Photon Fusion

2.0.0

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

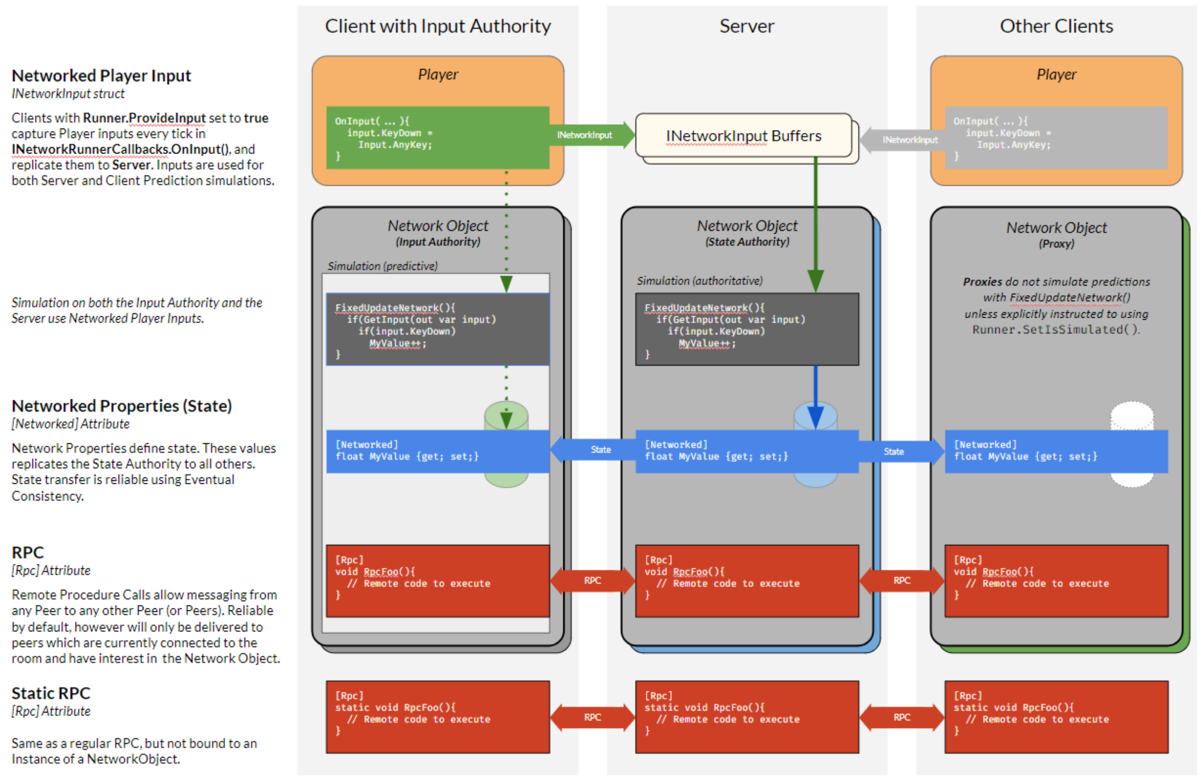
Photon Fusion Overview

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.



Networked Player Input
INetworkInput struct

Clients with **Runner.ProvideInput** set to **true** capture Player inputs every tick in **INetworkRunnerCallbacks.OnInput()**, and replicate them to **Server**. Inputs are used for both Server and Client Prediction simulations.

Simulation on both the Input Authority and the Server use Networked Player Inputs.

Networked Properties (State)
[Networked] Attribute

Network Properties define state. These values replicates the State Authority to all others. State transfer is reliable using Eventual Consistency.

RPC
[Rpc] Attribute

Remote Procedure Calls allow messaging from any Peer to any other Peer (or Peers). Reliable by default, however will only be delivered to peers which are currently connected to the room and have interest in the Network Object.

Static RPC
[Rpc] Attribute

Same as a regular RPC, but not bound to an Instance of a NetworkObject.

```
OnInput( ... ){
  input.KeyDown =
    Input.AnyKey;
}
```

```
FixedUpdateNetwork(){
  if(GetInput(out var input)
    if(input.KeyDown)
      MyValue++;
}
```

```
[Networked]
float MyValue {get; set;}
```

```
[Rpc]
void RpcFoo(){
  // Remote code to execute
}
```

```
[Rpc]
static void RpcFoo(){
  // Remote code to execute
}
```

Proxies do not simulate predictions with FixedUpdateNetwork() unless explicitly instructed to using `Runner.SetIsSimulated()`.

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.

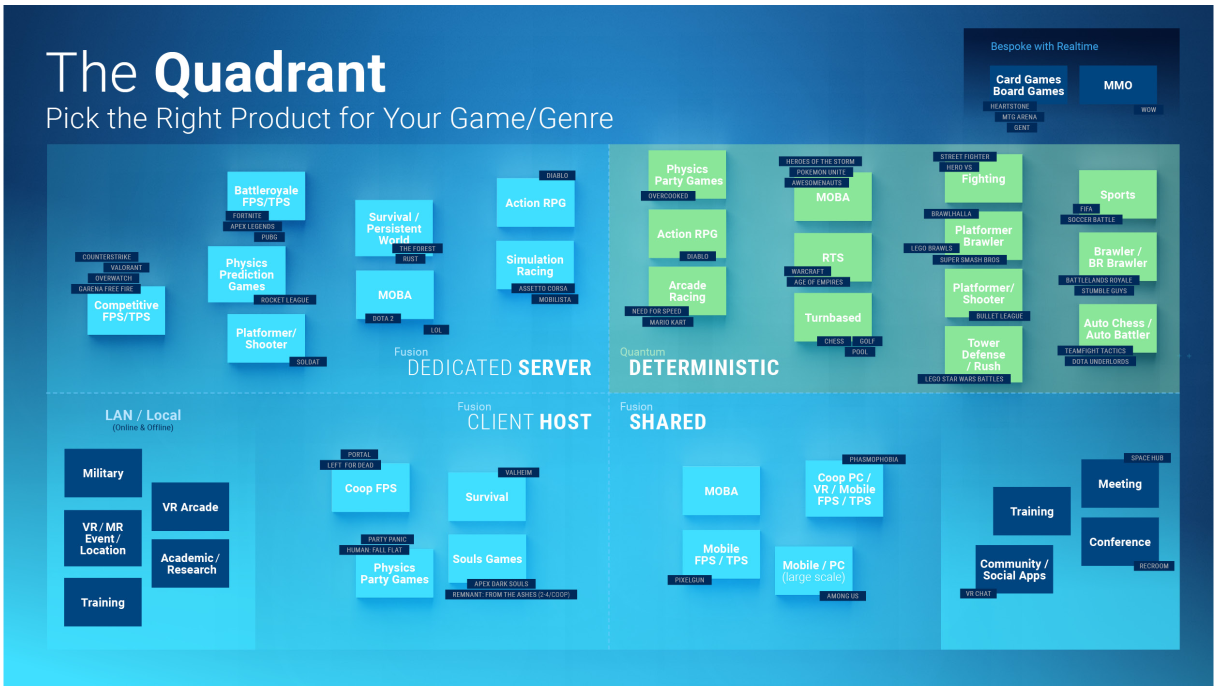
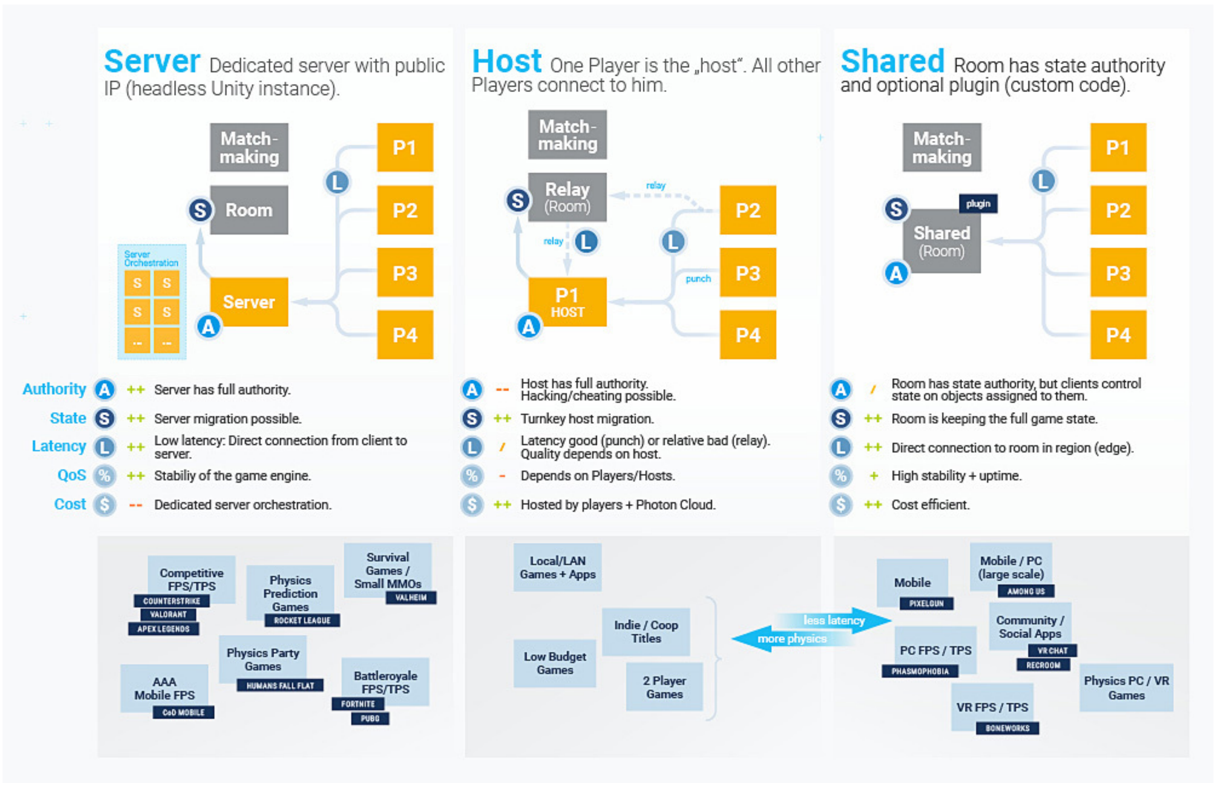


Figure 2.2 The Quadrant

2.2 Topology Differences



	Server		Host		Shared	
Authority	++	Server has full authority.	--	Host has full authority. Hacking/cheating possible.	/	Room has state authority, but clients control state on objects assigned to them.
State	++	Server migration possible.	++	Turnkey host migration.	++	Room is keeping the full game state.
Latency	++	Low latency: Direct connection from client to server.	/	Latency good (punch) or relative bad (relay). Quality depends on host.	++	Direct connection to room in region (edge).
QoS	++	Stabiliy of the game engine.	-	Depends on Players/Hosts.	+	High stability + uptime.
Cost	--	Dedicated server orchestration.	++	Hosted by players + Photon Cloud.	++	Cost efficient.

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:

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

Namespace Documentation

5.1 Fusion Namespace Reference

Classes

- class AccuracyAttribute

  Additional companion attribute to NetworkedAttribute, which indicates how floats should be compressed.
- struct Angle

  A Networked fusion type for degrees. This can be used with the NetworkedAttribute, in RPCs, or in NetworkInput structs.
- class AuthorityMasks

  Flag constants for input and state authority.
- 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 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 DoIfAttributeBase

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

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

  Base class for FieldsMask<T>.
- class HeapConfiguration

  Memory Heap Settings.
- 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 resimulation 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 Simulation↩ Behaviour and NetworkBehaviour classes.
- interface IAfterHostMigration

  Used to mark NetworkBehaviors that need to be react after a Host Migration process.
- interface IAfterTick

  Interface for AfterTick callback. Called after each tick simulation completes. Implement this interface on Simulation↩ Behaviour 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 IBeforeAllTicks

  Interface for BeforeAllTicks callback. Called before the resimulation 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 resimulation loop (on clients with prediction enabled), before state is set to the latest server snapshot. Implement this interface on SimulationBehaviour and NetworkBehaviour classes.
- interface IBeforeHitboxRegistration

  Interface for BeforeHitboxRegistration callback. Implement this interface on SimulationBehaviour and Network↩ Behaviour classes.
- interface IBeforeTick

  Interface for BeforeTick callback. Called before each tick is simulated. Implement this interface on Simulation↩ Behaviour 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 INetworkInput

  Flag interface for custom NetworkInput structs.
- 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 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 INetworkTRSPTeleport

  Implement this interface on a NetworkTRSP implementation to indicate it can be teleported.
- class InterpolatedErrorCorrectionSettings

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

Interface for SimulationEnter callback. Called when the NetworkObject joins AreaOfInterest. Implement this interface on SimulationBehaviour and NetworkBehaviour classes. Only applicable to SimulationConfig.StateReplicationModes.EventualConsistency.

- interface ISimulationExit

  Interface for the SimulationExit callback. Called when the NetworkObject leaves AreaOfInterest. Implement this interface on SimulationBehaviour and NetworkBehaviour classes. Only applicable to SimulationConfig.StateReplicationModes.EventualConsistency.
- struct LagCompensatedHit

  Defines a lag compensated query hit result.
- class LagCompensationSettings

  Settings for lag compensation history.
- class LobbyInfo

  Holds information about a Lobby.
- 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 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.
- class NetworkConfiguration

  Main network configuration class.
- struct NetworkDictionary

  Fusion type for networking Dictionaries. Maximum capacity is fixed, and is set with the CapacityAttribute.
- class NetworkedAttribute
- 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

  Translates INetworkInput structs and represents them in Fusions's unsafe allocated memory.
- struct NetworkLinkedList

  Fusion type for networking LinkedLists. Maximum capacity is fixed, and is set with the CapacityAttribute.

  Typical Usage:
- 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.
- struct NetworkObjectHeader

  Network object header information for a NetworkObject.
- struct NetworkObjectTypeId

  ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable.
- class NetworkPositionRotation

Use NetworkTransform (or any custom class derived from NetworkTRSP) to synchronize initial transform values. This component is non-functional.

- 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

  A decoupled NetworkObject prefab reference. Internally stored as a GUID.
- 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
- class NetworkRigidbody

  Use the Fusion Unity Physics Add-on, or your own variation of it to synchronize Rigidbodies. This component is non-functional.
- class NetworkRigidbody2D

  Use the Fusion Unity Physics Add-on, or your own variation of it to synchronize Rigidbodies. This component is non-functional.
- 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.
- struct NetworkSceneInfo

  The default implementation of INetworkSceneInfo. 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.
- class NetworkSimulationConfiguration

  Configuration for network conditions simulation (induced latency and loss).
- class NetworkString

  Fixed-size UTF32 string. All operations are alloc-free, except for converting to System.String.
- class NetworkStructWeavedAttribute

  Describes the total number of WORDs a Fusion.INetworkedStruct 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.
- struct PlayerRef

  Represents a Fusion player.
- class RenderAttribute

  Override default render settings for `[Networked]` properties.
- struct RenderTimeline

  Can be used to acquire RenderData for different points in time.
- 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 RpcInfo argument, that will include meta information about the RPC on the receiving peer.

- struct RpcInvokeInfo

  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:
- 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.MonoBehavour). Objects derived from this object can be associated with a NetworkRunner and Simulation. If a parent NetworkObject is found, this component will also be associated with that network entity.
- 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:
- class SimulationConfig

  Project configuration settings specific to how the Simulation class behaves.
- 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.
- class StatsMetaAttribute

  This stat goes on field elements of classes/structs and is used by FieldsMask.
- class UnitAttribute

  Unit Attribute class. Used to mark a field with the respective Units
- 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).

Enumerations

- enum **AnimatorSyncSettings**
- enum CompareOperator

  Comparison method for evaluating condition member value against compareToValues.
- enum ConnectionType

  Defines the type of the current connection with the Remote Peer, either the Server or a Client.
- enum **DrawIfMode**
- enum **EditorButtonVisibility**
- enum GameMode

  Fusion Game Mode.
- enum HitboxTypes

Defines the collision geometry type of a Hitbox.

- enum HitOptions
- enum **NetworkObjectAcquireResult**
- enum **NetworkObjectConnectionDataStatus**
- enum **NetworkObjectDestroyFlags**
- enum **NetworkObjectFlags** : int
- enum **NetworkObjectHeaderFlags** : int
- enum **NetworkObjectHeaderPlayerDataFlags** : int
- enum **NetworkObjectPacketFlags**
- enum **NetworkObjectRuntimeFlags** : int
- enum **NetworkPrefabTableGetPrefabResult**
- enum NetworkSceneInfoChangeSource

  What has contributed to the observed change in the scene info.
- enum **NetworkSceneInfoDefaultFlags** : uint
- enum NetworkSpawnFlags : short
- enum **NetworkSpawnStatus** : int
- enum **NetworkTypeIdKind**
- enum PageSizes

  Page Bit Shift Lookup Table.
- enum **PriorityLevel**
- enum RenderSource

  Indicates how available snapshot data should be used to render networked properties (in the chosen Render←↩ Timeframe).
- enum RenderTimeframe

  Indicates which point in time (or ¨timeframe¨) networked properties should be rendered in.
- enum RpcChannel
- enum RpcHostMode

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

  Results for the local RPC Invocation of the RPC method.
- enum 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 RpcSendMessageResult

  Result flags for the RPC message send operation.
- enum **RpcSources**
- enum **RpcTargets**
- enum **RpcTargetStatus**
- enum ScriptHeaderBackColor

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

  Icon to be rendered on the component graphic header in the Unity inspector.
- enum **ScriptHeaderStyle**
- enum SessionLobby

  Session Lobby Type.
- enum ShutdownReason

  Describes a list of Reason why the Fusion Runner was Shutdown.
- enum **SimulationBehaviourRuntimeFlags**
- enum **SimulationMessageInternalTypes**
- enum SimulationModes

  Flags for The type of network peer a simulation represents.
- enum 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 **StatAveraging**
- enum Topologies
- enum Units

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

Functions

- delegate FusionGlobalScriptableObjectLoadResult **FusionGlobalScriptableObjectLoadDelegate** (Type type)
- delegate void **FusionGlobalScriptableObjectUnloadDelegate** (FusionGlobalScriptableObject instance)
- delegate void **NetworkObjectSpawnDelegate** (NetworkSpawnOp result)
- unsafe delegate void **RpcInvokeDelegate** (NetworkBehaviour behaviour, SimulationMessage ∗message)
- unsafe delegate void **RpcStaticInvokeDelegate** (NetworkRunner runner, SimulationMessage ∗message)

5.1.1 Enumeration Type Documentation

5.1.1.1 CompareOperator

```
enum CompareOperator
```

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.

5.1.1.2 ConnectionType

```
enum ConnectionType
```

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 GameMode

```
enum GameMode
```

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.

5.1.1.4 HitboxTypes

```
enum HitboxTypes
```

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.

5.1.1.5 HitOptions

```
enum HitOptions
```

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.

5.1.1.6 NetworkSpawnFlags

```
enum NetworkSpawnFlags : short
```

Enumerator

DontDestroyOnLoad	Object get spawned as DontDestroyOnLoad on all clients.
SharedModeStateAuthMasterClient	In shared mode, override the state authority to PlayerRef.MasterClient.
SharedModeStateAuthLocalPlayer	In shared mode, override the state authority to local player.

5.1.1.7 RenderSource

```
enum RenderSource
```

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.

5.1.1.8 RenderTimeframe

```
enum RenderTimeframe
```

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

Enumerator

Auto	The timeframe will be chosen automatically.
Local	The default timeframe for owned and predicted objects.
Remote	The default timeframe for proxied objects.

5.1.1.9 RpcChannel

```
enum RpcChannel
```

Enumerator

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

5.1.1.10 RpcHostMode

```
enum RpcHostMode
```

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

Enumerator

SourceIsServer	If host invokes RPC RpcInfo.Source will be set to PlayerRef.None (default).
SourceIsHostPlayer	If host invokes RPC RpcInfo.Source will be set to the host's local player.

5.1.1.11 RpcLocalInvokeResult

```
enum RpcLocalInvokeResult
```

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 RpcAttribute.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
TagetPlayerIsNotLocal	Not invoked because target player is not local.

5.1.1.12 RpcSendCullResult

```
enum 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.

Enumerator

NotCulled	RPC has been sent. Check RpcInvokeInfo.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.
TargetPlayerIsLocalButRpcIsNotInvokableLocally	Send culled because target player is local and RpcAttribute.InvokeLocal is false.

5.1.1.13 RpcSendMessageResult

`enum RpcSendMessageResult`

Result flags for the RPC message send operation.

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.

5.1.1.14 SessionLobby

`enum SessionLobby`

Session Lobby Type.

Enumerator

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

5.1.1.15 ShutdownReason

`enum ShutdownReason`

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.

Enumerator

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.

5.1.1.16 SimulationModes

```
enum SimulationModes
```

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.

5.1.1.17 SimulationStages

```
enum 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.

