
 TryGet
 NetworkDictionary< K, V >, 401
 NetworkDictionaryReadOnly< K, V >, 406
 TryGet< T >
 NetworkInput, 423
 TryGetBehaviour< T >
 Behaviour, 97
 TryGetBehaviourStatistics
 NetworkRunner, 594
 TryGetFusionStatistics
 NetworkRunner, 594
 TryGetGlobal
 NetworkProjectConfigAsset, 537
 TryGetGlobalInternal
 FusionGlobalScriptableObject< T >, 150
 TryGetHostPlayer

Simulation, 775
 TryGetInputForPlayer< T >
 NetworkRunner, 594
 TryGetNetworkedBehaviourFromNetworkedObjectRef<
 T >
 NetworkRunner, 595
 TryGetNetworkedBehaviourId
 NetworkRunner, 595
 TryGetObjectRefFromNetworkedBehaviour
 NetworkRunner, 596
 TryGetPhysicsInfo
 NetworkRunner, 596
 TryGetPhysicsScene2D
 INetworkSceneManager, 218
 TryGetPhysicsScene3D
 INetworkSceneManager, 219
 TryGetPlayerObject
 NetworkRunner, 596
 TryGetRpcInvokeDelegateArray
 NetworkBehaviourUtils, 373
 TryGetRpcStaticInvokeDelegate
 NetworkBehaviourUtils, 373
 TryGetSceneInfo
 NetworkRunner, 597
 TryGetSnapshotsBuffers
 NetworkBehaviour, 325
 TryGetValue
 SerializableDictionary< TKey, TValue >, 753
 TryParse
 NetworkObjectGuid, 466
 NetworkPrefabRef, 516
 TrySet< T >
 NetworkInput, 423
 TrySetPhysicsInfo
 NetworkRunner, 597
 TrySpawn
 NetworkRunner, 598, 600, 601
 TrySpawn< T >
 NetworkRunner, 602
 Type
 ColliderDrawInfo, 255
 FixedBufferPropertyAttribute, 143
 Hitbox, 162
 LagCompensatedHit, 246
 NetworkInput, 424
 NetworkObjectHeader, 474
 NetworkObjectHeaderPtr, 476
 NetworkObjectMeta, 478
 NetworkProjectConfigAsset.SerializableSimulationBehaviourMeta,
 540
 TypeId
 NetworkObjectReleaseContext, 485
 NetworkObjectSpawnException, 488
 NetworkProjectConfig, 535
 Types
 SerializeReferenceTypePickerAttribute, 762
 UdpBlocked
 Fusion.Sockets.Stun, 64

Unit
 UnitAttribute, 884
UnitAttribute, 883
 Unit, 884
 UnitAttribute, 883
Units
 Fusion, 56, 57
Unity
 Fusion, 54
UnityAddressablesRuntimeKeyAttribute, 884
UnityArraySurrogate< T, ReaderWriter >, 224
 DataProperty, 226
 Init, 224
 Read, 225
 Write, 225
UnityAssetGuidAttribute, 884
UnityContextMenuItemAttribute, 884
 order, 885
 UnityContextMenuItemAttribute, 885
UnityDelayedAttribute, 885
 order, 886
UnityDictionarySurrogate< TKeyType, TKeyReader-
 Writer, TValueType, TValueReaderWriter >, 226
 DataProperty, 228
 Init, 227
 Read, 227
 Write, 227
UnityEngine, 65
UnityEngine.SceneManagement, 65
UnityEngine.Scripting, 65
UnityEngine.Serialization, 65
UnityFormerlySerializedAsAttribute, 886
 UnityFormerlySerializedAsAttribute, 886
UnityHeaderAttribute, 887
 order, 887
 UnityHeaderAttribute, 887
UnityLinkedListSurrogate< T, ReaderWriter >, 228
 DataProperty, 230
 Init, 229
 Read, 229
 Write, 229
UnityMinAttribute, 888
 order, 888
 UnityMinAttribute, 888
UnityMultilineAttribute, 889
 order, 889
UnityNonReorderableAttribute, 889
 order, 889
UnityNonSerializedAttribute, 890
UnityPlayerLoopSystemAddMode
 Fusion, 57
UnityRangeAttribute, 890
 order, 891
 UnityRangeAttribute, 890
UnityResourcePathAttribute, 891
 ResourceType, 892
 UnityResourcePathAttribute, 891
UnitySerializeField, 892
UnitySerializeReference, 892
UnitySpaceAttribute, 892
 order, 893
 UnitySpaceAttribute, 893
UnitySurrogateBase, 230
 Init, 231
 Read, 231
 Write, 231
UnityTooltipAttribute, 893
 order, 894
 UnityTooltipAttribute, 894
UnityValueSurrogate< T, TReaderWriter >, 232
 DataProperty, 234
 Init, 233
 Read, 233
 Write, 233
Unload
 NetworkPrefabTable, 525
UnloadAll
 NetworkPrefabTable, 525
Unloader
 FusionGlobalScriptableObjectLoadResult, 154
UnloadGlobal
 NetworkProjectConfig, 531
 NetworkProjectConfigAsset, 538
UnloadGlobalInternal
 FusionGlobalScriptableObject< T >, 151
UnloadPrefabOnReleasingLastInstance
 NetworkPrefabTableOptions, 527
UnloadScene
 INetworkSceneManager, 219
 NetworkRunner, 602
UnloadUnreferenced
 NetworkPrefabTable, 525
UnloadUnusedPrefabsOnShutdown
 NetworkPrefabTableOptions, 527
Unreachable
 Fusion, 54
UnregisterFromPlayerLoop
 NetworkRunnerUpdaterDefault, 615
Unreliable
 Fusion, 51
UnscaledDeltaTime
 SimulationConfig, 792
Update
 Simulation, 776
UpdateBufferTime
 LagCompensationStatisticsSnapshot, 850
UpdateBVHTime
 LagCompensationStatisticsSnapshot, 850
UpdateCustomProperties
 SessionInfo, 764
UpdateDelay
 HostMigrationConfig, 183
UpdateInternal
 NetworkRunner, 604
Updater