Enumerator

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

5.1.1.18 Topologies

```
enum Topologies
```

Enumerator

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

5.1.1.19 Units

```
enum Units
```

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

Enumerator

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

5.2 Fusion.Analyzer Namespace Reference

Enumerations

- enum **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.Internal Namespace Reference

5.5 Fusion.LagCompensation Namespace Reference

Classes

- 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 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.
- 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 HitType

  Queries can hit either fusion's custom Hitbox or Unity's standard Physx/Box2D colliders.

Functions

- delegate void **PreProcessingDelegate** (Query query, List< HitboxRoot > rootCandidates, HashSet< int > processedColliderIndices)

5.5.1 Enumeration Type Documentation

5.5.1.1 HitType

```
enum HitType
```

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 Fusion.Protocol Namespace Reference

Classes

- interface IMessage

  Represents a Protocol Message.

5.7 Fusion.Runtime Namespace Reference

5.8 Fusion.Runtime.Unity Namespace Reference

5.9 Fusion.Sockets Namespace Reference

Classes

- struct NetAddress

  Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address.
- struct NetBitBufferList

Represents a linked list of Fusion.Sockets.NetBitBuffer
- 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 NetCommands : byte

    Describe the Type of a Command Packet.
- enum NetConnectFailedReason : byte

    The reason a connection with a remote server has failed.
- enum **NetConnectionStatus**
- enum NetDisconnectReason : byte

    Disconnect Reason Flag.
- enum NetPacketType : byte

    Describe the type of a Networked Packet.
- enum **OnConnectionRequestReply**

5.9.1 Enumeration Type Documentation

5.9.1.1 NetConnectFailedReason

```
enum NetConnectFailedReason : byte
```

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 Fusion.Sockets.Stun Namespace Reference

Enumerations

- enum NATType : byte

    Specifies UDP network type.

5.10.1 Enumeration Type Documentation

5.10.1.1 NATType

```
enum NATType :  byte
```

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.
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 packet can send a UDP packet back to the internal host.

5.11 UnityEngine Namespace Reference

5.12 UnityEngine.SceneManagement Namespace Reference

5.13 UnityEngine.Scripting Namespace Reference

5.14 UnityEngine.Serialization Namespace Reference

Chapter 6

Class Documentation

6.1 AccuracyAttribute Class Reference

Additional companion attribute to NetworkedAttribute, which indicates how floats should be compressed.

Inherits Attribute.

Public Member Functions

- **AccuracyAttribute** (double accuracy)

  Constructor new accuracy.
- **AccuracyAttribute** (float accuracy)

  Constructor new accuracy.
- AccuracyAttribute (string defaultAccuracyTag)

  Constructor that takes a named AccuracyDefaults constant. Accuracy for this property will be acquired from the NetworkProjectConfig.AccuracyDefaults settings.

6.1.1 Detailed Description

Additional companion attribute to NetworkedAttribute, which indicates how floats should be compressed.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 AccuracyAttribute()

```
AccuracyAttribute (
            string defaultAccuracyTag )
```

Constructor that takes a named AccuracyDefaults constant. Accuracy for this property will be acquired from the NetworkProjectConfig.AccuracyDefaults settings.

Parameters

defaultAccuracyTag	

6.2 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)
- override bool **Equals** (object obj)
- override int **GetHashCode** ()
- override string **ToString** ()

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)
- static implicit **operator Angle** (float value)
- static implicit **operator Angle** (int value)
- static **operator double** (Angle value)
- static **operator float** (Angle value)
- static bool **operator!=** (Angle a, Angle b)
- static Angle **operator+** (Angle a, Angle b)
- static Angle **operator-** (Angle a, Angle b)
- static bool **operator<** (Angle a, Angle b)
- static bool **operator<=** (Angle a, Angle b)
- static bool **operator==** (Angle a, Angle b)
- static bool **operator>** (Angle a, Angle b)
- static bool **operator>=** (Angle a, Angle b)

Public Attributes

- int **_value**

Static Public Attributes

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

6.2.1 Detailed Description

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

6.3 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 service←Name=null)
- static Task Service (Func< Task< bool > > 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.

6.3.1 Detailed Description

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

6.3.2 Member Function Documentation

6.3.2.1 ContinueWhenAll()

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

Run a continuation Task after all other Tasks have completed.

Parameters

precedingTasks	List of pending tasks to wait
action	Action to run after the Tasks
cancellationToken	ellationToken used to stop the Action

Returns

Async Task based on the Action

6.3.2.2 Run()

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

Run an Action asynchronously.

Parameters

action	Action to be invoked
cancellationToken	CancellationToken used to stop the Action
options	Extra Task Creation options

Returns

Async Task based on the Action

6.3.2.3 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

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

Returns

Service Task

6.4 AuthorityMasks Class Reference

Flag constants for input and state authority.

Static Public Attributes

- const int **ALL** = STATE | INPUT | PROXY
- const int **INPUT** = 1 << 1
- const int **NONE** = 0
- const int **PROXY** = 1 << 2
- const int **STATE** = 1 << 0

6.4.1 Detailed Description

Flag constants for input and state authority.

6.5 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, NetworkPositionRotation, NetworkRigidbody, NetworkRigidbody2D, NetworkRunner, RunnerVisibilityNodes, 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()

6.5.1 Detailed Description

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

6.5.2 Member Function Documentation

6.5.2.1 AddBehaviour< T >()

```
T AddBehaviour< T > ( )
```

Wrapper for Unity's GameObject.AddComponent()

Type Constraints

T* : *Behaviour

6.5.2.2 GetBehaviour< T >()

```
T GetBehaviour< T > ( )
```

Wrapper for Unity's GameObject.GetComponentInChildren()

Type Constraints

T* : *Behaviour

6.5.2.3 TryGetBehaviour< T >()

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

Wrapper for Unity's GameObject.TryGetComponent()

Type Constraints

T* : *Behaviour

6.6 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)
- **DisplayAsEnumAttribute** (Type enumType)

Properties

- Type **EnumType** `[get]`
- string **EnumTypeMemberName** `[get]`

6.6.1 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.

6.7 DoIfAttributeBase 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**
- bool **_isDouble**
- long **_longValue**
- CompareOperator **Compare**
- string **ConditionMember**
- bool **ErrorOnConditionMemberNotFound** = true

Protected Member Functions

- **DoIfAttributeBase** (string propertyPath, double compareToValue, CompareOperator compare)
- **DoIfAttributeBase** (string propertyPath, long compareToValue, CompareOperator compare)

6.7.1 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. Null = 0, False = 0, True = 1, Unity Object = InstanceId

6.8 DrawIfAttribute Class Reference

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

Inherits DoIfAttributeBase.

Public Member Functions

- **DrawIfAttribute** (string propertyPath)
- DrawIfAttribute (string propertyPath, bool compareToValue, CompareOperator compare=CompareOperator.Equal, DrawIfMode mode=DrawIfMode.ReadOnly)

  Constructor.
- DrawIfAttribute (string propertyPath, double compareToValue, CompareOperator compare=CompareOperator.Equal, DrawIfMode mode=DrawIfMode.ReadOnly)

  Constructor.
- **DrawIfAttribute** (string propertyPath, 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.

Public Attributes inherited from DoIfAttributeBase

- double **_doubleValue**
- bool **_isDouble**
- long **_longValue**
- CompareOperator **Compare**
- string **ConditionMember**
- bool **ErrorOnConditionMemberNotFound** = true

Properties

- bool **Hide** `[get, set]`

Additional Inherited Members

Protected Member Functions inherited from DoIfAttributeBase

- **DoIfAttributeBase** (string propertyPath, double compareToValue, CompareOperator compare)
- **DoIfAttributeBase** (string propertyPath, long compareToValue, CompareOperator compare)

6.8.1 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

6.8.2 Constructor & Destructor Documentation

6.8.2.1 DrawIfAttribute() [1/2]

```
DrawIfAttribute (
            string propertyPath,
            double compareToValue,
            CompareOperator compare = CompareOperator::Equal,
            DrawIfMode mode = DrawIfMode::ReadOnly )
```

Constructor.

Parameters

propertyPath	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

Parameters

compareToValue	The value to compare the member value against.
mode	How the field should be hidden (disabled or removed)
compare	How the condition member value and compareToValye will be evaluated.

6.8.2.2 DrawIfAttribute() [2/2]

```
DrawIfAttribute (
            string propertyPath,
            bool compareToValue,
            CompareOperator compare = CompareOperator::Equal,
            DrawIfMode mode = DrawIfMode::ReadOnly )
```

Constructor.

Parameters

propertyPath	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

Parameters

compareToValue	The value to compare the member value against.
compare	How the condition member value and compareToValye will be evaluated.
mode	How the field should be hidden (disabled or removed)

6.9 FieldsMask< T > Class Template Reference

Base class for FieldsMask<T>.

Inherits FieldsMask.

Public Member Functions

- **FieldsMask** ()

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

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.

6.9.1 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.

6.10 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.

6.10.1 Detailed Description

Memory Heap Settings.

6.11 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.

Public Member Functions inherited from Behaviour

- 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()

Public Attributes

- Vector3 **BoxExtents**

  When Type is set to HitboxTypes.Box, this defines the local-space geometry for narrow-phase checks.
- float **CapsuleHeight**

  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)

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

Static Public Member Functions inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

6.11.1 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.

6.12 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, IBeforeAllTicks, IAfterTick, and IAfterUpdate.

Public Member Functions

- void AfterTick ()

  Called after each tick simulation completes.
- void AfterUpdate ()

  Called at the end of the Fusion Update loop, before all Unity MonoBehaviour.Update() callbacks.
- void BeforeAllTicks (bool resimulation, int tickCount)

  Called before the resimulation loop (when applicable), and also before the forward simulation loop. Only called on Updates where resimulation or forward ticks are processed.
- 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, Pre↩ProcessingDelegate 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, PreProcessing↩Delegate 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 layer↩Mask=-1, HitOptions options=HitOptions.None, bool clearHits=true, QueryTriggerInteraction queryTrigger↩Interaction=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 subtick↩Accuracy=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 Hit↩ Options.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 Hit↩ Options.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 Hit↩ Options.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 Hit↩ Options.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, PreProcessing↩ Delegate preProcessRoots=null)

    Performs a lag-compensated raycast query against all registered hitboxes. If the HitOptions.IncludePhysX or Hit↩ Options.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 Hit↩ Options.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.
- void **WriteBenchmarkResults** ()

    Used internally to write the benchmark results.

Public Member Functions inherited from SimulationBehaviour

- 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.

Public Member Functions inherited from Behaviour

- 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()

- void BeforeAllTicks (bool resimulation, int tickCount)

  Called before the resimulation loop (when applicable), and also before the forward simulation loop. Only called on Updates where resimulation or forward ticks are processed.

- void AfterTick ()

  Called after each tick simulation completes.

- void AfterUpdate ()

  Called at the end of the Fusion Update loop, before all Unity MonoBehaviour.Update() callbacks.

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

Static Public Member Functions inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

Properties inherited from SimulationBehaviour

- bool **CanReceiveRenderCallback** `[get]`
- bool **CanReceiveSimulationCallback** `[get]`
- NetworkObject **Object** `[get]`

  The NetworkObject this component is associated with.
- NetworkRunner **Runner** `[get]`

  The NetworkRunner this component is associated with.

6.12.1 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.

6.12.2 Member Function Documentation

6.12.2.1 AfterTick()

```
void AfterTick ( )
```

Called after each tick simulation completes.

Implements IAfterTick.

6.12.2.2 AfterUpdate()

```
void AfterUpdate ( )
```

Called at the end of the Fusion Update loop, before all Unity MonoBehaviour.Update() callbacks.

Implements IAfterUpdate.

6.12.2.3 BeforeAllTicks()

```
void BeforeAllTicks (
             bool resimulation,
             int tickCount )
```

Called before the resimulation loop (when applicable), and also before the forward simulation loop. Only called on Updates where resimulation or forward ticks are processed.

Parameters

resimulation	True if this is being called during the resimulation loop. False if during the forward simulation loop.
tickCount	How many resimulation or forward ticks are going to be processed.

Implements IBeforeAllTicks.

6.12.2.4 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

query	The query containing all necessary information.
hits	List to be filled with hits (both hitboxes and/or static colliders, if included).
clearHits	Clear list of hits before filling with new ones (defaults to true).

Returns

The total number of hits found.

6.12.2.5 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

center	Center of the box in world space.
extents	Half of the size of the box in each dimension.
orientation	Rotation of the box.
tick	The exact tick to be queried
tickTo	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the *alpha* parameter for interpolation between *tick* and *tickTo* . If HitOptions.SubtickAccuracy is included on *options* , 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 *alpha* .
alpha	Interpolation value when querying between *tick* and *tickTo* . If HitOptions.SubtickAccuracy is included on *options* , 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.
hits	List to be filled with hits (both hitboxes and/or static colliders, if included).
layerMask	Only objects with matching layers will be checked against.
options	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
clearHits	Clear list of hits before filling with new ones (defaults to true).
queryTriggerInteraction	Trigger interaction behavior when also querying PhysX.
preProcessRoots	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.

6.12.2.6 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

center	Center of the box in world space.
extents	Half of the size of the box in each dimension.
orientation	Rotation of the box.
player	Player who ¨owns¨ this overlap. Used by the server to find the exact hitbox snapshots to check against.
hits	List to be filled with hits (both hitboxes and/or static colliders, if included).
layerMask	Only objects with matching layers will be checked against.
options	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
clearHits	Clear list of hits before filling with new ones (defaults to true).
queryTriggerInteraction	Trigger interaction behavior when also querying PhysX.
preProcessRoots	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.

6.12.2.7 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

query	The query containing all necessary information.
hits	List to be filled with hits (both hitboxes and/or static colliders, if included).
clearHits	Clear list of hits before filling with new ones (defaults to true).

Returns

The total number of hits found.

6.12.2.8 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

origin	Sphere center, in world-space
radius	Sphere radius
tick	The tick to be queried
tickTo	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the *alpha* parameter for interpolation between *tick* and *tickTo* . If HitOptions.SubtickAccuracy is included on *options* , 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 *alpha* .
alpha	Interpolation value when querying between *tick* and *tickTo* . If HitOptions.SubtickAccuracy is included on *options* , 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.
hits	List to be filled with hits (both hitboxes and/or static colliders, if included).
layerMask	Only objects with matching layers will be checked against.

Parameters

options	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
clearHits	Clear list of hits before filling with new ones (defaults to true).
queryTriggerInteraction	Trigger interaction behavior when also querying PhysX.
preProcessRoots	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

6.12.2.9 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

origin	Sphere center, in world-space
radius	Sphere radius
player	Player who ¨owns¨ this overlap. Used by the server to find the exact hitbox snapshots to check against.
hits	List to be filled with hits (both hitboxes and/or static colliders, if included).
layerMask	Only objects with matching layers will be checked against.
options	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
clearHits	Clear list of hits before filling with new ones (defaults to true).
queryTriggerInteraction	Trigger interaction behavior when also querying PhysX.
preProcessRoots	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

6.12.2.10 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

hitbox	The target hitbox to be queried in the past
tick	The tick to be queried
tickTo	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the *alpha* parameter for interpolation between *tick* and *tickTo* . If *subtickAccuracy* 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 *alpha* .
alpha	Interpolation value when querying between *tick* and *tickTo* . If *subtickAccuracy* 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.
position	Will be filled with the hitbox position at the time of the tick
rotation	Will be filled with the hitbox rotation at the time of the tick
subtickAccuracy	If the query should interpolate between ticks to reflect exactly what was seen on the client.

6.12.2.11 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

hitbox	The target hitbox to be queried in the past
player	Player who ¨owns¨ this overlap. Used by the server to find the exact hitbox snapshots to check against.
position	Will be filled with the hitbox position at the time of the tick
rotation	Will be filled with the hitbox rotation at the time of the tick
subTickAccuracy	If the query should interpolate between ticks to reflect exactly what was seen on the client.

6.12.2.12 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 Hit↩ Options.IncludeBox2D flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

query	The query containing all necessary information.
hit	Raycast results will be filled in here.

Returns

The total number of hits found.

6.12.2.13 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 Hit↩ Options.IncludeBox2D flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

origin	Raycast origin, in world-space
direction	Raycast direction, in world-space
length	Raycast length
tick	Simulation tick number to use as the time reference for the lag compensation (use this for server AI, and similar).
tickTo	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the *alpha* parameter for interpolation between *tick* and *tickTo* . If HitOptions.SubtickAccuracy is included on *options* , 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 *alpha* .

Parameters

alpha	Interpolation value when querying between *tick* and *tickTo* . If HitOptions.SubtickAccuracy is included on *options* , 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.
hit	Raycast results will be filled in here.
layerMask	Only objects with matching layers will be checked against.
options	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
queryTriggerInteraction	Trigger interaction behavior when also querying PhysX.
preProcessRoots	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

6.12.2.14 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 Hit↩ Options.IncludeBox2D flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

origin	Raycast origin, in world-space
direction	Raycast direction, in world-space
length	Raycast length
player	Player who ¨owns¨ this raycast. Used by the server to find the exact hitbox snapshots to check against.
hit	Raycast results will be filled in here.
layerMask	Only objects with matching layers will be checked against.
options	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
queryTriggerInteraction	Trigger interaction behavior when also querying PhysX.

Parameters

preProcessRoots	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

6.12.2.15 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 Hit↩ Options.IncludeBox2D flag is indicated, query will also include static colliders, Unity colliders are recommended for static geometry, rather than Hitboxes.

Parameters

query	The query containing all necessary information.
hits	List to be filled with hits (both hitboxes and/or static colliders, if included).
clearHits	Clear list of hits before filling with new ones (defaults to true).

Returns

The total number of hits found.

6.12.2.16 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 Hit↩ Options.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

origin	Raycast origin, in world-space
direction	Raycast direction, in world-space
length	Raycast length
tick	Simulation tick number to use as the time reference for the lag compensation (use this for server AI, and similar).
tickTo	Simulation tick number to use as the time reference for the lag compensation. If provided, must be combined with the *alpha* parameter for interpolation between *tick* and *tickTo* . If HitOptions.SubtickAccuracy is included on *options* , 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 *alpha* .
alpha	Interpolation value when querying between *tick* and *tickTo* . If HitOptions.SubtickAccuracy is included on *options* , 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.
hits	List to be filled with hits (both hitboxes and/or static colliders, if included).
layerMask	Only objects with matching layers will be checked against.
options	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
clearHits	Clear list of hits before filling with new ones (defaults to true).
queryTriggerInteraction	Trigger interaction behavior when also querying PhysX.
preProcessRoots	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

6.12.2.17 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 Hit↩ Options.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

origin	Raycast origin, in world-space
direction	Raycast direction, in world-space
length	Raycast length
player	Player who ¨owns¨ this raycast. Used by the server to find the exact hitbox snapshots to check against.
hits	List to be filled with hits (both hitboxes and/or static colliders, if included).
layerMask	Only objects with matching layers will be checked against.
options	Opt-in flags to compute with sub-tick accuracy (HitOptions.SubtickAccuracy) and/or to include PhysX (HitOptions.IncludePhysX) or Box2D (HitOptions.IncludeBox2D).
clearHits	Clear list of hits before filling with new ones (defaults to true).
queryTriggerInteraction	Trigger interaction behavior when also querying PhysX.
preProcessRoots	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

6.13 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 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** ()
- 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 Member Functions inherited from NetworkBehaviour

- virtual void **CopyBackingFieldsToState** (bool firstTime)
- void CopyStateFrom (NetworkBehaviour source)

  Copies entire state of passed in source NetworkBehaviour
- virtual void **CopyStateToBackingFields** ()
- virtual void Despawned (NetworkRunner runner, bool hasState)

  Called before the network object is despawned.
- override void FixedUpdateNetwork ()

  Fusion FixedUpdate timing callback.
- ArrayReader< T > **GetArrayReader< T >** (string property)
- BehaviourReader< T > GetBehaviourReader< T > (string property)
- ChangeDetector **GetChangeDetector** (ChangeDetector.Source source, bool copyInitial=true)
- DictionaryReader< K, V > **GetDictionaryReader< K, V >** (string property)
- T? GetInput< T > ()
- 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 Fixed↩ UpdateNetwork).
- LinkListReader< T > **GetLinkListReader< T >** (string property)
- int **GetLocalAuthorityMask** ()

  Gets a bitmask of AuthorityMasks flags, representing the current local authority over this NetworkObject.
- PropertyReader< T > GetPropertyReader< T > (string property)
- ref T ReinterpretState< T > (int offset=0)

  Allows read and write access to the internal state buffer.
- 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)

Public Member Functions inherited from SimulationBehaviour

- 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.

Public Member Functions inherited from Behaviour

- 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()

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 of 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 Spawned.
- Vector3 Offset

  Local-space offset of the broadphase bounding sphere from its transform position.

Public Attributes inherited from NetworkBehaviour

- int **offset**

  Gives access to the offset (in 32 bit words) and count (in 32 bit words) of this behaviour backing data.

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)

Protected Member Functions inherited from NetworkBehaviour

- virtual bool **ReplicateTo** (PlayerRef player)

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 current local player.
- HitboxManager **Manager** `[get]`

  Reference to associated hitbox manager (from which lag compensated queries can be performed).

Properties inherited from NetworkBehaviour

- 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]`
- bool **StateBufferIsValid** `[get]`
- int int count **WordInfo** `[get]`

Properties inherited from SimulationBehaviour

- bool **CanReceiveRenderCallback** `[get]`
- bool **CanReceiveSimulationCallback** `[get]`
- NetworkObject **Object** `[get]`

  The NetworkObject this component is associated with.
- NetworkRunner **Runner** `[get]`

  The NetworkRunner this component is associated with.

Additional Inherited Members

Static Public Member Functions inherited from NetworkBehaviour

- static ArrayReader< T > **GetArrayReader< T >** (Type behaviourType, string property)
- static BehaviourReader< T > GetBehaviourReader< T > (NetworkRunner runner, Type behaviourType, string property)
- static BehaviourReader< TProperty > GetBehaviourReader< TBehaviour, TProperty > (NetworkRunner runner, string property)
- static DictionaryReader< K, V > **GetDictionaryReader< K, V >** (Type behaviourType, string property)
- static LinkListReader< T > **GetLinkListReader< T >** (Type behaviourType, string property)
- static PropertyReader< T > GetPropertyReader< T > (Type behaviourType, string property)
- static PropertyReader< TProperty > GetPropertyReader< TBehaviour, TProperty > (string property)
- 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 > ()
- static T ∗ MakePtr< T > (T defaultValue)

- static ref T MakeRef< T > ()
- static ref T MakeRef< T > (T defaultValue)
- static int **NetworkDeserialize** (NetworkRunner runner, byte ∗data, ref NetworkBehaviour result)
- static int **NetworkSerialize** (NetworkRunner runner, NetworkBehaviour obj, byte ∗data)
- static NetworkBehaviour **NetworkUnwrap** (NetworkRunner runner, NetworkBehaviourId wrapper)
- static NetworkBehaviourId **NetworkWrap** (NetworkRunner runner, NetworkBehaviour obj)
- static implicit operator NetworkBehaviourId (NetworkBehaviour behaviour)

  Converts NetworkBehaviour to NetworkBehaviourId.

Static Public Member Functions inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

6.13.1 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.

6.13.2 Member Enumeration Documentation

6.13.2.1 ConfigFlags

```
enum ConfigFlags : int
```

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.

6.13.3 Member Function Documentation

6.13.3.1 Despawned()

```
override void Despawned (
            NetworkRunner runner,
            bool hasState )  [virtual]
```

Called before the network object is despawned.

Parameters

hasState	If the state of the behaviour is still accessible

Reimplemented from NetworkBehaviour.

6.13.3.2 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

hitbox	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 *hitbox* is outside the valid range.
AssertException	In Debug configuration, if the *hitbox* is not part of this root.

6.13.3.3 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

hitbox	A hitbox instance under the hierarchy of this root.
setActive	If the hitbox should be activated or deactivated.

Exceptions

ArgumentOutOfRangeException	If the Hitbox.HitboxIndex of the *hitbox* is outside the valid range.
AssertException	In Debug configuration, if the *hitbox* is not part of this root.

6.13.4 Member Data Documentation

6.13.4.1 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 of 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.

6.13.4.2 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.

6.14 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.

6.14.1 Detailed Description

Project configuration settings specific to how the Host Migration behaves.

6.15 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.

6.15.1 Detailed Description

Transitory Holder with all necessary information to restart the Fusion Runner after the Host Migration has completed.

6.16 IAfterAllTicks Interface Reference

Interface for AfterAllTicks callback. Called after the resimulation 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 resimulation loop (when applicable), and also after the forward simulation loop. Only called on Updates where resimulation or forward ticks are processed.

6.16.1 Detailed Description

Interface for AfterAllTicks callback. Called after the resimulation loop (when applicable), and also after the forward simulation loop. Implement this interface on SimulationBehaviour and NetworkBehaviour classes.

6.16.2 Member Function Documentation

6.16.2.1 AfterAllTicks()

```
void AfterAllTicks (
             bool resimulation,
             int tickCount )
```

Called after the resimulation loop (when applicable), and also after the forward simulation loop. Only called on Updates where resimulation or forward ticks are processed.

Parameters

resimulation	True if this is being called during the resimulation loop. False if during the forward simulation loop.
tickCount	How many resimulation or forward ticks are going to be processed.

6.17 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.

6.17.1 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.

6.18 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.

6.18.1 Detailed Description

Used to mark NetworkBehaviors that need to be react after a Host Migration process.

6.19 IAfterTick Interface Reference

Interface for AfterTick callback. Called after each tick simulation completes. Implement this interface on Simulation↩
Behaviour and NetworkBehaviour classes.

Inherited by HitboxManager.

Public Member Functions

- void AfterTick ()

  Called after each tick simulation completes.

6.19.1 Detailed Description

Interface for AfterTick callback. Called after each tick simulation completes. Implement this interface on Simulation↩ Behaviour and NetworkBehaviour classes.

6.19.2 Member Function Documentation

6.19.2.1 AfterTick()

```
void AfterTick ( )
```

Called after each tick simulation completes.

Implemented in HitboxManager.

6.20 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.

Inherited by HitboxManager.

Public Member Functions

- void AfterUpdate ()

  Called at the end of the Fusion Update loop, before all Unity MonoBehaviour.Update() callbacks.

6.20.1 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.

6.20.2 Member Function Documentation

6.20.2.1 AfterUpdate()

```
void AfterUpdate ( )
```

Called at the end of the Fusion Update loop, before all Unity MonoBehaviour.Update() callbacks.

Implemented in HitboxManager.

6.21 IBeforeAllTicks Interface Reference

Interface for BeforeAllTicks callback. Called before the resimulation loop (when applicable), and also before the forward simulation loop. Implement this interface on SimulationBehaviour and NetworkBehaviour classes.

Inherited by HitboxManager, and NetworkTransform.

Public Member Functions

- void BeforeAllTicks (bool resimulation, int tickCount)

  Called before the resimulation loop (when applicable), and also before the forward simulation loop. Only called on Updates where resimulation or forward ticks are processed.

6.21.1 Detailed Description

Interface for BeforeAllTicks callback. Called before the resimulation loop (when applicable), and also before the forward simulation loop. Implement this interface on SimulationBehaviour and NetworkBehaviour classes.

6.21.2 Member Function Documentation

6.21.2.1 BeforeAllTicks()

```
void BeforeAllTicks (
            bool resimulation,
            int tickCount )
```

Called before the resimulation loop (when applicable), and also before the forward simulation loop. Only called on Updates where resimulation or forward ticks are processed.

Parameters

resimulation	True if this is being called during the resimulation loop. False if during the forward simulation loop.
tickCount	How many resimulation or forward ticks are going to be processed.

Implemented in HitboxManager.

6.22 IBeforeClientPredictionReset Interface Reference

Callback interface for BeforeClientPredictionReset. Called at the very start of the resimulation loop (on clients with prediction enabled), before state is set to the latest server snapshot. Implement this interface on Simulation↩
Behaviour and NetworkBehaviour classes.

Public Member Functions

- void **BeforeClientPredictionReset** ()

  Called at the very start of the resimulation loop (on clients with prediction enabled), before state is set to the latest server snapshot.

6.22.1 Detailed Description

Callback interface for BeforeClientPredictionReset. Called at the very start of the resimulation loop (on clients with prediction enabled), before state is set to the latest server snapshot. Implement this interface on Simulation←
Behaviour and NetworkBehaviour classes.

6.23 IBeforeHitboxRegistration Interface Reference

Interface for BeforeHitboxRegistration callback. Implement this interface on SimulationBehaviour and Network←
Behaviour classes.

Public Member Functions

- void **BeforeHitboxRegistration** ()

  Called immediately before the HitboxManager registers hitboxes in a snapshot.

6.23.1 Detailed Description

Interface for BeforeHitboxRegistration callback. Implement this interface on SimulationBehaviour and Network←
Behaviour classes.

6.24 IBeforeTick Interface Reference

Interface for BeforeTick callback. Called before each tick is simulated. Implement this interface on Simulation←
Behaviour and NetworkBehaviour classes.

Public Member Functions

- void **BeforeTick** ()

  Called before each tick is simulated.

6.24.1 Detailed Description

Interface for BeforeTick callback. Called before each tick is simulated. Implement this interface on Simulation←
Behaviour and NetworkBehaviour classes.

6.25 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.

6.25.1 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.

6.26 INetworkInput Interface Reference

Flag interface for custom NetworkInput structs.

6.26.1 Detailed Description

Flag interface for custom NetworkInput structs.

6.27 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 NetworkPrefabAcquire↩Context context, out NetworkObject result)
- void **ReleaseInstance** (NetworkRunner runner, in NetworkObjectReleaseContext context)

6.27.1 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().

6.28 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)
- void **OnObjectEnterAOI** (NetworkRunner runner, NetworkObject obj, PlayerRef player)
- void **OnObjectExitAOI** (NetworkRunner runner, NetworkObject obj, PlayerRef player)
- 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)
- void **OnReliableDataReceived** (NetworkRunner runner, PlayerRef player, ReliableKey key, ArraySegment< byte > data)
- void **OnSceneLoadDone** (NetworkRunner runner)
- void **OnSceneLoadStart** (NetworkRunner runner)
- 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.

6.28.1 Detailed Description

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

6.28.2 Member Function Documentation

6.28.2.1 OnConnectRequest()

```
void OnConnectRequest (
            NetworkRunner runner,
            NetworkRunnerCallbackArgs.ConnectRequest request,
            byte[ ] token )
```

Callback when NetworkRunner receives a Connection Request from a Remote Client.

Parameters

runner	Local NetworkRunner
request	Request information
token	Request Token

6.28.2.2 OnCustomAuthenticationResponse()

```
void OnCustomAuthenticationResponse (
            NetworkRunner runner,
            Dictionary< string, object > data )
```

Callback is invoked when the Authentication procedure returns a response from the Authentication Server.

Parameters

runner	The runner this object exists on
data	Custom Authentication Reply Values

6.28.2.3 OnHostMigration()

```
void OnHostMigration (
            NetworkRunner runner,
            HostMigrationToken hostMigrationToken )
```

Callback is invoked when the Host Migration process has started.

Parameters

runner	The runner this object exists on
hostMigrationToken	Migration Token that stores all necessary information to restart the Fusion Runner

6.28.2.4 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 */ });
```

6.28.2.5 OnInputMissing()

```
void OnInputMissing (
            NetworkRunner runner,
            PlayerRef player,
            NetworkInput input )
```

Parameters

runner	
input	

6.28.2.6 OnSessionListUpdated()

```
void OnSessionListUpdated (
            NetworkRunner runner,
            List< SessionInfo > sessionList )
```

This callback is invoked when a new List of Sessions is received from Photon Cloud.

Parameters

runner	The runner this object exists on
sessionList	Updated list of Session

6.28.2.7 OnShutdown()

```
void OnShutdown (
            NetworkRunner runner,
            ShutdownReason shutdownReason )
```

Called when the runner is shutdown.

Parameters

runner	The runner being shutdown
shutdownReason	Describes the reason Fusion was Shutdown

6.28.2.8 OnUserSimulationMessage()

```
void OnUserSimulationMessage (
            NetworkRunner runner,
            SimulationMessagePtr message )
```

This callback is invoked when a manually dispatched simulation message is received from a remote peer.

Parameters

runner	The runner this message is for
message	The message pointer

6.29 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)
- void **Shutdown** (NetworkRunner runner)

6.29.1 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 StartGame↩ Args.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.

6.30 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.

6.30.1 Detailed Description

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

6.30.2 Member Function Documentation

6.30.2.1 Teleport()

```
void Teleport (
            Vector3?  position = null,
            Quaternion?  rotation = null )
```

Teleports to the indicated values, and network the Teleport event.

Implemented in NetworkTransform.

6.31 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

6.31.1 Detailed Description

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

6.31.2 Member Data Documentation

6.31.2.1 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.

6.31.2.2 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.

6.31.2.3 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.

6.31.2.4 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.

6.31.2.5 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.

6.31.2.6 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.

6.31.2.7 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.

6.31.2.8 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.

6.31.2.9 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.

6.32 ISimulationEnter Interface Reference

Interface for SimulationEnter callback. Called when the NetworkObject joins AreaOfInterest. Implement this interface on SimulationBehaviour and NetworkBehaviour classes. Only applicable to SimulationConfig.State↩ReplicationModes.EventualConsistency.

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.

6.32.1 Detailed Description

Interface for SimulationEnter callback. Called when the NetworkObject joins AreaOfInterest. Implement this interface on SimulationBehaviour and NetworkBehaviour classes. Only applicable to SimulationConfig.State↩ReplicationModes.EventualConsistency.

6.33 ISimulationExit Interface Reference

Interface for the SimulationExit callback. Called when the NetworkObject leaves AreaOfInterest. Implement this interface on SimulationBehaviour and NetworkBehaviour classes. Only applicable to SimulationConfig.State↩ReplicationModes.EventualConsistency.

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.

6.33.1 Detailed Description

Interface for the SimulationExit callback. Called when the NetworkObject leaves AreaOfInterest. Implement this interface on SimulationBehaviour and NetworkBehaviour classes. Only applicable to SimulationConfig.State↩ReplicationModes.EventualConsistency.

6.34 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 **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).

6.34.1 Detailed Description

Defines a lag compensated query hit result.

6.34.2 Member Function Documentation

6.34.2.1 operator LagCompensatedHit() [1/2]

```
static operator LagCompensatedHit (
            RaycastHit raycastHit )  [explicit], [static]
```

Creates a LagCompensatedHit structure from the information on a Unity RaycastHit.

Parameters

raycastHit	The RaycastHit used as source.

Returns

The built LagCompensatedHit structure.

6.34.2.2 operator LagCompensatedHit() [2/2]

```
static operator LagCompensatedHit (
            RaycastHit2D raycastHit2D )  [explicit], [static]
```

Creates a LagCompensatedHit structure from the information on a Unity RaycastHit2D.

Parameters

raycastHit2D	The RaycastHit2D used as source.

Returns

The built LagCompensatedHit structure.

6.35 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)

6.35.1 Detailed Description

Class that represents a box overlap query. Used to query against the NetworkRunner.LagCompensation API.

6.35.2 Constructor & Destructor Documentation

6.35.2.1 BoxOverlapQuery() [1/2]

```
BoxOverlapQuery (
            ref BoxOverlapQueryParams boxOverlapParams )
```

Create a new BoxOverlapQuery with the given boxOverlapParams.

Parameters

boxOverlapParams	The parameters to be used when creating the query.

6.35.2.2 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

boxOverlapParams	The parameters to be used when creating the query.
physXOverlapHitsCache	Array to write the results of the PhysX query if used.
box2DOverlapHitsCache	Array to write the results of the Box2D query if used.

6.36 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**
- Vector3 **Extents**
- QueryParams **QueryParams**
- Quaternion **Rotation**
- int **StaticHitsCapacity**

6.36.1 Detailed Description

Base parameters needed to execute a box overlap query.

6.36.2 Constructor & Destructor Documentation

6.36.2.1 BoxOverlapQueryParams()

```
BoxOverlapQueryParams (
            QueryParams queryParams,
            Vector3 center,
            Vector3 extents,
            Quaternion rotation,
            int staticHitsCapacity )
```

Create a new BoxOverlapQueryParams

Parameters

queryParams	Parameters to be used
center	The query center
extents	The query extents
rotation	The query rotation
staticHitsCapacity	Capacity for the cached PhysX and Box2D static hits.

6.37 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** ()

6.37.1 Detailed Description

Provide a way to iterate over BVH and return a BVHNodeDrawInfo for each node.

6.38 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.

6.38.1 Detailed Description

Container class to provide the necessary info to draw nodes from the BVH.

6.39 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.
- float **CapsuleHeight** `[get]`

  The height for capsule colliders.

  See also

  HitboxTypes
- 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.

6.39.1 Detailed Description

Container class to provide the necessary information to draw a hitbox collider.

6.40 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** ()

6.40.1 Detailed Description

Provide a way to iterate over the HitboxBuffer.HitboxSnapshot and return the ColliderDrawInfo for each collider on the snapshot.

6.41 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.

6.41.1 Detailed Description

Provide access to iterate over the lag compensation system components and give the necessary information to draw them.

6.41.2 Member Function Documentation

6.41.2.1 GizmosDrawWireCapsule()

```
static void GizmosDrawWireCapsule (
             Vector3 topCenter,
             Vector3 bottomCenter,
             float capsuleRadius )  [static]
```

Method to draw capsules out of simple shapes.

Parameters

topCenter	The top capsule end position
bottomCenter	The bottom capsule end posistion
capsuleRadius	The capsule radius

6.42 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.

6.42.1 Detailed Description

Details regarding a shape intersection. It does not carry information about the intersection happening or not.

6.43 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**
- QueryParams **QueryParams**

6.43.1 Detailed Description

Query parameters for position rotation query.

6.43.2 Constructor & Destructor Documentation

6.43.2.1 PositionRotationQueryParams()

```
PositionRotationQueryParams (
            QueryParams queryParams,
            Hitbox hitbox )
```

Create a new PositionRotationQueryParams.

Parameters

queryParams	Parameters to be used
hitbox	The hitbox to be queried

6.44 QueryParams Struct Reference

Base parameters needed to execute a query.

Public Attributes

- float? **Alpha**
- LayerMask **LayerMask**
- HitOptions **Options**
- PlayerRef **Player**
- PreProcessingDelegate **PreProcessingDelegate**
- int **Tick**
- int? **TickTo**
- QueryTriggerInteraction **TriggerInteraction**
- void ∗ **UserArgs**

6.44.1 Detailed Description

Base parameters needed to execute a query.

6.45 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.

Public Member Functions inherited from RaycastQuery

- RaycastQuery (ref RaycastQueryParams raycastQueryParams)

  Create a new RaycastQuery with the given RaycastQueryParams

Additional Inherited Members

Public Attributes inherited from RaycastQuery

- Vector3 **Direction**
- float **Length**
- Vector3 **Origin**

Protected Member Functions inherited from RaycastQuery

- override bool **Check** (ref AABB bounds)

6.45.1 Detailed Description

Class that represents a raycast all query. Used to query against the NetworkRunner.LagCompensation API.

6.45.2 Constructor & Destructor Documentation

6.45.2.1 RaycastAllQuery() [1/2]

```
RaycastAllQuery (
            ref RaycastQueryParams raycastQueryParams )
```

Create a new RaycastAllQuery with the given RaycastQueryParams.

Parameters

raycastQueryParams	The parameters to be used when creating the query.

6.45.2.2 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

raycastQueryParams	The parameters to be used when creating the query.
physXRaycastHitsCache	Array to write the results of the PhysX query if used.
box2DRaycastHitCache	Array to write the results of the Box2D query if used.

6.46 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**
- float **Length**
- Vector3 **Origin**

Protected Member Functions

- override bool **Check** (ref AABB bounds)

6.46.1 Detailed Description

Class that represents a raycast query. Used to query against the NetworkRunner.LagCompensation API.

6.46.2 Constructor & Destructor Documentation

6.46.2.1 RaycastQuery()

```
RaycastQuery (
            ref RaycastQueryParams raycastQueryParams )
```

Create a new RaycastQuery with the given RaycastQueryParams

Parameters

raycastQueryParams	The parameters to be used when creating the query.

6.47 RaycastQueryParams Struct Reference

Base parameters needed to execute a raycast query.

Public Member Functions

- RaycastQueryParams (QueryParams queryParams, Vector3 origin, Vector3 direction, float length, int static↩ HitsCapacity=64)

  Create a new RaycastQueryParams

Public Attributes

- Vector3 **Direction**
- float **Length**
- Vector3 **Origin**
- QueryParams **QueryParams**
- int **StaticHitsCapacity**

6.47.1 Detailed Description

Base parameters needed to execute a raycast query.

6.47.2 Constructor & Destructor Documentation

6.47.2.1 RaycastQueryParams()

```
RaycastQueryParams (
            QueryParams queryParams,
            Vector3 origin,
            Vector3 direction,
            float length,
            int staticHitsCapacity = 64 )
```

Create a new RaycastQueryParams

Parameters

queryParams	Parameters to be used
origin	The raycast origin
direction	The raycast direction
length	The raycast max length
staticHitsCapacity	Capacity for the cached PhysX and Box2D static hits.

6.48 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** ()

6.48.1 Detailed Description

Provide a way to iterate over the HitboxBuffer and return the HitboxColliderContainerDraw container for each snapshot on the buffer.

6.49 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[ ] physXOverlap↩ HitsCache, Collider2D[ ] box2DOverlapHitsCache)

  Create a new SphereOverlapQuery with the given SphereOverlapQueryParams.

Public Attributes

- Vector3 **Center**
- float **Radius**

Protected Member Functions

- override bool **Check** (ref AABB bounds)

6.49.1 Detailed Description

Class that represents a sphere overlap query. Used to query against the NetworkRunner.LagCompensation API.

6.49.2 Constructor & Destructor Documentation

6.49.2.1 SphereOverlapQuery() [1/2]