StartGameArgs, 837
 UpdateSettings
 NetworkRunnerUpdaterDefault, 616
 Url
 ScriptHelpAttribute, 749
 UseCachedRegions
 StartGameArgs, 837
 UseDefaultPhotonCloudPorts
 StartGameArgs, 837
 UseFullAssemblyQualifiedName
 SerializableTypeAttribute, 760
 UserArgs
 Query, 264
 QueryParams, 267
 UserId
 NetworkRunner, 612
 UseSerializableDictionary
 NetworkProjectConfig, 535
 UseSlider
 RangeExAttribute, 705
 V1
 Fusion, 47
 Valid
 NetworkBehaviourBuffer, 345
 NetworkBehaviourBufferInterpolator, 355
 Validate
 TickRate, 871
 ValidateResult
 TickRate, 867
 ValidateSelection
 TickRate, 871
 Value
 NetworkObjectNestingKey, 481
 NetworkSceneLoadId, 633
 NetworkString< TSize >, 670
 SerializableType< BaseType >, 759
 valueEncoded
 FloatCompressed, 147
 Values
 SerializableDictionary< TKey, TValue >, 755
 Vector2
 NetworkBehaviourBufferInterpolator, 352, 353
 Vector2Compressed, 894
 Equals, 895, 896
 GetHashCode, 896
 operator Vector2, 896
 operator Vector2Compressed, 897
 operator!=, 897
 operator==, 897
 X, 898
 xEncoded, 898
 Y, 898
 yEncoded, 898
 Vector3
 NetworkBehaviourBufferInterpolator, 353, 354
 Vector3Compressed, 899
 Equals, 900
 GetHashCode, 901
 operator Vector3, 901
 operator Vector3Compressed, 902
 operator!=, 902
 operator==, 903
 X, 904
 xEncoded, 903
 Y, 904
 yEncoded, 903
 Z, 904
 zEncoded, 903
 Vector4
 NetworkBehaviourBufferInterpolator, 354
 Vector4Compressed, 904
 Equals, 905, 906
 GetHashCode, 906
 operator Vector4, 906
 operator Vector4Compressed, 907
 operator!=, 907
 operator==, 907
 W, 909
 wEncoded, 908
 X, 909
 xEncoded, 908
 Y, 909
 yEncoded, 908
 Z, 909
 zEncoded, 908
 VerifyHash
 IDataEncryption, 126
 VerifyRawNetworkUnwrap< T >
 ReadWriteUtilsForWeaver, 717
 VerifyRawNetworkWrap< T >
 ReadWriteUtilsForWeaver, 717
 Version
 NetworkPrefabTable, 526
 NetworkProjectConfig, 536
 NetworkSceneInfo, 631
 Versioning, 691
 GetCurrentVersion, 692
 GetProductVersion, 692
 Visibility
 EditorButtonAttribute, 122
 W
 QuaternionCompressed, 703
 Vector4Compressed, 909
 WaitForResult
 INetworkAssetSource< T >, 203
 Waiting
 NetworkRunnerCallbackArgs.ConnectRequest,
 614
 WarnIfAttribute, 909
 AsBox, 910
 Message, 911
 WarnIfAttribute, 910
 WarnIfNoPreserveAttribute
 SerializableTypeAttribute, 760
 WasPressed