```
SphereOverlapQuery (
            ref SphereOverlapQueryParams sphereOverlapParams )
```

Create a new SphereOverlapQuery with the given SphereOverlapQueryParams.

Parameters

sphereOverlapParams	The parameters to be used when creating the query.

6.49.2.2 SphereOverlapQuery() [2/2]

```
SphereOverlapQuery (
            ref SphereOverlapQueryParams sphereOverlapParams,
            Collider[] physXOverlapHitsCache,
            Collider2D[] box2DOverlapHitsCache )
```

Create a new SphereOverlapQuery with the given SphereOverlapQueryParams.

Parameters

sphereOverlapParams	The parameters to be used when creating the query.
physXOverlapHitsCache	Array to write the results of the PhysX query if used.
box2DOverlapHitsCache	Array to write the results of the Box2D query if used.

6.50 SphereOverlapQueryParams Struct Reference

Base parameters needed to execute a sphere overlap query.

Public Member Functions

- SphereOverlapQueryParams (QueryParams queryParams, Vector3 center, float radius, int staticHits↩ Capacity)

  Create a new SphereOverlapQueryParams.

Public Attributes

- Vector3 **Center**
- QueryParams **QueryParams**
- float **Radius**
- int **StaticHitsCapacity**

6.50.1 Detailed Description

Base parameters needed to execute a sphere overlap query.

6.50.2 Constructor & Destructor Documentation

6.50.2.1 SphereOverlapQueryParams()

```
SphereOverlapQueryParams (
            QueryParams queryParams,
            Vector3 center,
            float radius,
            int staticHitsCapacity )
```

Create a new SphereOverlapQueryParams.

Parameters

queryParams	Parameters to be used
center	The query center
radius	The query radius
staticHitsCapacity	Capacity for the cached PhysX and Box2D static hits.

6.51 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
- 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.

6.51.1 Detailed Description

Settings for lag compensation history.

6.52 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.

6.52.1 Detailed Description

Holds information about a Lobby.

6.53 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, StopOnT > (this Transform transform)
- static T[ ] FindObjectOfTypeInOrder< T > (this UnityEngine.SceneManagement.Scene scene)
- static CastT[ ] FindObjectOfTypeInOrder< T, CastT > (this UnityEngine.SceneManagement.Scene scene)
- 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 CastT[ ] FindObjectsOfTypeInOrder< T, CastT > (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, CastT > (this UnityEngine.SceneManagement.Scene scene, List< CastT > 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, StopOnT > (this Transform t, bool includeInactive)
- static T GetNestedComponentInParent< T, StopOnT > (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, StopOnT > (this Transform t)

  UNTESTED.
- static List< T > GetNestedComponentsInChildren< T > (this Transform t, List< T > list, bool include↩ Inactive=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, SearchT, StopT > (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, StopOnT > (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, StopT > (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.

6.53.1 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.

6.53.2 Member Function Documentation

6.53.2.1 EnsureRootComponentExists< T, StopOnT >()

```
static T EnsureRootComponentExists< T, StopOnT > (
            this Transform transform )  [static]
```

Type Constraints

T* : *Component

StopOnT* : *Component

6.53.2.2 FindObjectOfTypeInOrder< T >()

```
static T[] FindObjectOfTypeInOrder< T > (
            this UnityEngine::SceneManagement::Scene scene )  [static]
```

Type Constraints

T* : *class

6.53.2.3 FindObjectOfTypeInOrder< T, CastT >()

```
static CastT[] FindObjectOfTypeInOrder< T, CastT > (
            this UnityEngine::SceneManagement::Scene scene )  [static]
```

Type Constraints

T* : *class

CastT* : *class

6.53.2.4 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

6.53.2.5 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

T	

Parameters

scene	
list	Supplied list that will be populated by this find.
includeInactive	Whether results should include inactive components.

Type Constraints

T* : *class

6.53.2.6 FindObjectsOfTypeInOrder< T, CastT >() [1/2]

```
static CastT[] FindObjectsOfTypeInOrder< T, CastT > (
            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

T	The type being searched for.
CastT	Casts all found objects to this type, and returns collection of this type. Objects that fail cast are excluded.

Parameters

scene	
includeInactive	Whether results should include inactive components.

Type Constraints

T* : *class

CastT* : *class

6.53.2.7 FindObjectsOfTypeInOrder< T, CastT >() [2/2]

```
static void FindObjectsOfTypeInOrder< T, CastT > (
            this UnityEngine::SceneManagement::Scene scene,
            List< CastT > 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

T	
CastT	

Parameters

scene	
list	Supplied list that will be filled with found objects.
includeInactive	Whether results should include inactive components.

Type Constraints

> ***T*** **:** ***class***
> ***CastT*** **:** ***class***

6.53.2.8 GetNestedComponentInChildren< T, StopOnT >()

```
static T GetNestedComponentInChildren< T, StopOnT > (
             this Transform t,
             bool includeInactive )  [static]
```

Type Constraints

> ***T*** **:** ***class***
> ***StopOnT*** **:** ***class***

6.53.2.9 GetNestedComponentInParent< T, StopOnT >()

```
static T GetNestedComponentInParent< T, StopOnT > (
             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***
> ***StopOnT*** **:** ***class***

6.53.2.10 GetNestedComponentInParents< T, StopOnT >()

```
static T GetNestedComponentInParents< T, StopOnT > (
             this Transform t )  [static]
```

UNTESTED.

Type Constraints

> ***T*** **:** ***class***
> ***StopOnT*** **:** ***class***

6.53.2.11 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

6.53.2.12 GetNestedComponentsInChildren< T, SearchT, StopT >()

```
static void GetNestedComponentsInChildren< T, SearchT, StopT > (
             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.
SearchT	Find components of this class or interface type.
StopT	When this component is found, no further recursing will be performed on that node.

Type Constraints

T* : *class

SearchT* : *class

6.53.2.13 GetNestedComponentsInChildren< T, StopOnT >()

```
static List< T > GetNestedComponentsInChildren< T, StopOnT > (
             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

StopOnT* : *class

6.53.2.14 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

6.53.2.15 GetNestedComponentsInParents< T, StopT >()

```
static void GetNestedComponentsInParents< T, StopT > (
            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

StopT* : *class

6.53.2.16 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

6.54 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.

Public Member Functions

- void **Clear** ()
- void CopyFrom (List< T > source, int sourceOffset, int sourceCount)

    Copies a range of values in from a supplied source list.
- void CopyFrom (T[ ] source, int sourceOffset, int sourceCount)

    Copies a range of values in from a supplied source array.
- 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)
- void CopyTo (T[ ] array, bool throwIfOverflow=true)

    Copies values to the supplied array.
- T **Get** (int index)

    Returns the array value at supplied index.
- Enumerator **GetEnumerator** ()
- 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 **ToListString** ()

    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** ()
- override string **ToString** ()

Static Public Member Functions

- static implicit **operator NetworkArrayReadOnly** (NetworkArray< T > value)

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]`

6.54.1 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.

6.54.2 Member Function Documentation

6.54.2.1 CopyFrom() [1/2]

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

Copies a range of values in from a supplied source list.

Parameters

sourceOffset	Starting index of elements in source.
sourceCount	Number of sequential source elements to copy in.

6.54.2.2 CopyFrom() [2/2]

```
void CopyFrom (
            T[] source,
            int sourceOffset,
            int sourceCount )
```

Copies a range of values in from a supplied source array.

Parameters

sourceOffset	Starting index of elements in source.
sourceCount	Number of sequential source elements to copy in.

6.54.2.3 CopyTo()

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

Copies values to the supplied array.

Parameters

array	
throwIfOverflow	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.

6.55 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.

Public Member Functions

- virtual void **CopyBackingFieldsToState** (bool firstTime)
- void CopyStateFrom (NetworkBehaviour source)

  Copies entire state of passed in source NetworkBehaviour
- virtual void **CopyStateToBackingFields** ()
- virtual void Despawned (NetworkRunner runner, bool hasState)

  Called before the network object is despawned.
- override void FixedUpdateNetwork ()

  Fusion FixedUpdate timing callback.
- ArrayReader< T > **GetArrayReader< T >** (string property)
- BehaviourReader< T > GetBehaviourReader< T > (string property)
- ChangeDetector **GetChangeDetector** (ChangeDetector.Source source, bool copyInitial=true)
- DictionaryReader< K, V > **GetDictionaryReader< K, V >** (string property)
- T? GetInput< T > ()
- 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 Fixed←UpdateNetwork).
- LinkListReader< T > **GetLinkListReader< T >** (string property)
- int **GetLocalAuthorityMask** ()

  Gets a bitmask of AuthorityMasks flags, representing the current local authority over this NetworkObject.
- PropertyReader< T > GetPropertyReader< T > (string property)
- ref T ReinterpretState< T > (int offset=0)

  Allows read and write access to the internal state buffer.
- 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)

Public Member Functions inherited from SimulationBehaviour

- 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.

Public Member Functions inherited from Behaviour

- 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 ArrayReader< T > **GetArrayReader< T >** (Type behaviourType, string property)
- static BehaviourReader< T > GetBehaviourReader< T > (NetworkRunner runner, Type behaviourType, string property)
- static BehaviourReader< TProperty > GetBehaviourReader< TBehaviour, TProperty > (NetworkRunner runner, string property)
- static DictionaryReader< K, V > **GetDictionaryReader< K, V >** (Type behaviourType, string property)
- static LinkListReader< T > **GetLinkListReader< T >** (Type behaviourType, string property)
- static PropertyReader< T > GetPropertyReader< T > (Type behaviourType, string property)
- static PropertyReader< TProperty > GetPropertyReader< TBehaviour, TProperty > (string property)
- 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 > ()
- static T ∗ MakePtr< T > (T defaultValue)
- static ref T MakeRef< T > ()
- static ref T MakeRef< T > (T defaultValue)
- static int **NetworkDeserialize** (NetworkRunner runner, byte ∗data, ref NetworkBehaviour result)
- static int **NetworkSerialize** (NetworkRunner runner, NetworkBehaviour obj, byte ∗data)
- static NetworkBehaviour **NetworkUnwrap** (NetworkRunner runner, NetworkBehaviourId wrapper)
- static NetworkBehaviourId **NetworkWrap** (NetworkRunner runner, NetworkBehaviour obj)
- static implicit operator NetworkBehaviourId (NetworkBehaviour behaviour)

  Converts NetworkBehaviour to NetworkBehaviourId.

Static Public Member Functions inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

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)

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]`
- bool **StateBufferIsValid** `[get]`
- int int count **WordInfo** `[get]`

Properties inherited from SimulationBehaviour

- bool **CanReceiveRenderCallback** `[get]`
- bool **CanReceiveSimulationCallback** `[get]`
- NetworkObject **Object** `[get]`

  The NetworkObject this component is associated with.
- NetworkRunner **Runner** `[get]`

  The NetworkRunner this component is associated with.

6.55.1 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.

6.55.2 Member Function Documentation

6.55.2.1 CopyStateFrom()

```
void CopyStateFrom (
             NetworkBehaviour source )
```

Copies entire state of passed in source NetworkBehaviour

Parameters

source	Source NetworkBehaviour to copy data from

6.55.2.2 Despawned()

```
virtual void Despawned (
             NetworkRunner runner,
             bool hasState )  [virtual]
```

Called before the network object is despawned.

Parameters

hasState	If the state of the behaviour is still accessible

Reimplemented in HitboxRoot.

6.55.2.3 FixedUpdateNetwork()

```
override void FixedUpdateNetwork ( )  [virtual]
```

Fusion FixedUpdate timing callback.

Reimplemented from SimulationBehaviour.

6.55.2.4 GetBehaviourReader< T >() [1/2]

```
static BehaviourReader< T > GetBehaviourReader< T > (
             NetworkRunner runner,
             Type behaviourType,
             string property )  [static]
```

Type Constraints

T* : *NetworkBehaviour

6.55.2.5 GetBehaviourReader< T >() [2/2]

```
BehaviourReader< T > GetBehaviourReader< T > (
             string property )
```

Type Constraints

T* : *NetworkBehaviour

6.55.2.6 GetBehaviourReader< TBehaviour, TProperty >()

```
static BehaviourReader< TProperty > GetBehaviourReader< TBehaviour, TProperty > (
            NetworkRunner runner,
            string property )  [static]
```

Type Constraints

TBehaviour* : *NetworkBehaviour

TProperty* : *NetworkBehaviour

6.55.2.7 GetInput< T >() [1/2]

```
T? GetInput< T > ( )
```

Template Parameters

T	

Type Constraints

T* : *unmanaged

T* : *INetworkInput

6.55.2.8 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 Fixed←UpdateNetwork).

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

6.55.2.9 GetPropertyReader< T >() [1/2]

```
PropertyReader< T > GetPropertyReader< T > (
            string property )
```

Type Constraints

T* : *unmanaged

6.55.2.10 GetPropertyReader< T >() [2/2]

```
static PropertyReader< T > GetPropertyReader< T > (
             Type behaviourType,
             string property )  [static]
```

Type Constraints

T* : *unmanaged

6.55.2.11 GetPropertyReader< TBehaviour, TProperty >()

```
static PropertyReader< TProperty > GetPropertyReader< TBehaviour, TProperty > (
             string property )  [static]
```

Type Constraints

TBehaviour* : *NetworkBehaviour

TProperty* : *unmanaged

6.55.2.12 MakePtr< T >() [1/2]

```
static T * MakePtr< T > ( )  [static]
```

Type Constraints

T* : *unmanaged

6.55.2.13 MakePtr< T >() [2/2]

```
static T * MakePtr< T > (
             T defaultValue )  [static]
```

Type Constraints

T* : *unmanaged

6.55.2.14 MakeRef< T >() [1/2]

```
static ref T MakeRef< T > ( )  [static]
```

Type Constraints

T* : *unmanaged

6.55.2.15 MakeRef< T >() [2/2]

```
static ref T MakeRef< T > (
             T defaultValue )  [static]
```

Type Constraints

T* : *unmanaged

6.55.2.16 operator NetworkBehaviourId()

```
static implicit operator NetworkBehaviourId (
             NetworkBehaviour behaviour )  [static]
```

Converts NetworkBehaviour to NetworkBehaviourId.

Parameters

behaviour	

Returns

6.55.2.17 ReinterpretState< T >()

```
ref T ReinterpretState< T > (
             int offset = 0 )
```

Allows read and write access to the internal state buffer.

Parameters

offset	The offset to generate a ref for, in integer words

Template Parameters

T	

Returns

Reference to the location in memory defined by offset

Type Constraints

T* : *unmanaged

6.55.2.18 Spawned()

```
virtual void Spawned ( )  [virtual]
```

Post spawn callback.

Reimplemented in NetworkMecanimAnimator, and NetworkTransform.

6.56 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)
- Quaternion **Read** (NetworkBehaviour.PropertyReader< Quaternion > reader)
- Vector2 **Read** (NetworkBehaviour.PropertyReader< Vector2 > reader)
- Vector3 **Read** (NetworkBehaviour.PropertyReader< Vector3 > reader)
- Vector4 **Read** (NetworkBehaviour.PropertyReader< Vector4 > reader)
- T Read< T > (NetworkBehaviour.BehaviourReader< T > reader)
- T Read< T > (NetworkBehaviour.PropertyReader< T > reader)
- unsafe T ReinterpretState< T > (int offset=0)

Static Public Member Functions

- static implicit **operator bool** (NetworkBehaviourBuffer buffer)

Properties

- int **Length** `[get]`
- int **this[int index]** `[get]`
- Tick **Tick** `[get]`
- bool **Valid** `[get]`

6.56.1 Detailed Description

Provides low level accesss to data buffers that can be read using a NetworkBehaviour.Reader.

6.56.2 Member Function Documentation

6.56.2.1 Read< T >() [1/2]

```
T Read< T > (
             NetworkBehaviour::BehaviourReader< T > reader )
```

Type Constraints

T* : *NetworkBehaviour

6.56.2.2 Read< T >() [2/2]

```
T Read< T > (
            NetworkBehaviour::PropertyReader< T > reader )
```

Type Constraints

T* : *unmanaged

6.56.2.3 ReinterpretState< T >()

```
unsafe T ReinterpretState< T > (
            int offset = 0 )
```

Type Constraints

T* : *unmanaged

6.57 NetworkConfiguration Class Reference

Main network configuration class.

Public Types

- enum 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 **MtuDefault** `[get]`

  Max number of bytes that can be used by Fusion to fill up a UDP package.
- int **SocketRecvBufferSize** `[get]`

  Size in Kilobytes of the underlying socket receive buffer.
- int **SocketSendBufferSize** `[get]`

  Size in Kilobytes of the underlying socket send buffer.

6.57.1 Detailed Description

Main network configuration class.

6.57.2 Member Enumeration Documentation

6.57.2.1 ReliableDataTransfers

```
enum ReliableDataTransfers
```

Flag for allowed Reliable Data transfer modes.

Enumerator

ClientToServer	Allow Client to Server.
ClientToClientWithServerProxy	Allow Client to Client using Server as Proxy.

6.57.3 Member Function Documentation

6.57.3.1 Init()

```
NetworkConfiguration Init ( )
```

Initializes and creates a copy of this NetworkProjectConfig.

Returns

6.57.4 Member Data Documentation

6.57.4.1 ReliableDataTransferModes

```
ReliableDataTransfers ReliableDataTransferModes
```

Initial value:

```
=
     ReliableDataTransfers.ClientToServer |
     ReliableDataTransfers.ClientToClientWithServerProxy
```

Current ReliableDataTransferModes mode.

6.57.5 Property Documentation

6.57.5.1 ConnectionPingInterval

```
double ConnectionPingInterval  [get]
```

Interval in seconds between PING messages sent to a remote connection, in order to keep that connection alive.

Currently unused.

6.58 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.

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)
- 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** ()
- IEnumerator< KeyValuePair< K, V > > IEnumerable< KeyValuePair< K, V > >. **GetEnumerator** ()
- IEnumerator IEnumerable. **GetEnumerator** ()

- K **GetKey** (int entry)
- int **GetNxt** (int entry)
- V **GetVal** (int entry)
- int **Insert** (K key, V val)
- **NetworkDictionary** (int ∗data, int capacity, IElementReaderWriter< K > keyReaderWriter, IElement←ReaderWriter< V > valReaderWriter)
- 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** ()
- 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)

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
- 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.

6.58.1 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.

6.58.2 Member Function Documentation

6.58.2.1 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

value	The value to compare against.
equalityComparer	Specify custom IEqualityComparer to be used for compare.

6.58.2.2 Remove() `[1/2]`

```
bool Remove (
            K key )
```

Remove entry from Dictionary.

Parameters

key	

Returns

Returns true if key was found.

6.58.2.3 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

key	The key to remove.
value	Returns value of removed item. Returns default value if key did not exist.

Returns

Returns true if key was found.

6.58.2.4 TryGet()

```
bool TryGet (
            K key,
            out V value )
```

Attempts to get the value for a given key. If found, returns true.

Parameters

key	The key to remove.
value	Returns value of removed item. Returns default value if key did not exist.

Returns

Returns true if key was found.

6.59 NetworkedAttribute Class Reference

Inherits Attribute.

Public Member Functions

- **NetworkedAttribute** ()

  Default constructor for NetworkedAttribute.
- **NetworkedAttribute** (string group)

Properties

- string **Default** `[get, set]`

  Name of the field that holds the default value for this networked property.
- string **Group** `[get, set]`

6.59.1 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. OnChanged can be assigned with the name of a method in the same NetworkBehaviour. The named method will get called whenever this property value has been changed by the State Authority. `| [Networked(OnDataReceived = nameof(MyCallbackMethod)]`
`| public int MyProperty { get; set; } | | protected static void MyCallback↩ Method(Changed<ChangedCallbackParent> changed) { | changed.LoadNew(); | var newval = changed.Behaviour.MyProperty; | changed.LoadOld(); | var oldval = changed.Behaviour.MyProperty; | Debug.Log($¨Changed from {oldval} to {newval}¨); | }`

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 { } }`

6.60 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.

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

Public Member Functions inherited from Behaviour

- 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()

- 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)
- void **OnObjectEnterAOI** (NetworkRunner runner, NetworkObject obj, PlayerRef player)

- void **OnObjectExitAOI** (NetworkRunner runner, NetworkObject obj, PlayerRef player)
- 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)
- void **OnReliableDataReceived** (NetworkRunner runner, PlayerRef player, ReliableKey key, ArraySegment< byte > data)
- void **OnSceneLoadDone** (NetworkRunner runner)
- void **OnSceneLoadStart** (NetworkRunner runner)
- 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.

Static Public Member Functions inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

6.60.1 Detailed Description

Companion component for NetworkRunner. Exposes INetworkRunnerCallbacks as UnityEvents, which can be wired up to other components in the inspector.

6.61 NetworkId Struct Reference

The unique identifier for a network entity.

Inherits INetworkStruct, IEquatable< NetworkId >, IComparable, and IComparable< NetworkId >.

Public Member Functions

- int **CompareTo** (NetworkId other)
- int IComparable. **CompareTo** (object obj)
- bool **Equals** (NetworkId other)
- override bool **Equals** (object obj)
- override int **GetHashCode** ()
- string ToNamePrefixString ()

  String conversion specifically for use in prefixing names of GameObjects.
- override string **ToString** ()
- void **Write** (NetBitBuffer ∗buffer)

Static Public Member Functions

- static implicit **operator bool** (NetworkId id)
- static bool **operator!=** (NetworkId a, NetworkId b)
- static bool **operator==** (NetworkId a, NetworkId b)
- static NetworkId **Read** (NetBitBuffer ∗buffer)
- static void **Write** (NetBitBuffer ∗buffer, NetworkId id)

Public Attributes

- uint **Raw**

Static Public Attributes

- const int **ALIGNMENT** = 4
- const int **BLOCK_SIZE** = 8
- 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

Properties

- static EqualityComparer **Comparer** = new EqualityComparer() `[get]`
- bool **IsReserved** `[get]`
- bool **IsValid** `[get]`

6.61.1 Detailed Description

The unique identifier for a network entity.

6.61.2 Member Function Documentation

6.61.2.1 ToNamePrefixString()

```
string ToNamePrefixString ( )
```

String conversion specifically for use in prefixing names of GameObjects.

Returns

6.62 NetworkInput Struct Reference

Translates INetworkInput structs and represents them in Fusions's unsafe allocated memory.

Public Member Functions

- bool **Convert** (Type type)
- bool Convert< T > ()
- T Get< T > ()
- bool Is< T > ()
- **NetworkInput** (int *ptr, int wordCount)
- bool Set< T > (T value)
- 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]`
- bool **IsValid** `[get]`
- Type **Type** `[get]`
- bool **Valid** `[get]`
- int **WordCount** `[get]`

6.62.1 Detailed Description

Translates INetworkInput structs and represents them in Fusions's unsafe allocated memory.

6.62.2 Member Function Documentation

6.62.2.1 Convert< T >()

```
bool Convert< T > ( )
```

Type Constraints

T : unmanaged

T : INetworkInput

T : Convert

T : typeof

T : T

6.62.2.2 Get< T >()

```
T Get< T > ( )
```

Type Constraints

T : unmanaged

T : INetworkInput

6.62.2.3 Is< T >()

```
bool Is< T > ( )
```

Type Constraints

T **:** ***unmanaged***

T **:** ***INetworkInput***

6.62.2.4 Set< T >()

```
bool Set< T > (
             T value )
```

Type Constraints

T **:** ***unmanaged***

T **:** ***INetworkInput***

6.62.2.5 TryGet< T >()

```
bool TryGet< T > (
             out T input )
```

Tries to export data as the indicated T INetworkInput struct.

Type Constraints

T **:** ***unmanaged***

T **:** ***INetworkInput***

6.62.2.6 TrySet< T >()

```
bool TrySet< T > (
             T input )
```

Tries to import data from a INetworkInput struct.

Type Constraints

T **:** ***unmanaged***

T **:** ***INetworkInput***

6.63 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.

Public Member Functions

- void INetworkLinkedList. **Add** (object item)
- 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)
- Enumerator **GetEnumerator** ()
- 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 with this value. Returns -1 if not found.
- **NetworkLinkedList** (byte ∗data, int capacity, IElementReaderWriter< T > rw)
- T **Read** (int ∗entry)
- NetworkLinkedList< T > **Remap** (void ∗list)
- 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
- const int **HEAD** = 1
- const int **INVALID** = 0
- const int **META_WORDS** = 3
- 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.

6.63.1 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

T	T can be a primitive, or an INetworkStruct.

6.63.2 Member Function Documentation

6.63.2.1 Add()

```
void Add (
            T value )
```

Adds a value to the end of the list.

Parameters

value	

6.63.2.2 IndexOf()

```
int IndexOf (
            T value,
            IEqualityComparer< T > equalityComparer )
```

Returns the index with this value. Returns -1 if not found.

Parameters

equalityComparer	Specify custom IEqualityComparer to be used for compare.

6.64 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 Member Functions inherited from NetworkBehaviour

- virtual void **CopyBackingFieldsToState** (bool firstTime)
- void CopyStateFrom (NetworkBehaviour source)

  Copies entire state of passed in source NetworkBehaviour
- virtual void **CopyStateToBackingFields** ()
- virtual void Despawned (NetworkRunner runner, bool hasState)

  Called before the network object is despawned.
- override void FixedUpdateNetwork ()

  Fusion FixedUpdate timing callback.
- ArrayReader< T > **GetArrayReader< T >** (string property)
- BehaviourReader< T > GetBehaviourReader< T > (string property)
- ChangeDetector **GetChangeDetector** (ChangeDetector.Source source, bool copyInitial=true)
- DictionaryReader< K, V > **GetDictionaryReader< K, V >** (string property)
- T? GetInput< T > ()
- 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 Fixed←UpdateNetwork).
- LinkListReader< T > **GetLinkListReader< T >** (string property)
- int **GetLocalAuthorityMask** ()

  Gets a bitmask of AuthorityMasks flags, representing the current local authority over this NetworkObject.
- PropertyReader< T > GetPropertyReader< T > (string property)
- ref T ReinterpretState< T > (int offset=0)

  Allows read and write access to the internal state buffer.
- 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)

- 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.

Public Member Functions inherited from Behaviour

- 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()

- void AfterAllTicks (bool resimulation, int tickCount)

  Called after the resimulation loop (when applicable), and also after the forward simulation loop. Only called on Updates where resimulation or forward ticks are processed.

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.

Public Attributes inherited from NetworkBehaviour

- int **offset**

  Gives access to the offset (in 32 bit words) and count (in 32 bit words) of this behaviour backing data.

Properties

- override? int **DynamicWordCount** `[get]`

Properties inherited from NetworkBehaviour

- 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]`
- bool **StateBufferIsValid** `[get]`
- int int count **WordInfo** `[get]`

Properties inherited from SimulationBehaviour

- bool **CanReceiveRenderCallback** `[get]`
- bool **CanReceiveSimulationCallback** `[get]`
- NetworkObject **Object** `[get]`

  The NetworkObject this component is associated with.
- NetworkRunner **Runner** `[get]`

  The NetworkRunner this component is associated with.

Additional Inherited Members

Static Public Member Functions inherited from NetworkBehaviour

- static ArrayReader< T > **GetArrayReader< T >** (Type behaviourType, string property)
- static BehaviourReader< T > GetBehaviourReader< T > (NetworkRunner runner, Type behaviourType, string property)
- static BehaviourReader< TProperty > GetBehaviourReader< TBehaviour, TProperty > (NetworkRunner runner, string property)
- static DictionaryReader< K, V > **GetDictionaryReader< K, V >** (Type behaviourType, string property)
- static LinkListReader< T > **GetLinkListReader< T >** (Type behaviourType, string property)
- static PropertyReader< T > GetPropertyReader< T > (Type behaviourType, string property)
- static PropertyReader< TProperty > GetPropertyReader< TBehaviour, TProperty > (string property)
- 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 > ()
- static T ∗ MakePtr< T > (T defaultValue)
- static ref T MakeRef< T > ()
- static ref T MakeRef< T > (T defaultValue)
- static int **NetworkDeserialize** (NetworkRunner runner, byte ∗data, ref NetworkBehaviour result)
- static int **NetworkSerialize** (NetworkRunner runner, NetworkBehaviour obj, byte ∗data)
- static NetworkBehaviour **NetworkUnwrap** (NetworkRunner runner, NetworkBehaviourId wrapper)
- static NetworkBehaviourId **NetworkWrap** (NetworkRunner runner, NetworkBehaviour obj)
- static implicit operator NetworkBehaviourId (NetworkBehaviour behaviour)

  Converts NetworkBehaviour to NetworkBehaviourId.

Static Public Member Functions inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

Protected Member Functions inherited from NetworkBehaviour

- virtual bool **ReplicateTo** (PlayerRef player)

6.64.1 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.

6.64.2 Member Function Documentation

6.64.2.1 Render()

```
override 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 from SimulationBehaviour.

6.64.2.2 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.Set↩ Trigger() 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

triggerHash	
passThroughOnInputAuthority	Will call Animator.SetTrigger() immediately on the InputAuthority. If false, SetTrigger() will not be called on the Input Authority at all and Animator.SetTrigger() should be called explicitly as needed.

6.64.2.3 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.Set↩ Trigger() 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

trigger	
passThroughOnInputAuthority	Will call Animator.SetTrigger() immediately on the InputAuthority. If false, SetTrigger() will not be called on the Input Authority at all and Animator.SetTrigger() should be called explicitly as needed.

6.64.2.4 Spawned()

```
override void Spawned ( )  [virtual]
```

Post spawn callback.

Reimplemented from NetworkBehaviour.

6.65 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)
- void **ReleaseStateAuthoirty** ()

  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)
- 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.

Public Member Functions inherited from Behaviour

- 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 int GetWordCount (NetworkObject obj)

  Get the word count for a 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 (NetworkRunner runner, NetworkObject obj)

  Return the obj NetworkId.
- static implicit **operator NetworkId** (NetworkObject obj)

Static Public Member Functions inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

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.
- 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.
- 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** ()
- virtual void **OnDestroy** ()

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 **IsSceneObject** `[get]`

  Returns true if this network entity existed as part of a scene, rather than having been dynamically spawned.
- 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 **IsSpawnedPrefabNestedObject** `[get]`

  Returns true if this network entity is a spawned prefab's nested object, rather than being a scene object or a root prefab object.
- bool **IsSpawnedPrefabRoot** `[get]`

Returns true if this network entity is a spawned prefab's root, rather than being a scene object or a nested prefab 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, set]`

  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.

6.65.1 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.

6.65.2 Member Function Documentation

6.65.2.1 CopyStateFrom() `[1/2]`

```
void CopyStateFrom (
            NetworkObject source )
```

Copies the entire State from another NetworkObject

Parameters

source	NetworkObject to copy the State from

6.65.2.2 CopyStateFrom() `[2/2]`

```
void CopyStateFrom (
            NetworkObjectHeaderPtr source )
```

Copies the entire State from another NetworkObject based on the NetworkObjectHeaderPtr

Parameters

source	NetworkObjectHeaderPtr to copy the state from

6.65.2.3 GetWordCount()

```
static int GetWordCount (
            NetworkObject obj )  [static]
```

Get the word count for a NetworkObject

Parameters

obj	The object to get the word count from

Returns

Exceptions

Exception	

6.65.2.4 NetworkUnwrap()

```
static void NetworkUnwrap (
            NetworkRunner runner,
            NetworkId wrapper,
            ref NetworkObject result )  [static]
```

Return the NetworkObject reference on *result* that matches the provided NetworkId

Parameters

runner	The NetworkRunner that will be used to try to find a NetworkObject with ID equals to *wrapper*
wrapper	The NetworkId to be searched
result	The found NetworkObject. null if the provided NetworkId is not valid

6.65.2.5 NetworkWrap()

```
static NetworkId NetworkWrap (
            NetworkRunner runner,
            NetworkObject obj )  [static]
```

Return the *obj* NetworkId.

Parameters

runner	The NetworkRunner that *obj* is assigned to
obj	The NetworkObject to get the ID from

Returns

The NetworkId of the object. Default if the object is not alive (null or destroyed)

6.65.2.6 SetPlayerAlwaysInterested()

```
void SetPlayerAlwaysInterested (
            PlayerRef player,
            bool alwaysInterested )
```

Add or remove specific player interest in this NetworkObject. Only the NetworkObject State Authority can set interest.

SimulationConfig.ReplicationMode must be set to SimulationConfig.StateReplicationModes.EventualConsistency.

Parameters

player	
alwaysInterested	

6.66 NetworkObjectHeader Struct Reference

Network object header information for a NetworkObject.

Inherits INetworkStruct, and IEquatable< NetworkObjectHeader >.

Public Member Functions

- bool **Equals** (NetworkObjectHeader other)
- override bool **Equals** (object obj)
- override int **GetHashCode** ()
- override string **ToString** ()

Static Public Member Functions

- static int * **GetBehaviourChangedTickArray** (NetworkObjectHeader *header)
- static int * **GetDataPointer** (NetworkObjectHeader *header)
- static int **GetDataWordCount** (NetworkObjectHeader *header)
- static NetworkTRSPData * **GetMainNetworkTRSPData** (NetworkObjectHeader *header)
- static bool **HasMainNetworkTRSP** (NetworkObjectHeader *header)
- static bool **operator!=** (NetworkObjectHeader left, NetworkObjectHeader right)
- static bool **operator==** (NetworkObjectHeader left, NetworkObjectHeader right)

Public Attributes

- fixed int **_reserved** [10]
- short **BehaviourCount**
- NetworkObjectHeaderFlags **Flags**
- NetworkId **Id**
- PlayerRef **InputAuthority**
- NetworkObjectNestingKey **NestingKey**
- NetworkId **NestingRoot**
- PlayerRef **StateAuthority**
- NetworkObjectTypeId **Type**
- short **WordCount**

Static Public Attributes

- const int **PLAYER_DATA_WORD** = 36 / Allocator.REPLICATE_WORD_SIZE
- const int **SIZE** = 80
- const int **WORDS** = SIZE / Allocator.REPLICATE_WORD_SIZE

Properties

- int **ByteCount** `[get]`

  how many bytes this headers object is

6.66.1 Detailed Description

Network object header information for a NetworkObject.

6.67 NetworkObjectTypeId Struct Reference

ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable.

Inherits INetworkStruct, and IEquatable< NetworkObjectTypeId >.

Classes

- class EqualityComparer

  NetworkObjectTypeId Comparer

Public Member Functions

- bool **Equals** (NetworkObjectTypeId other)
- override bool **Equals** (object obj)
- override int **GetHashCode** ()
- override string **ToString** ()

Static Public Member Functions

- static NetworkObjectTypeId **FromCustom** (uint raw)
- static NetworkObjectTypeId **FromPrefabId** (NetworkPrefabId prefabId)
- static NetworkObjectTypeId **FromSceneRefAndObjectIndex** (SceneRef sceneRef, int objIndex, Network↩ SceneLoadId loadId=default)
- static NetworkObjectTypeId **FromStruct** (ushort structId)
- static implicit **operator NetworkObjectTypeId** (NetworkPrefabId prefabId)
- static bool **operator!=** (NetworkObjectTypeId a, NetworkObjectTypeId b)
- static bool **operator==** (NetworkObjectTypeId a, NetworkObjectTypeId b)

Public Attributes

- uint **_value0**
- uint **_value1**

Static Public Attributes

- const int **ALIGNMENT** = 4
- const int **MAX_SCENE_OBJECT_INDEX** = (1 << SCENE_OBJECT_INDEX_BITS) - 1
- const int **SIZE** = 8
- const ushort **STRUCT_TYPE_PLAYERDATA** = 1

Properties

- uint **AsCustom** `[get]`
- ushort **AsInternalStructId** `[get]`
- NetworkPrefabId **AsPrefabId** `[get]`
- NetworkSceneObjectId **AsSceneObjectId** `[get]`
- static EqualityComparer **Comparer** = new EqualityComparer() `[get]`
- bool **IsCustom** `[get]`
- bool **IsNone** `[get]`
- bool **IsPrefab** `[get]`
- bool **IsSceneObject** `[get]`
- bool **IsStruct** `[get]`
- bool **IsValid** `[get]`
- NetworkTypeIdKind **Kind** `[get]`
- static NetworkObjectTypeId **PlayerData** `[get]`

6.67.1 Detailed Description

ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable.

6.68 NetworkObjectTypeId.EqualityComparer Class Reference

NetworkObjectTypeId Comparer

Inherits IEqualityComparer< NetworkObjectTypeId >.

Public Member Functions

- bool **Equals** (NetworkObjectTypeId x, NetworkObjectTypeId y)
- int **GetHashCode** (NetworkObjectTypeId obj)

6.68.1 Detailed Description

NetworkObjectTypeId Comparer

6.69 NetworkPositionRotation Class Reference

Use NetworkTransform (or any custom class derived from NetworkTRSP) to synchronize initial transform values. This component is non-functional.

Inherits Behaviour.

Additional Inherited Members

Public Member Functions inherited from Behaviour

- 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 inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

6.69.1 Detailed Description

Use NetworkTransform (or any custom class derived from NetworkTRSP) to synchronize initial transform values. This component is non-functional.

6.70 NetworkPrefabId Struct Reference

ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable.

Inherits INetworkStruct, IEquatable< NetworkPrefabId >, IComparable, and IComparable< NetworkPrefabId >.

Public Member Functions

- int **CompareTo** (NetworkPrefabId other)
- int IComparable. **CompareTo** (object obj)
- bool **Equals** (NetworkPrefabId other)
- override bool **Equals** (object obj)
- override int **GetHashCode** ()
- override string **ToString** ()
- string **ToString** (bool brackets, bool prefix)

Static Public Member Functions

- static NetworkPrefabId **FromIndex** (int index)
- static NetworkPrefabId **FromRaw** (uint value)
- static bool **operator!=** (NetworkPrefabId a, NetworkPrefabId b)
- static bool **operator==** (NetworkPrefabId a, NetworkPrefabId b)

Public Attributes

- uint **RawValue**

Static Public Attributes

- const int **ALIGNMENT** = 4
- const int **MAX_INDEX** = int.MaxValue - 1
- const int **SIZE** = 4

Properties

- int **AsIndex** `[get]`
- bool **IsNone** `[get]`
- bool **IsValid** `[get]`

6.70.1 Detailed Description

ID for a NetworkObject Prefab which has been cataloged in a NetworkProjectConfig.PrefabTable.

6.71 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.TryAdd(NetworkObjectGuid, INetworkPrefabSource, out NetworkPrefabId) 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.

6.71.1 Detailed Description

Meta data for a NetworkObject prefab which has been cataloged in a NetworkProjectConfig.PrefabTable.

6.72 NetworkPrefabRef Struct Reference

A decoupled NetworkObject prefab reference. Internally stored as a GUID.

Inherits INetworkStruct, IEquatable< NetworkPrefabRef >, and IComparable< NetworkPrefabRef >.

Public Member Functions

- int **CompareTo** (NetworkPrefabRef other)
- bool **Equals** (NetworkPrefabRef other)
- override bool **Equals** (object obj)
- override int **GetHashCode** ()
- **NetworkPrefabRef** (byte ∗guid)
- **NetworkPrefabRef** (byte[ ] guid)
- **NetworkPrefabRef** (long data0, long data1)
- **NetworkPrefabRef** (string guid)
- override string **ToString** ()
- string **ToString** (string format)
- string **ToUnityGuidString** ()

Static Public Member Functions

- static implicit **operator Guid** (NetworkPrefabRef guid)
- static **operator NetworkObjectGuid** (NetworkPrefabRef t)
- static implicit **operator NetworkPrefabRef** (Guid guid)
- static bool **operator!=** (NetworkPrefabRef a, NetworkPrefabRef b)
- static bool **operator==** (NetworkPrefabRef a, NetworkPrefabRef b)
- static NetworkPrefabRef **Parse** (string str)
- static bool **TryParse** (string str, out NetworkPrefabRef guid)

Public Attributes

- fixed long **RawGuidValue** [2]

Static Public Attributes

- const int **ALIGNMENT** = 4
- const int **SIZE** = 16

Properties

- static NetworkPrefabRef **Empty** `[get]`
- bool **IsValid** `[get]`

6.72.1 Detailed Description

A decoupled NetworkObject prefab reference. Internally stored as a GUID.

6.73 NetworkProjectConfig Class Reference

The core Fusion config file that is shared with all peers at startup.

Public Types

- enum 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 ReplicationFeatures

Public Member Functions

- delegate NetworkProjectConfigAsset **AssetLoadingDelegate** ()

  GlobalAssetLoading
- delegate void **AssetUnloadingDelegate** (NetworkProjectConfigAsset asset)

  GlobalAssetUnloading
- int? **GetExecutionOrder** (Type type)
- 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.
- bool **EnqueueIncompleteSynchronousSpawns**

  This flag changes the behaviour of NetworkRunner.Spawn to return null (instead of throwing an exception) and NetworkRunner.TrySpawn) 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**

  Signal if the NetworkId of the NetworkObject should be included on the name of the GameObject.
- bool **NullChecksForNetworkedProperties** = true

  If set, the weaver will add a check to all [Networked] properties on each NetworkBehaviour to verify if owing Network↩ Object 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 **TypeId** = CurrentTypeId

  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.Runntime.dll.
- const string **CurrentTypeId** = 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 NetworkRunner.System.Diagnostics.FileVersionInfo **FusionVersionInfo** `[get]`
- 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.
- static AssetLoadingDelegate **GlobalAssetLoading**

  Invoked when a config is a about to be loaded from a default location (a Resource DefaultResourceName). If the event returns a non-null value, it will accepted as the config source and no attempt to load the default asset will be made.
- static AssetUnloadingDelegate **GlobalAssetUnloading**

  Invoked when a config is about to be unloaded (due to UnloadGlobal).