NetworkButtons, 389
WasPressed< T >
 NetworkButtons, 389
WasReleased
 NetworkButtons, 389
WasReleased< T >
 NetworkButtons, 390
WeaverGeneratedAttribute, 911
wEncoded
 QuaternionCompressed, 703
 Vector4Compressed, 908
WORD_COUNT
 NetworkPhysicsInfo, 499
 NetworkSceneInfo, 631
 SimulationInputHeader, 802
WordCount
 DefaultForPropertyAttribute, 102
 NetworkBehaviourWeavedAttribute, 378
 NetworkedWeavedAttribute, 409
 NetworkInput, 424
 NetworkInputWeavedAttribute, 427
 NetworkObjectHeader, 474
 NetworkStructWeavedAttribute, 672
WordOffset
 DefaultForPropertyAttribute, 102
 NetworkedWeavedAttribute, 409
WORDS
 NetworkObjectHeader, 475
 NetworkTRSPData, 680
 TickRate.Resolved, 874
WordsReadCount
 FusionStatisticsSnapshot, 847
WordsReadSize
 FusionStatisticsSnapshot, 847
WordsWrittenCount
 FusionStatisticsSnapshot, 847
WordsWrittenSize
 FusionStatisticsSnapshot, 847
Wrap
 SerializableDictionary< TKey, TValue >, 754
Write
 IElementReaderWriter< T >, 197
 IUnitySurrogate, 222
 NetworkId, 418, 419
 PlayerRef, 688
 RpcHeader, 731
 UnityArraySurrogate< T, ReaderWriter >, 225
 UnityDictionarySurrogate< TKeyType, TKeyReaderWriter, TValueType, TValueReaderWriter >, 227
 UnityLinkedListSurrogate< T, ReaderWriter >, 229
 UnitySurrogateBase, 231
 UnityValueSurrogate< T, TReaderWriter >, 233
Write< T >
 PlayerRef, 688
WriteBoolean
 ReadWriteUtilsForWeaver, 718
WriteEmptyNetworkBehaviourRef
 ReadWriteUtils, 710
 WriteFloat
 ReadWriteUtils, 710
 WriteInt
 SimulationMessage, 809
 WriteNetworkBehaviourRef
 ReadWriteUtils, 711
 WriteNetworkedObjectRef
 SimulationMessage, 809
 WriteNullBehaviourRef
 ReadWriteUtils, 711
 WriteQuaternion
 ReadWriteUtils, 711
 WriteStringUtf32NoHash
 ReadWriteUtilsForWeaver, 718
 WriteStringUtf32WithHash
 ReadWriteUtilsForWeaver, 718
 WriteStringUtf8NoHash
 ReadWriteUtilsForWeaver, 719
 WriteVector2
 ReadWriteUtils, 712
 WriteVector3
 ReadWriteUtils, 712
 SimulationMessage, 810
 WriteVector4
 ReadWriteUtils, 712

X
 QuaternionCompressed, 704
 Vector2Compressed, 898
 Vector3Compressed, 904
 Vector4Compressed, 909
x
 Simulation.AreaOfInterest, 785
xEncoded
 QuaternionCompressed, 703
 Vector2Compressed, 898
 Vector3Compressed, 903
 Vector4Compressed, 908

Y
 QuaternionCompressed, 704
 Vector2Compressed, 898
 Vector3Compressed, 904
 Vector4Compressed, 909
yEncoded
 QuaternionCompressed, 703
 Vector2Compressed, 898
 Vector3Compressed, 903
 Vector4Compressed, 908

Z
 QuaternionCompressed, 704
 Vector3Compressed, 904
 Vector4Compressed, 909
zEncoded
 QuaternionCompressed, 703
 Vector3Compressed, 903
 Vector4Compressed, 908