6.73.1 Detailed Description

The core Fusion config file that is shared with all peers at startup.

6.73.2 Member Enumeration Documentation

6.73.2.1 PeerModes

```
enum 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.

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.

6.73.2.2 ReplicationFeatures

```
enum ReplicationFeatures
```

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).

6.73.3 Member Function Documentation

6.73.3.1 Deserialize()

```
static NetworkProjectConfig Deserialize (
            string data )  [static]
```

De-serialize a NetworkProjectConfig from a JSON string (typically sent by the Room's Creator).

Parameters

data	JSON string of a serialized NetworkProjectConfig

Returns

NetworkProjectConfig reference de-serialized from JSON string

6.73.3.2 Serialize()

```
static string Serialize (
            NetworkProjectConfig config )  [static]
```

Serialize a NetworkProjectConfig into a JSON string.

Parameters

config	NetworkProjectConfig reference

Returns

JSON String

6.73.4 Member Data Documentation

6.73.4.1 AssembliesToWeave

```
string [] AssembliesToWeave
```

Initial value:
```
= new string[] {
      "Fusion.Unity",
      "Assembly-CSharp",
      "Assembly-CSharp-firstpass",
      "Fusion.UnityPhysics",
    }
```

Names of assemblies Fusion is going to weave. Not case sensitive.

6.74 NetworkProjectConfigAsset Class Reference

Manages and references the current instance of NetworkProjectConfig

Inherits FusionGlobalScriptableObject< T >.

Static Public Member Functions

- static bool **TryGetGlobal** (out NetworkProjectConfigAsset global)
- static void **UnloadGlobal** ()

Public Attributes

- SerializableSimulationBehaviourMeta[ ] **BehaviourMeta** = Array.Empty<SerializableSimulationBehaviour↩ Meta>()

  An auto-generated list containing meta information about all the SimulationBehaviours in the project, e.g. execution order.
- NetworkProjectConfig **Config** = new NetworkProjectConfig()
- NetworkPrefabTableOptions **PrefabOptions** = NetworkPrefabTableOptions.Default
- 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 registered at runtime with NetworkPrefabTable.TryAdd.

Properties

- static NetworkProjectConfigAsset **Global** `[get]`
- static bool **IsGlobalLoaded** `[get]`

6.74.1 Detailed Description

Manages and references the current instance of NetworkProjectConfig

6.75 NetworkRigidbody Class Reference

Use the Fusion Unity Physics Add-on, or your own variation of it to synchronize Rigidbodies. This component is non-functional.

Inherits Behaviour.

Additional Inherited Members

Public Member Functions inherited from Behaviour

- 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 inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

    Wrapper for Unity's GameObject.Destroy()

6.75.1 Detailed Description

Use the Fusion Unity Physics Add-on, or your own variation of it to synchronize Rigidbodies. This component is non-functional.

6.76 NetworkRigidbody2D Class Reference

Use the Fusion Unity Physics Add-on, or your own variation of it to synchronize Rigidbodies. This component is non-functional.

Inherits Behaviour.

Additional Inherited Members

Public Member Functions inherited from Behaviour

- 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 inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

6.76.1 Detailed Description

Use the Fusion Unity Physics Add-on, or your own variation of it to synchronize Rigidbodies. This component is non-functional.

6.77 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 BuildTypes

  Enumeration of Fusion.Runtime.dll options.
- enum 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 obj, PlayerRef? inputAuthority=null, bool allocate=true, bool? masterClient↩ ObjectOverride=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)

  Disconnect a player from the server.
- bool **EnsureRunnerSceneIsActive** (out Scene previousActiveScene)
- bool **Exists** (NetworkId id)

  Returns if the Fusion.Simulation contains a NetworkObject with given id in the current State SimulationSnapshot.
- bool **Exists** (NetworkObject obj)

  Returns if the Fusion.Simulation contains a reference to a NetworkObject in the current State SimulationSnapshot.
- NetworkObject FindObject (NetworkId oref)

  Get the NetworkObject instance for this NetworkRunner from a NetworkId.
- SimulationBehaviour[ ] GetAllBehaviours (Type type)

  Returns array of all SimulationBehaviour 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.
- void GetAreaOfInterestGizmoData (List<(Vector3 center, Vector3 size, int playerCount, int objectCount)> result)

  Clears the passed results collection, and adds all current AOI cell data. 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.

- 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.Is←Server)
- 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 gameObject)
- 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 IsPlayerActive (PlayerRef player)
- bool IsPlayerValid (PlayerRef player)

- async Task< StartGameResult > JoinSessionLobby (SessionLobby sessionLobby, string lobbyID=null, AuthenticationValues authentication=null, FusionAppSettings customAppSettings=null, bool? useDefault←CloudPorts=false, CancellationToken cancellationToken=default, bool useCachedRegions=false)

  Join the Peer to a specific Lobby, either a prebuild or a custom one.
- NetworkSceneAsyncOp **LoadScene** (SceneRef sceneRef, LoadSceneMode loadSceneMode=Load←SceneMode.Single, LocalPhysicsMode localPhysicsMode=LocalPhysicsMode.None, bool setActiveOn←Load=DefaultSetActiveOnLoad)
- NetworkSceneAsyncOp **LoadScene** (SceneRef sceneRef, LoadSceneParameters parameters, bool set←ActiveOnLoad=DefaultSetActiveOnLoad)
- NetworkSceneAsyncOp **LoadScene** (string sceneName, LoadSceneMode loadSceneMode=Load←SceneMode.Single, LocalPhysicsMode localPhysicsMode=LocalPhysicsMode.None, bool setActiveOn←Load=DefaultSetActiveOnLoad)
- NetworkSceneAsyncOp **LoadScene** (string sceneName, LoadSceneParameters parameters, bool set←ActiveOnLoad=DefaultSetActiveOnLoad)
- void **MakeDontDestroyOnLoad** (GameObject obj)
- bool **MoveGameObjectToSameScene** (GameObject gameObject, GameObject other)
- bool **MoveGameObjectToScene** (GameObject gameObject, SceneRef sceneRef)
- 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.
- 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 **RemoveSimulationBehavior** (SimulationBehaviour behaviour)

  Unregister a SimulationBehaviour instance from the SimulationBehaviourUpdater callbacks. Invalid if NetworkRunner has not been started and initialized.
- void **RenderInternal** ()

  This method is meant to be called by INetworkRunnerUpdater.
- void SendReliableDataToPlayer (PlayerRef player, ReliableKey key, byte[ ] data)

  Send an arbitrary data buffer to a target Player.
- void SendReliableDataToServer (ReliableKey key, byte[ ] data)

  Send an arbitrary 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.
- bool SetIsSimulated (NetworkObject obj, bool simulate)

  Sets the simulation state for this object, if it takes part in the NetworkFixedUpdate, etc.

- 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.
- 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, Quaternion? rotation, PlayerRef? input←Authority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? masterClientObjectOverride=null)
- 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, Quaternion? rotation, PlayerRef? input←Authority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? masterClientObjectOverride=null)
- 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.
- NetworkObject **Spawn** (NetworkObjectGuid prefabGuid, Vector3? position, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? masterClientObjectOverride=null)
- 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.
- NetworkObject **Spawn** (NetworkPrefabId typeId, Vector3? position, Quaternion? rotation, PlayerRef? input←Authority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? masterClientObjectOverride=null)
- 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.
- NetworkObject **Spawn** (NetworkPrefabRef prefabRef, Vector3? position, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? masterClientObjectOverride=null)
- 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

- T Spawn< T > (T prefab, Vector3? position, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? master←ClientObjectOverride=null)
- 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, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? masterClientObjectOverride=null, NetworkObjectSpawnDelegate onCompleted=null)
- 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, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroy←OnLoad=false, bool? masterClientObjectOverride=null, NetworkObjectSpawnDelegate onCompleted=null)
- 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.
- NetworkSpawnOp **SpawnAsync** (NetworkObjectGuid prefabGuid, Vector3? position, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOn←Load=false, bool? masterClientObjectOverride=null, NetworkObjectSpawnDelegate onCompleted=null)
- 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.
- NetworkSpawnOp **SpawnAsync** (NetworkPrefabId typeId, Vector3? position, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroy←OnLoad=false, bool? masterClientObjectOverride=null, NetworkObjectSpawnDelegate onCompleted=null)
- 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.
- NetworkSpawnOp **SpawnAsync** (NetworkPrefabRef prefabRef, Vector3? position, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOn←Load=false, bool? masterClientObjectOverride=null, NetworkObjectSpawnDelegate onCompleted=null)
- 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.
- NetworkSpawnOp SpawnAsync< T > (T prefab, Vector3? position, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? masterClientObjectOverride=null, NetworkObjectSpawnDelegate onCompleted=null)
- 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 bref, 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 TryGetBehaviourStats (List<(Type, BehaviourStats)> result)

    Populate the provided list with all registered behaviours and their BehaviourStats.
- bool TryGetInputForPlayer< T > (PlayerRef player, out T input)

    Outputs the NetworkInput from player, translated to the indicated INetworkInput.
- T TryGetNetworkedBehaviourFromNetworkedObjectRef< T > (NetworkId id)

    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 TryGetObjectStats (NetworkId id, out NetworkObjectStats stats)

    Try to get NetworkObjectStats buffer for a NetworkObject from this NetworkRunner.
- bool **TryGetPhysicsInfo** (out NetworkPhysicsInfo info)
- bool TryGetPlayerObject (PlayerRef player, out NetworkObject networkObject)

    Try to gets the NetworkObject associated with a specific player.
- bool TryGetPlayerStats (PlayerRef player, out SimulationConnectionStats stats)

    Try to get SimulationConnectionStats buffer for a player reference from this NetworkRunner.
- bool TryGetSceneInfo (out NetworkSceneInfo sceneInfo)

    Tries to get the NetworkSceneInfo of this NetworkRunner.
- bool TryGetSimulationStats (out SimulationStats stats)

    Try to get SimulationStats buffer from this NetworkRunner.
- bool **TrySetPhysicsInfo** (NetworkPhysicsInfo info)
- NetworkSpawnStatus **TrySpawn** (GameObject prefab, out NetworkObject obj, Vector3? position, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? masterClientObjectOverride=null)
- 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, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? masterClientObjectOverride=null)
- 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.
- NetworkSpawnStatus **TrySpawn** (NetworkObjectGuid prefabGuid, out NetworkObject obj, Vector3? position, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? masterClientObjectOverride=null)
- NetworkSpawnStatus TrySpawn (NetworkObjectGuid prefabGuid, out NetworkObject obj, Vector3? position=null, Quaternion? rotation=null, PlayerRef? inputAuthority=null, OnBefore←Spawned=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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

- NetworkSpawnStatus **TrySpawn** (NetworkPrefabId typeId, out NetworkObject obj, Vector3? position, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? masterClientObjectOverride=null)
- 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.
- NetworkSpawnStatus **TrySpawn** (NetworkPrefabRef prefabRef, out NetworkObject obj, Vector3? position, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOnLoad=false, bool? masterClientObjectOverride=null)
- NetworkSpawnStatus TrySpawn (NetworkPrefabRef prefabRef, out NetworkObject obj, Vector3? position=null, Quaternion? rotation=null, PlayerRef? inputAuthority=null, OnBeforeSpawned onBefore↩Spawned=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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.
- NetworkSpawnStatus TrySpawn< T > (T prefab, out T obj, Vector3? position, Quaternion? rotation, PlayerRef? inputAuthority, OnBeforeSpawned onBeforeSpawned, bool syncPhysics, bool dontDestroyOn↩Load=false, bool? masterClientObjectOverride=null)
- 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)
- void **UpdateInternal** (double dt)

  This method is meant to be called by INetworkRunnerUpdater.

Public Member Functions inherited from Behaviour

- 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 List< NetworkRunner >.Enumerator GetInstancesEnumerator ()

  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.

Static Public Member Functions inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

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]`
- NetworkProjectConfig **Config** `[get]`

  Returns the NetworkProjectConfig reference.
- ConnectionType **CurrentConnectionType** `[get]`

  Check the current Connection Type with the Remote Server.
- SceneRef **CurrentScene** `[get]`

  Returns the current loaded network scene.
- float **DeltaTime** `[get]`

  Returns the fixed tick time interval. Derived from the SimulationConfig.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 **IsSceneMaster** `[get]`

  Is this runner responsible for scene management.
- 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. Player↩ Refs 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.
- 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 SimulationSnapshot 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]`
- 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.

6.77.1 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.

6.77.2 Member Enumeration Documentation

6.77.2.1 BuildTypes

`enum BuildTypes`

Enumeration of Fusion.Runtime.dll options.

Enumerator

Debug	Use the Debug version of the Fusion.Runntime.dll.
Release	Use the Debug version of the Fusion.Runntime.dll.

6.77.2.2 States

`enum States`

Initialization stages of Fusion.

Enumerator

Starting	Runner is about to start.
Running	Runner is running.
Shutdown	Runner is shutdown.

6.77.3 Member Function Documentation

6.77.3.1 AddCallbacks()

```
void AddCallbacks (
            params INetworkRunnerCallbacks[] callbacks )
```

Register an INetworkRunnerCallbacks instance for callbacks from this NetworkRunner.

Parameters

callbacks	

6.77.3.2 Attach()

```
void Attach (
            NetworkObject obj,
            PlayerRef? inputAuthority = null,
            bool allocate = true,
            bool? masterClientObjectOverride = null )
```

Attaches a user created network object to the network.

Parameters

obj	The object to attach
inputAuthority	If assigned who is the default input authority for this object

6.77.3.3 Despawn()

```
void Despawn (
            NetworkObject networkObject )
```

Destroys a NetworkObject.

Parameters

networkObject	

6.77.3.4 DestroySingleton< T >()

```
void DestroySingleton< T > ( )
```

Removes a specific SimulationBehaviour from this NetworkRunner gameobject, if it exists.

Type Constraints

T : SimulationBehaviour

6.77.3.5 Disconnect()

```
void Disconnect (
             PlayerRef player,
             byte[ ] token = null )
```

Disconnect a player from the server.

Parameters

player	Player to disconnect

6.77.3.6 FindObject()

```
NetworkObject FindObject (
             NetworkId oref )
```

Get the NetworkObject instance for this NetworkRunner from a NetworkId.

Parameters

oref	

Returns

null if object cannot be found.

6.77.3.7 GetAllBehaviours()

```
SimulationBehaviour[ ] GetAllBehaviours (
             Type type )
```

Returns array of all SimulationBehaviour registered with this NetworkRunner.

Parameters

type	

Returns

6.77.3.8 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

T	SimulationBehaviour type

Returns

The result list

Type Constraints

T* : *SimulationBehaviour

6.77.3.9 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

result	The list to add the behaviours

Template Parameters

T	SimulationBehaviour type

Type Constraints

T* : *SimulationBehaviour

6.77.3.10 GetAreaOfInterestGizmoData()

```
void GetAreaOfInterestGizmoData (
            List<(Vector3 center, Vector3 size, int playerCount, int objectCount)> result )
```

Clears the passed results collection, and adds all current AOI cell data. Each element in the List represents one AOI cell.

Parameters

result	

6.77.3.11 GetInputForPlayer< T >()

```
T? GetInputForPlayer< T > (
            PlayerRef player )
```

Returns the NetworkInput data from player, converted to the indicated INetworkInput.

Type Constraints

> ***T : unmanaged***
>
> ***T : INetworkInput***

6.77.3.12 GetInstancesEnumerator()

```
static List< NetworkRunner >.Enumerator GetInstancesEnumerator ( )  [static]
```

Get enumerator for the collection of all NetworkRunners. Allows to enumerate alloc-free.

Returns

6.77.3.13 GetInterfaceListHead()

```
SimulationBehaviourListScope GetInterfaceListHead (
            Type type,
            int index,
            out SimulationBehaviour head )
```

Get the interface list head.

Parameters

type	The interface type
index	The desired index on the list of behaviourList
head	The head reference

Returns

> A disposable SimulationBehaviourListScope to be used on an `using` scope

6.77.3.14 GetInterfaceListNext()

```
SimulationBehaviour GetInterfaceListNext (
            SimulationBehaviour behaviour )
```

Get the next behaviour.

Parameters

behaviour	The reference behaviour to get the next one

Returns

6.77.3.15 GetInterfaceListPrev()

```
SimulationBehaviour GetInterfaceListPrev (
            SimulationBehaviour behaviour )
```

Get the previous behaviour.

Parameters

behaviour	The reference behaviour to get the previous one

Returns

6.77.3.16 GetInterfaceListsCount()

```
int GetInterfaceListsCount (
            Type type )
```

Get the number of interfaces of the desired type that are registered on the behaviour updater.

Parameters

type	The interface type

Returns

6.77.3.17 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

player	PlayerRef to get the Actor Number (ID)

Returns

Actor Number associated with the PlayerRef, otherwise null.

6.77.3.18 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

player	PlayerRef to check for a Connection Token

Returns

Copy of the Connection Token

6.77.3.19 GetPlayerConnectionType()

```
ConnectionType GetPlayerConnectionType (
            PlayerRef player )
```

Return the ConnectionType with a Remote PlayerRef. Valid only when invoked from a Server (NetworkRunner.Is←
Server)

Parameters

player	Remote Player to check the ConnectionType

Returns

ConnectionType with a PlayerRef

6.77.3.20 GetPlayerObject()

```
NetworkObject GetPlayerObject (
            PlayerRef player )
```

Gets the network object associated with a specific player.

Parameters

player	

Returns

Network object if one is associated with the player

6.77.3.21 GetPlayerRtt()

```
double GetPlayerRtt (
            PlayerRef playerRef )
```

Returns the player round trip time (ping) in seconds.

Parameters

playerRef	The player you want the round trip time for

6.77.3.22 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

player	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.

6.77.3.23 GetResumeSnapshotNetworkObjects()

```
IEnumerable< NetworkObject > GetResumeSnapshotNetworkObjects ( )
```

Iterate over the old NetworkObjects from the Resume Snapshot.

Returns

Iterable list of NetworkObject

6.77.3.24 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

6.77.3.25 GetRunnerForGameObject()

```
static NetworkRunner GetRunnerForGameObject (
             GameObject gameObject )  [static]
```

Get the NetworkRunner a GameObject instance belongs to.

Parameters

gameObject	GameObject to check for a NetworkRunner

Returns

NetworkRunner reference, or null if not found

6.77.3.26 GetRunnerForScene()

```
static NetworkRunner GetRunnerForScene (
             Scene scene )  [static]
```

Get the NetworkRunner from a specific Scene.

Parameters

scene	Scene to check for a NetworkRunner

Returns

NetworkRunner reference, or null if not found

6.77.3.27 GetSingleton< T >()

```
T GetSingleton< T > ( )
```

Ensures that a specific SimulationBehaviour component exists on this NetworkRunner gameobject.

Type Constraints

T* : *SimulationBehaviour

6.77.3.28 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

6.77.3.29 InstantiateInRunnerScene< T >() [1/2]

```
T InstantiateInRunnerScene< T > (
            T original )
```

Instantiates an object in the scene of this runner.

Type Constraints

T : Component

6.77.3.30 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

6.77.3.31 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)

6.77.3.32 IsPlayerActive()

```
bool IsPlayerActive (
            PlayerRef player )
```

Parameters

player	

Returns

6.77.3.33 IsPlayerValid()

```
bool IsPlayerValid (
            PlayerRef player )
```

Parameters

player	

Returns

6.77.3.34 JoinSessionLobby()

```
async Task< StartGameResult > JoinSessionLobby (
            SessionLobby sessionLobby,
            string lobbyID = null,
            AuthenticationValues authentication = null,
            FusionAppSettings customAppSettings = null,
            bool? useDefaultCloudPorts = false,
            CancellationToken cancellationToken = default,
            bool useCachedRegions = false )
```

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

sessionLobby	Lobby Type to Join
lobbyID	Lobby ID
authentication	Authentication Values used to authenticate this peer
customAppSettings	Custom Photon Application Settings
useDefaultCloudPorts	Signal if the LoadBalancingClient should use the Default or Alternative Ports
cancellationToken	Optional Cancellation Token
useCachedRegions	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.

6.77.3.35 MoveToRunnerScene()

```
void MoveToRunnerScene (
            GameObject instance,
            SceneRef?  targetSceneRef = null )
```

Moves an object to the scene of this runner.

Parameters

instance	
targetSceneRef	

6.77.3.36 MoveToRunnerScene< T >()

```
void MoveToRunnerScene< T > (
            T component )
```

Moves an object to the scene of this runner.

Template Parameters

T	

Parameters

component	Component of object to move

Type Constraints

T* : *Component

6.77.3.37 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.

6.77.3.38 RegisterSceneObjects()

```
int RegisterSceneObjects (
            SceneRef scene,
            NetworkObject[] objects,
            NetworkSceneLoadId loadId = default )
```

Registers.

Parameters

scene	
objects	

Exceptions

ArgumentException	
ArgumentNullException	

6.77.3.39 RemoveCallbacks()

```
void RemoveCallbacks (
            params INetworkRunnerCallbacks[] callbacks )
```

Unregister an INetworkRunnerCallbacks instance for callbacks from this NetworkRunner.

Parameters

callbacks	

6.77.3.40 SendReliableDataToPlayer()

```
void SendReliableDataToPlayer (
            PlayerRef player,
            ReliableKey key,
            byte[] data )
```

Send an arbitrary data buffer to a target Player.

Parameters

player	Player that should receive the buffer
data	Buffer to be sent

6.77.3.41 SendReliableDataToServer()

```
void SendReliableDataToServer (
            ReliableKey key,
            byte[] data )
```

Send an arbitrary data buffer to the Server.

Parameters

data	Buffer to be sent

6.77.3.42 SendRpc() [1/2]

```
void SendRpc (
             SimulationMessage * message )
```

Sends RPC message. Not meant to be used directly, ILWeaver calls this.

Parameters

message	

6.77.3.43 SendRpc() [2/2]

```
void SendRpc (
             SimulationMessage * message,
             out RpcSendResult info )
```

Sends RPC message. Not meant to be used directly, ILWeaver calls this.

Parameters

message	
info	

6.77.3.44 SetAreaOfInterestCellSize()

```
void SetAreaOfInterestCellSize (
             int size )
```

Set the area of interest cell size.

Parameters

size	

Exceptions

Exception	Can't change cell size in shared mode

6.77.3.45 SetAreaOfInterestGrid()

```
void SetAreaOfInterestGrid (
             int x,
             int y,
             int z )
```

Set the area of interest grid dimensions.

Parameters

x	
y	
z	

Exceptions

Exception	Can't change grid size in shared mode

6.77.3.46 SetIsSimulated()

```
bool SetIsSimulated (
            NetworkObject obj,
            bool simulate )
```

Sets the simulation state for this object, if it takes part in the NetworkFixedUpdate, etc.

Parameters

obj	the object to change state for
simulate	true if it should be simulated, false if otherwise

Returns

true if the state of the object changed, false otherwise

6.77.3.47 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

player	The player
networkObject	The object
alwaysInterested	If he's always interested, or not.

6.77.3.48 SetPlayerObject()

```
void SetPlayerObject (
            PlayerRef player,
            NetworkObject networkObject )
```

Sets the network object associated with this player.

Parameters

player	
networkObject	

6.77.3.49 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

prefab	A GameObject with a NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

Returns

NetworkObject reference, or null if it was not able to spawn the object

6.77.3.50 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 NetworkPosition↵ Rotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

Parameters

prefab	Prefab used to spawn the NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

Returns

NetworkObject reference, or null if it was not able to spawn the object

6.77.3.51 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

Parameters

prefabGuid	Object Guid used to spawn the NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

Returns

NetworkObject reference, or null if it was not able to spawn the object

6.77.3.52 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

Parameters

typeId	Prefab ID used to spawn the NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

Returns

NetworkObject reference, or null if it was not able to spawn the object

6.77.3.53 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

Parameters

prefabRef	Prefab Ref used to spawn the NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

Returns

NetworkObject reference, or null if it was not able to spawn the object

6.77.3.54 Spawn< T >() [1/2]

```
T Spawn< T > (
            T prefab,
            Vector3?  position,
            Quaternion?  rotation,
            PlayerRef?  inputAuthority,
            OnBeforeSpawned onBeforeSpawned,
            bool syncPhysics,
            bool dontDestroyOnLoad = false,
            bool?  masterClientObjectOverride = null )
```

Type Constraints

T* : *SimulationBehaviour

6.77.3.55 Spawn< T >() [2/2]

```
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

T	Must be a Type derived from SimulationBehaviour

Parameters

prefab	SimulationBehaviour used to spawn the NetworkObject

Returns

T reference, or null if it was not able to spawn the object

¨), <param name=¨position¨>Spawn Position</param> <param name=¨rotation¨>Spawn Rotation</param> <param name=¨inputAuthority¨>Player Input Authority</param> <param name=¨onBeforeSpawned¨><see cref=¨OnBeforeSpawned¨/> reference</param> <param name=¨flags">Spawn flags

Type Constraints

T* : *SimulationBehaviour

6.77.3.56 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

prefab	A GameObject with a NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

,

Parameters

onCompleted	A callback to fire once the spawn is done.

6.77.3.57 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 NetworkPosition↩Rotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

Parameters

prefab	Prefab used to spawn the NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

,

Parameters

onCompleted	A callback to fire once the spawn is done.

6.77.3.58 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

Parameters

prefabGuid	Object Guid used to spawn the NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

,

Parameters

onCompleted	A callback to fire once the spawn is done.

6.77.3.59 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

Parameters

typeId	Prefab ID used to spawn the NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

,

Parameters

onCompleted	A callback to fire once the spawn is done.

6.77.3.60 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

Parameters

prefabRef	Prefab Ref used to spawn the NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

,

Parameters

onCompleted	A callback to fire once the spawn is done.

6.77.3.61 SpawnAsync< T >() [1/2]

```
NetworkSpawnOp SpawnAsync< T > (
            T prefab,
            Vector3?  position,
            Quaternion?  rotation,
            PlayerRef?  inputAuthority,
            OnBeforeSpawned onBeforeSpawned,
            bool syncPhysics,
            bool dontDestroyOnLoad = false,
            bool?  masterClientObjectOverride = null,
            NetworkObjectSpawnDelegate onCompleted = null )
```

Type Constraints

T* : *SimulationBehaviour

6.77.3.62 SpawnAsync< T >() [2/2]

```
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

T	Must be a Type derived from SimulationBehaviour

Parameters

prefab	SimulationBehaviour used to spawn the NetworkObject

Returns

T reference, or null if it was not able to spawn the object

¨), <param name=¨position¨>Spawn Position</param> <param name=¨rotation¨>Spawn Rotation</param> <param name=¨inputAuthority¨>Player Input Authority</param> <param name=¨onBeforeSpawned¨><see cref=¨OnBeforeSpawned¨/> reference</param> <param name=¨flags">Spawn flags

Type Constraints

T* : *SimulationBehaviour

6.77.3.63 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

args	Custom arguments used to setup the Fusion Simulation

Returns

Task that can be awaited to chain actions

6.77.3.64 TryFindBehaviour()

```
bool TryFindBehaviour (
            NetworkBehaviourId bref,
            out NetworkBehaviour behaviour )
```

Get the NetworkBehaviour instance for this NetworkRunner from a NetworkBehaviourId.

Parameters

bref	
behaviour	

Returns

True if object was found.

6.77.3.65 TryFindBehaviour< T >()

```
bool TryFindBehaviour< T > (
            NetworkBehaviourId id,
            out T behaviour )
```

Try to find a NetworkBehaviour with the provided NetworkBehaviourId.

Parameters

id	The NetworkBehaviourId to search for
behaviour	The behaviour found

Template Parameters

T	A NetworkBehaviour type

Returns

Returns true if the behaviour was found and it is alive. False otherwise

Type Constraints

T* : *NetworkBehaviour

6.77.3.66 TryFindObject()

```
bool TryFindObject (
            NetworkId objectId,
            out NetworkObject networkObject )
```

Get the NetworkObject instance for this NetworkRunner from a NetworkId.

Parameters

objectId	Object NetworkID to look forward
networkObject	NetworkObject reference, if found

Returns

True if object was found.

6.77.3.67 TryGetBehaviourStats()

```
bool TryGetBehaviourStats (
            List< (Type, BehaviourStats)> result )
```

Populate the provided list with all registered behaviours and their BehaviourStats.

Parameters

result	The list to be populated

Returns

Returns true if at least one item is added to the list

6.77.3.68 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

6.77.3.69 TryGetNetworkedBehaviourFromNetworkedObjectRef< T >()

```
T TryGetNetworkedBehaviourFromNetworkedObjectRef< T > (
            NetworkId id )
```

Tries to return the first instance of T found on the root of a NetworkObject.

Template Parameters

T	

Parameters

id	

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

6.77.3.70 TryGetNetworkedBehaviourId()

```
NetworkBehaviourId TryGetNetworkedBehaviourId (
            NetworkBehaviour behaviour )
```

Tries to return a NetworkBehaviourId for the NetworkBehaviour provided.

Parameters

behaviour	

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.

6.77.3.71 TryGetObjectRefFromNetworkedBehaviour()

```
NetworkId TryGetObjectRefFromNetworkedBehaviour (
            NetworkBehaviour behaviour )
```

Tries to return the behaviour NetworkId.

Parameters

behaviour	

Returns

Returns the NetworkId of the provided behaviour. Returns default if the behaviour is not alive or the Network↩ Object that has this behaviour is not valid.

6.77.3.72 TryGetObjectStats()

```
bool TryGetObjectStats (
            NetworkId id,
            out NetworkObjectStats stats )
```

Try to get NetworkObjectStats buffer for a NetworkObject from this NetworkRunner.

Returns

Returns false if stats were not available.

6.77.3.73 TryGetPlayerObject()

```
bool TryGetPlayerObject (
            PlayerRef player,
            out NetworkObject networkObject )
```

Try to gets the NetworkObject associated with a specific player.

Parameters

player	
networkObject	Network object if one is associated with the player

Returns

Signals if it was able to get a NetworkObject for the player provided

6.77.3.74 TryGetPlayerStats()

```
bool TryGetPlayerStats (
            PlayerRef player,
            out SimulationConnectionStats stats )
```

Try to get SimulationConnectionStats buffer for a player reference from this NetworkRunner.

Returns

Returns false if stats were not available.

6.77.3.75 TryGetSceneInfo()

```
bool TryGetSceneInfo (
            out NetworkSceneInfo sceneInfo )
```

Tries to get the NetworkSceneInfo of this NetworkRunner.

Parameters

sceneInfo	The result NetworkSceneInfo

Returns

Returns true if it was able to get the scene info

6.77.3.76 TryGetSimulationStats()

```
bool TryGetSimulationStats (
            out SimulationStats stats )
```

Try to get SimulationStats buffer from this NetworkRunner.

Returns

Returns false if stats were not available.

6.77.3.77 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

prefab	A GameObject with a NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

Returns

NetworkSpawnStatus reference, or null if it was not able to spawn the object

6.77.3.78 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 NetworkPosition↩ Rotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

Parameters

prefab	Prefab used to spawn the NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

Returns

NetworkSpawnStatus reference, or null if it was not able to spawn the object

6.77.3.79 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

Parameters

prefabGuid	Object Guid used to spawn the NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

Returns

NetworkSpawnStatus reference, or null if it was not able to spawn the object

6.77.3.80 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

Parameters

typeId	Prefab ID used to spawn the NetworkObject
position	Spawn Position
rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

Returns

NetworkSpawnStatus reference, or null if it was not able to spawn the object

6.77.3.81 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 NetworkPositionRotation, or any of its derived classes such as NetworkTransform to replicate the initial transform state.

Parameters

prefabRef	Prefab Ref used to spawn the NetworkObject
position	Spawn Position

Parameters

rotation	Spawn Rotation
inputAuthority	Player Input Authority
onBeforeSpawned	OnBeforeSpawned reference
flags	Spawn flags

Returns

NetworkSpawnStatus reference, or null if it was not able to spawn the object

6.77.3.82 TrySpawn< T >() [1/2]

```
NetworkSpawnStatus TrySpawn< T > (
             T prefab,
             out T obj,
             Vector3? position,
             Quaternion? rotation,
             PlayerRef? inputAuthority,
             OnBeforeSpawned onBeforeSpawned,
             bool syncPhysics,
             bool dontDestroyOnLoad = false,
             bool? masterClientObjectOverride = null )
```

Type Constraints

T* : *SimulationBehaviour

6.77.3.83 TrySpawn< T >() [2/2]

```
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

T	Must be a Type derived from SimulationBehaviour

Parameters

prefab	SimulationBehaviour used to spawn the NetworkObject

Returns

T reference, or null if it was not able to spawn the object

¨), <param name=¨position¨>Spawn Position</param> <param name=¨rotation¨>Spawn Rotation</param> <param name=¨inputAuthority¨>Player Input Authority</param> <param name=¨onBeforeSpawned¨><see cref=¨OnBeforeSpawned¨/> reference</param> <param name=¨flags">Spawn flags

Type Constraints

T* : *SimulationBehaviour

6.77.4 Property Documentation

6.77.4.1 UserId

```
string UserId  [get]
```

Photon Client UserID.

Returns null if Peer is not connected to Photon Cloud

6.78 NetworkRunnerCallbackArgs Class Reference

Stores data types used on the INetworkRunnerCallbacks interface.

Classes

- class ConnectRequest

  Data holder of a Connection Request from a remote client.

6.78.1 Detailed Description

Stores data types used on the INetworkRunnerCallbacks interface.

6.79 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.

6.79.1 Detailed Description

Data holder of a Connection Request from a remote client.

6.80 NetworkSceneInfo Struct Reference

The default implementation of INetworkSceneInfo. 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, Local↩ PhysicsMode localPhysicsMode=LocalPhysicsMode.None, bool activeOnLoad=false)
- bool **Equals** (NetworkSceneInfo other)
- override bool **Equals** (object obj)
- override int **GetHashCode** ()
- int **IndexOf** ((SceneRef SceneRef, NetworkLoadSceneParameters SceneParams) scene)
- int **IndexOf** (SceneRef sceneRef, NetworkLoadSceneParameters sceneParams)
- bool **RemoveSceneRef** (SceneRef sceneRef)
- override string **ToString** ()

Static Public Member Functions

- static implicit **operator NetworkSceneInfo** (SceneRef sceneRef)

Static Public Attributes

- const int **MaxScenes** = 8
- const int **SIZE** = 52
- const int **WORD_COUNT** = 13

Properties

- int **SceneCount** `[get]`
- FixedArray< NetworkLoadSceneParameters > **SceneParams** `[get]`
- FixedArray< SceneRef > **Scenes** `[get]`
- int **Version** `[get]`

6.80.1 Detailed Description

The default implementation of INetworkSceneInfo. 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.

6.81 NetworkSimulationConfiguration Class Reference

Configuration for network conditions simulation (induced latency and loss).

Public Member Functions

- NetworkSimulationConfiguration **Clone** ()
- NetConfigSimulation **Create** ()

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.Wave↩ Shape.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.

6.81.1 Detailed Description

Configuration for network conditions simulation (induced latency and loss).

6.82 NetworkString< Size > Class Template Reference

Fixed-size UTF32 string. All operations are alloc-free, except for converting to System.String.

Inherits INetworkString, INetworkStruct, IEquatable< NetworkString< Size > >, and IEnumerable< char >.

Public Member Functions

- void **Assign** (string value)
- int **Compare** (NetworkString< Size > s)
- int **Compare** (ref NetworkString< Size > s)
- int **Compare** (string s)
- int Compare< OtherSize > (NetworkString< OtherSize > other)
- int Compare< OtherSize > (ref NetworkString< OtherSize > other)
- bool **Contains** (char c)
- bool **Contains** (string str)
- bool **Contains** (uint codePoint)
- bool Contains< OtherSize > (NetworkString< OtherSize > str)
- bool Contains< OtherSize > (ref NetworkString< OtherSize > str)
- bool **EndsWith** (string s)
- bool EndsWith< OtherSize > (ref NetworkString< OtherSize > other)
- bool **Equals** (NetworkString< Size > other)
- override bool **Equals** (object obj)
- bool **Equals** (ref NetworkString< Size > other)
- bool **Equals** (string s)
- bool Equals< OtherSize > (NetworkString< OtherSize > other)
- bool Equals< OtherSize > (ref NetworkString< OtherSize > other)
- 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** ()
- IEnumerator< char > IEnumerable< char >. **GetEnumerator** ()
- IEnumerator IEnumerable. **GetEnumerator** ()
- override int **GetHashCode** ()
- int **IndexOf** (char c, int startIndex, int count)
- int **IndexOf** (char c, int startIndex=0)
- int **IndexOf** (string str, int startIndex, int count)
- int **IndexOf** (string str, int startIndex=0)
- int **IndexOf** (uint codePoint, int startIndex, int count)
- int **IndexOf** (uint codePoint, int startIndex=0)
- int IndexOf< OtherSize > (NetworkString< OtherSize > str, int startIndex, int count)
- int IndexOf< OtherSize > (NetworkString< OtherSize > str, int startIndex=0)
- int IndexOf< OtherSize > (ref NetworkString< OtherSize > str, int startIndex, int count)
- int IndexOf< OtherSize > (ref NetworkString< OtherSize > str, int startIndex=0)
- **NetworkString** (string value)
- bool Set (string value)

Converts value to UTF32 string and stores it internally.

- bool **StartsWith** (string s)
- bool StartsWith< OtherSize > (ref NetworkString< OtherSize > other)
- NetworkString< Size > **Substring** (int startIndex)
- NetworkString< Size > **Substring** (int startIndex, int length)
- NetworkString< Size > **ToLower** ()
- override string **ToString** ()
- NetworkString< Size > **ToUpper** ()

Static Public Member Functions

- static int GetCapacity< Size > ()
- static implicit **operator NetworkString** (string str)
- static **operator string** (NetworkString< Size > str)
- static bool **operator!=** (NetworkString< Size > a, NetworkString< Size > b)
- static bool **operator!=** (NetworkString< Size > a, string b)
- static bool **operator!=** (string a, NetworkString< Size > b)
- static bool **operator==** (NetworkString< Size > a, NetworkString< Size > b)
- static bool **operator==** (NetworkString< Size > a, string b)
- static bool **operator==** (string a, NetworkString< Size > b)

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(ref string, bool) to get possibly alloc-free conversion.

6.82.1 Detailed Description

Fixed-size UTF32 string. All operations are alloc-free, except for converting to System.String.

Template Parameters

Size	

Type Constraints

Size **:** ***unmanaged***

Size **:** ***IFixedStorage***

6.82.2 Member Function Documentation

6.82.2.1 Compare< OtherSize >() [1/2]

```
int Compare< OtherSize > (
             NetworkString< OtherSize > other )
```

Type Constraints

OtherSize* : *unmanaged

OtherSize* : *IFixedStorage

OtherSize* : *Compare

OtherSize* : *ref

OtherSize* : *other

6.82.2.2 Compare< OtherSize >() [2/2]

```
int Compare< OtherSize > (
             ref NetworkString< OtherSize > other )
```

Type Constraints

OtherSize* : *unmanaged

OtherSize* : *IFixedStorage

6.82.2.3 Contains< OtherSize >() [1/2]

```
bool Contains< OtherSize > (
             NetworkString< OtherSize > str )
```

Type Constraints

OtherSize* : *unmanaged

OtherSize* : *IFixedStorage

OtherSize* : *IndexOf

OtherSize* : *ref

OtherSize* : *str

6.82.2.4 Contains< OtherSize >() [2/2]

```
bool Contains< OtherSize > (
             ref NetworkString< OtherSize > str )
```

Type Constraints

OtherSize* : *unmanaged

OtherSize* : *IFixedStorage

OtherSize* : *IndexOf

OtherSize* : *ref

OtherSize* : *str

6.82.2.5 EndsWith< OtherSize >()

```
bool EndsWith< OtherSize > (
            ref NetworkString< OtherSize > other )
```

Type Constraints

OtherSize* : *unmanaged

OtherSize* : *IFixedStorage

6.82.2.6 Equals< OtherSize >() [1/2]

```
bool Equals< OtherSize > (
            NetworkString< OtherSize > other )
```

Type Constraints

OtherSize* : *unmanaged

OtherSize* : *IFixedStorage

OtherSize* : *Compare

OtherSize* : *ref

OtherSize* : *other

6.82.2.7 Equals< OtherSize >() [2/2]

```
bool Equals< OtherSize > (
            ref NetworkString< OtherSize > other )
```

Type Constraints

OtherSize* : *unmanaged

OtherSize* : *IFixedStorage

OtherSize* : *Compare

OtherSize* : *ref

OtherSize* : *other

6.82.2.8 Get()

```
bool Get (
            ref string cache )
```

Checks if *cache* is equivalent and if not converts to UTF16 and stores the result in *cache* .

Parameters

cache	
ignoreCase	

Returns

False if no conversion was performed, true otherwise.

6.82.2.9 GetCapacity< Size >()

```
static int GetCapacity< Size > ( )  [static]
```

Type Constraints

Size* : *unmanaged

Size* : *IFixedStorage

6.82.2.10 GetCharCount()

```
int GetCharCount ( )
```

Calculates the length of the equivalent UTF16 string.

Returns

6.82.2.11 IndexOf< OtherSize >() [1/4]

```
int IndexOf< OtherSize > (
              NetworkString< OtherSize > str,
              int startIndex,
              int count )
```

Type Constraints

OtherSize* : *unmanaged

OtherSize* : *IFixedStorage

OtherSize* : *IndexOf

OtherSize* : *ref

OtherSize* : *str

OtherSize* : *startIndex

OtherSize* : *count

6.82.2.12 IndexOf< OtherSize >() [2/4]

```
int IndexOf< OtherSize > (
            NetworkString< OtherSize > str,
            int startIndex = 0 )
```

Type Constraints

OtherSize* : *unmanaged
OtherSize* : *IFixedStorage
OtherSize* : *IndexOf
OtherSize* : *ref
OtherSize* : *str
OtherSize* : *startIndex
OtherSize* : *SafeLength
OtherSize* : *startIndex

6.82.2.13 IndexOf< OtherSize >() [3/4]

```
int IndexOf< OtherSize > (
            ref NetworkString< OtherSize > str,
            int startIndex,
            int count )
```

Type Constraints

OtherSize* : *unmanaged
OtherSize* : *IFixedStorage

6.82.2.14 IndexOf< OtherSize >() [4/4]

```
int IndexOf< OtherSize > (
            ref NetworkString< OtherSize > str,
            int startIndex = 0 )
```

Type Constraints

OtherSize* : *unmanaged
OtherSize* : *IFixedStorage
OtherSize* : *IndexOf
OtherSize* : *ref
OtherSize* : *str
OtherSize* : *startIndex
OtherSize* : *SafeLength
OtherSize* : *startIndex

6.82.2.15 Set()

```
bool Set (
            string value )
```

Converts *value* to UTF32 string and stores it internally.

Parameters

value	

Returns

False if *value* was too long to fit and had to be trimmed.

6.82.2.16 StartsWith< OtherSize >()

```
bool StartsWith< OtherSize > (
            ref NetworkString< OtherSize > other )
```

Type Constraints

OtherSize* : *unmanaged

OtherSize* : *IFixedStorage

6.82.3 Property Documentation

6.82.3.1 this[int index]

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

Returns UTF32 scalar at *index* position. To iterate over characters, use GetEnumerator.

Parameters

index	

Returns

6.83 NetworkStructWeavedAttribute Class Reference

Describes the total number of WORDs a Fusion.INetworkedStruct uses.

Inherits Attribute.

Public Member Functions

- **NetworkStructWeavedAttribute** (int wordCount)

Properties

- int **WordCount** `[get]`

6.83.1 Detailed Description

Describes the total number of WORDs a Fusion.INetworkedStruct uses.

6.84 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.

- virtual void SetAreaOfInterestOverride (NetworkObject obj)

  Manually set the NetworkObject used as the AreaOfInterestOverride.

Public Member Functions inherited from NetworkBehaviour

- virtual void **CopyBackingFieldsToState** (bool firstTime)
- void CopyStateFrom (NetworkBehaviour source)

  Copies entire state of passed in source NetworkBehaviour
- virtual void **CopyStateToBackingFields** ()
- virtual void Despawned (NetworkRunner runner, bool hasState)

  Called before the network object is despawned.
- override void FixedUpdateNetwork ()

  Fusion FixedUpdate timing callback.
- ArrayReader< T > **GetArrayReader< T >** (string property)
- BehaviourReader< T > GetBehaviourReader< T > (string property)
- ChangeDetector **GetChangeDetector** (ChangeDetector.Source source, bool copyInitial=true)
- DictionaryReader< K, V > **GetDictionaryReader< K, V >** (string property)
- T? GetInput< T > ()
- 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 Fixed↩ UpdateNetwork).

- LinkListReader< T > **GetLinkListReader< T >** (string property)
- int **GetLocalAuthorityMask** ()

  Gets a bitmask of AuthorityMasks flags, representing the current local authority over this NetworkObject.
- PropertyReader< T > GetPropertyReader< T > (string property)
- ref T ReinterpretState< T > (int offset=0)

  Allows read and write access to the internal state buffer.
- 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)

- 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.

Public Member Functions inherited from Behaviour

- 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()

- void Teleport (Vector3? position=null, Quaternion? rotation=null)

  Teleports to the indicated values, and network the Teleport event.

- void BeforeAllTicks (bool resimulation, int tickCount)

  Called before the resimulation loop (when applicable), and also before the forward simulation loop. Only called on Updates where resimulation or forward ticks are processed.

- void AfterAllTicks (bool resimulation, int tickCount)

  Called after the resimulation loop (when applicable), and also after the forward simulation loop. Only called on Updates where resimulation or forward ticks are processed.

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.

Public Attributes inherited from NetworkBehaviour

- int **offset**

  Gives access to the offset (in 32 bit words) and count (in 32 bit words) of this behaviour backing data.

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.

Properties inherited from NetworkTRSP

- 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.

Properties inherited from NetworkBehaviour

- 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]`
- bool **StateBufferIsValid** `[get]`
- int int count **WordInfo** `[get]`

Properties inherited from SimulationBehaviour

- bool **CanReceiveRenderCallback** `[get]`
- bool **CanReceiveSimulationCallback** `[get]`
- NetworkObject **Object** `[get]`

  The NetworkObject this component is associated with.
- NetworkRunner **Runner** `[get]`

  The NetworkRunner this component is associated with.

Additional Inherited Members

Static Public Member Functions inherited from NetworkBehaviour

- static ArrayReader< T > **GetArrayReader< T >** (Type behaviourType, string property)
- static BehaviourReader< T > GetBehaviourReader< T > (NetworkRunner runner, Type behaviourType, string property)
- static BehaviourReader< TProperty > GetBehaviourReader< TBehaviour, TProperty > (NetworkRunner runner, string property)
- static DictionaryReader< K, V > **GetDictionaryReader< K, V >** (Type behaviourType, string property)
- static LinkListReader< T > **GetLinkListReader< T >** (Type behaviourType, string property)
- static PropertyReader< T > GetPropertyReader< T > (Type behaviourType, string property)
- static PropertyReader< TProperty > GetPropertyReader< TBehaviour, TProperty > (string property)
- 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 > ()
- static T ∗ MakePtr< T > (T defaultValue)
- static ref T MakeRef< T > ()
- static ref T MakeRef< T > (T defaultValue)
- static int **NetworkDeserialize** (NetworkRunner runner, byte ∗data, ref NetworkBehaviour result)
- static int **NetworkSerialize** (NetworkRunner runner, NetworkBehaviour obj, byte ∗data)
- static NetworkBehaviour **NetworkUnwrap** (NetworkRunner runner, NetworkBehaviourId wrapper)
- static NetworkBehaviourId **NetworkWrap** (NetworkRunner runner, NetworkBehaviour obj)
- static implicit operator NetworkBehaviourId (NetworkBehaviour behaviour)

  Converts NetworkBehaviour to NetworkBehaviourId.

Static Public Member Functions inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

Protected Member Functions inherited from NetworkBehaviour

- virtual bool **ReplicateTo** (PlayerRef player)

Static Protected Member Functions inherited from NetworkTRSP

- 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 AreaOf↩ InterestOverride.
- static void **SetParentTransform** (NetworkTRSP behaviour, Transform transform, NetworkBehaviour↩ Id 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.

6.84.1 Detailed Description

Add to any NetworkObject Transform, or its associated child Transforms to automatically synchronize TRSP (Position/Rotation/Scale/Parent).

6.84.2 Member Function Documentation

6.84.2.1 Render()

```
override 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 from SimulationBehaviour.

6.84.2.2 SetAreaOfInterestOverride()

```
override void SetAreaOfInterestOverride (
            NetworkObject obj ) [virtual]
```

Manually set the NetworkObject used as the AreaOfInterestOverride.

Parameters

obj	

Reimplemented from NetworkTRSP.

6.84.2.3 Spawned()

```
override void Spawned ( ) [virtual]
```

Post spawn callback.

Reimplemented from NetworkBehaviour.

6.84.2.4 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.

6.85 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.

Public Member Functions inherited from NetworkBehaviour

- virtual void **CopyBackingFieldsToState** (bool firstTime)
- void CopyStateFrom (NetworkBehaviour source)

    Copies entire state of passed in source NetworkBehaviour
- virtual void **CopyStateToBackingFields** ()
- virtual void Despawned (NetworkRunner runner, bool hasState)

    Called before the network object is despawned.
- override void FixedUpdateNetwork ()

    Fusion FixedUpdate timing callback.
- ArrayReader< T > **GetArrayReader< T >** (string property)
- BehaviourReader< T > GetBehaviourReader< T > (string property)
- ChangeDetector **GetChangeDetector** (ChangeDetector.Source source, bool copyInitial=true)
- DictionaryReader< K, V > **GetDictionaryReader< K, V >** (string property)
- T? GetInput< T > ()
- 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 Fixed↩ UpdateNetwork).
- LinkListReader< T > **GetLinkListReader< T >** (string property)
- int **GetLocalAuthorityMask** ()

Gets a bitmask of AuthorityMasks flags, representing the current local authority over this NetworkObject.

- PropertyReader< T > GetPropertyReader< T > (string property)
- ref T ReinterpretState< T > (int offset=0)

  Allows read and write access to the internal state buffer.
- 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)

Public Member Functions inherited from SimulationBehaviour

- 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.

Public Member Functions inherited from Behaviour

- 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 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 AreaOf↩ InterestOverride.
- static void **SetParentTransform** (NetworkTRSP behaviour, Transform transform, NetworkBehaviour↩ Id 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.

Properties inherited from NetworkBehaviour

- 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]`
- bool **StateBufferIsValid** `[get]`
- int int count **WordInfo** `[get]`

Properties inherited from SimulationBehaviour

- bool **CanReceiveRenderCallback** `[get]`
- bool **CanReceiveSimulationCallback** `[get]`
- NetworkObject **Object** `[get]`

  The NetworkObject this component is associated with.
- NetworkRunner **Runner** `[get]`

  The NetworkRunner this component is associated with.

Additional Inherited Members

Static Public Member Functions inherited from NetworkBehaviour

- static ArrayReader< T > **GetArrayReader< T >** (Type behaviourType, string property)
- static BehaviourReader< T > GetBehaviourReader< T > (NetworkRunner runner, Type behaviourType, string property)
- static BehaviourReader< TProperty > GetBehaviourReader< TBehaviour, TProperty > (NetworkRunner runner, string property)
- static DictionaryReader< K, V > **GetDictionaryReader< K, V >** (Type behaviourType, string property)
- static LinkListReader< T > **GetLinkListReader< T >** (Type behaviourType, string property)
- static PropertyReader< T > GetPropertyReader< T > (Type behaviourType, string property)
- static PropertyReader< TProperty > GetPropertyReader< TBehaviour, TProperty > (string property)
- 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 > ()
- static T ∗ MakePtr< T > (T defaultValue)

- static ref T MakeRef< T > ()
- static ref T MakeRef< T > (T defaultValue)
- static int **NetworkDeserialize** (NetworkRunner runner, byte ∗data, ref NetworkBehaviour result)
- static int **NetworkSerialize** (NetworkRunner runner, NetworkBehaviour obj, byte ∗data)
- static NetworkBehaviour **NetworkUnwrap** (NetworkRunner runner, NetworkBehaviourId wrapper)
- static NetworkBehaviourId **NetworkWrap** (NetworkRunner runner, NetworkBehaviour obj)
- static implicit operator NetworkBehaviourId (NetworkBehaviour behaviour)

  Converts NetworkBehaviour to NetworkBehaviourId.

Static Public Member Functions inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

Public Attributes inherited from NetworkBehaviour

- 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 inherited from NetworkBehaviour

- virtual bool **ReplicateTo** (PlayerRef player)

6.85.1 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.

6.85.2 Member Function Documentation

6.85.2.1 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 Area↩
OfInterestOverride.

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

parent	The direct parent of the

6.85.2.2 SetAreaOfInterestOverride()

```
virtual void SetAreaOfInterestOverride (
            NetworkObject obj )  [virtual]
```

Manually set the NetworkObject used as the AreaOfInterestOverride.

Parameters

obj	

Reimplemented in NetworkTransform.

6.86 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
- NetworkBehaviourId **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.

6.86.1 Detailed Description

Data structure storing spatial (Position/Rotation/Scale/Parent) synchronization data for spatial synchronization components, NetworkTRSP and its subclass NetworkTransform.

6.87 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**
- bool **InvertY**

6.87.1 Detailed Description

Enables a special inspector drawer for Unity Rect type, specially designed for editing RectTransforms using normalized values.

6.87.2 Constructor & Destructor Documentation

6.87.2.1 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

invertY	
aspectRatio	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.

6.88 PlayerRef Struct Reference

Represents a Fusion player.

Inherits INetworkStruct, and IEquatable< PlayerRef >.

Public Member Functions

- override bool **Equals** (object obj)
- bool **Equals** (PlayerRef other)
- override int **GetHashCode** ()
- override string **ToString** ()

Static Public Member Functions

- static PlayerRef **FromEncoded** (int encoded)
- static PlayerRef **FromIndex** (int index)
- static bool **operator!=** (PlayerRef a, PlayerRef b)
- static bool **operator==** (PlayerRef a, PlayerRef b)
- static unsafe PlayerRef **Read** (NetBitBuffer ∗buffer)
- static unsafe void **Write** (NetBitBuffer ∗buffer, PlayerRef playerRef)
- static unsafe void Write< T > (T ∗buffer, PlayerRef playerRef)

Public Attributes

- int **_index**

Static Public Attributes

- const int **MASTER_CLIENT_RAW** = -1
- const int **SIZE** = 4

Properties

- int AsIndex `[get]`

  Returns the PlayerRef int as an integer Id value.
- static IEqualityComparer< PlayerRef > **Comparer** = new IndexEqualityComparer() `[get]`
- 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.

6.88.1 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 Player↩ Ref.

```
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
```

6.88.2 Member Function Documentation

6.88.2.1 Write< T >()

```
static unsafe void Write< T > (
            T * buffer,
            PlayerRef playerRef )  [static]
```

Type Constraints

T : unmanaged

T : INetBitWriteStream

6.88.3 Property Documentation

6.88.3.1 AsIndex

```
int AsIndex [get]
```

Returns the PlayerRef int as an integer Id value.

-1=None -2=MasterClient >=0=PlayerId

6.88.3.2 PlayerId

```
int PlayerId [get]
```

Returns the PlayerRef as an integer Id value.

-1=None -2=MasterClient

6.88.3.3 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

6.89 IMessage Interface Reference

Represents a Protocol Message.

6.89.1 Detailed Description

Represents a Protocol Message.

Used to tag the Messages in ICommunicator.

6.90 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)

Properties

- string Method `[get, set]`
- RenderSource Source `[get, set]`
- RenderTimeframe Timeframe `[get, set]`

6.90.1 Detailed Description

Override default render settings for `[Networked]` properties.

6.90.2 Property Documentation

6.90.2.1 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) { /* ... */ }
```

6.90.2.2 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.

6.90.2.3 Timeframe

```
RenderTimeframe Timeframe [get], [set]
```

Force this property to be rendered in this RenderTimeframe.

This setting is prioritized over NetworkBehaviour and NetworkObject overrides.

6.91 RenderTimeline Struct Reference

Can be used to acquire RenderData for different points in time.

Static Public Member Functions

- static void **GetRenderBuffers** (NetworkBehaviour behaviour, out NetworkBehaviourBuffer from, out NetworkBehaviourBuffer to, out float alpha)

6.91.1 Detailed Description

Can be used to acquire RenderData for different points in time.

6.92 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 RpcInfo 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.

Properties

- RpcChannel **Channel** = RpcChannel.Reliable `[get, set]`

    Specifies which RpcChannel to use. Default value is RpcChannel.Reliable
- RpcHostMode **HostMode** = RpcHostMode.SourceIsServer `[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.
- bool **InvokeResim** `[get, set]`
- 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.

6.92.1 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 RpcInfo 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, Rpc↩ Info info = default) { }` To target a specific Player, use the RpcTargetAttribute: `| [Rpc] | public void RpcFoo([RpcTarget] PlayerRef targetPlayer) {}` Use RpcInvokeInfo 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) { }`

6.92.2 Constructor & Destructor Documentation

6.92.2.1 RpcAttribute()

```
RpcAttribute (
            RpcSources sources,
            RpcTargets targets )
```

Constructor for RpcAttributes.

Parameters

sources	The legal RpcSources types that can trigger this Rpc. Default is RpcSources.All
targets	The RpcTargets types that will receive and invoke this method. Default is RpcTargets.All

6.93 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** ()

Public Attributes

- RpcLocalInvokeResult **LocalInvokeResult**
- RpcSendCullResult **SendCullResult**
- RpcSendResult **SendResult**

6.93.1 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); | }
```

6.94 RpcSendResult Struct Reference

RPC send operation result information.

Public Member Functions

- override string **ToString** ()

Public Attributes

- int **MessageSize**

  The size of the RPC message.
- RpcSendMessageResult **Result**

  Result flags for the RPC send operation.

6.94.1 Detailed Description

RPC send operation result information.

6.95 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.

6.95.1 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) { }
```

6.96 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.

6.96.1 Detailed Description

Holds information about the Game Session.

6.96.2 Member Function Documentation

6.96.2.1 operator bool()

```
static implicit operator bool (
              SessionInfo sessionInfo )  [static]
```

Check if the SessionInfo reference is not Null and is Valid.

Parameters

sessionInfo	

6.96.2.2 ToString()

```
override string ToString ( )
```

String representation of a SessionInfo

Returns

Formatted SessionInfo

6.96.2.3 UpdateCustomProperties()

```
bool UpdateCustomProperties (
            Dictionary< string, SessionProperty > customProperties )
```

Update or change the Custom Properties of the current joined Room.

Parameters

customProperties	New custom properties

6.97 Simulation Class Reference

Main simulation class.

Inherits ILogSourceProxy, and INetPeerGroupCallbacks.

Public Member Functions

- void GetAreaOfInterestGizmoData (List<(Vector3 center, Vector3 size, int playerCount, int objectCount)> result)

  Clears the passed results collection, and adds all current AOI cell data. Each element in the List represents one AOI cell.
- PlayerRef **GetInputAuthority** (NetworkObject networkObject)
- SimulationInput **GetInputForPlayer** (PlayerRef player)
- void **GetObjectsAndPlayersInAreaOfInterestCell** (int cellKey, List< PlayerRef > players, List< NetworkId > objects)

  Used by RunnerAOIGizmos component. Supplies data about current active AOI cells.
- PlayerRef **GetStateAuthority** (NetworkObject networkObject)
- bool HasAnyActiveConnections ()

  Signal if the Server has any Active Connection with any number of Clients.
- bool **IsInputAuthority** (NetworkObject networkObject, PlayerRef playerRef)

- bool? **IsInterestedIn** (NetworkObject obj, PlayerRef player)
- bool **IsLocalSimulationInputAuthority** (NetworkObject obj)
- bool **IsLocalSimulationStateAuthority** (NetworkId id)
- bool **IsLocalSimulationStateAuthority** (NetworkObject obj)
- bool **IsLocalSimulationStateOrInputSource** (NetworkObject obj)
- bool **IsStateAuthority** (NetworkObject networkObject, PlayerRef playerRef)
- bool **IsStateAuthority** (PlayerRef stateSource, PlayerRef playerRef)
- bool **TryGetHostPlayer** (out PlayerRef player)
- int Update (double dt)

  Forwards the Simulation based on the Delta Time.

Public Attributes

- SimulationStats **_stats** = new SimulationStats()

Protected Member Functions

- virtual void **AfterSimulation** ()
- virtual void **AfterUpdate** ()
- virtual void **BeforeFirstTick** ()
- virtual int **BeforeSimulation** ()
- virtual void **BeforeUpdate** ()
- virtual void **NetworkConnected** (NetConnection ∗connection)
- virtual void **NetworkDisconnected** (NetConnection ∗connection, NetDisconnectReason reason)
- virtual void **NetworkReceiveDone** ()
- virtual void **NoSimulation** ()

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 SimulationConfig.TickRate.
- int **InputCount** `[get]`
- 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]`
- SimulationModes **Mode** `[get]`

  Gets the SimulationModes flags for The type of network peer this simulation represents.
- NetConfig ∗ **NetConfigPointer** `[get]`
- int **ObjectCount** `[get]`
- IReadOnlyDictionary< NetworkId, NetworkObjectMeta > **Objects** `[get]`
- NetworkProjectConfig **ProjectConfig** `[get]`

  The NetworkProjectConfig file used by this Simulation.
- float **RemoteAlpha** `[get]`
- Tick **RemoteTick** `[get]`
- Tick **RemoteTickPrevious** `[get]`
- 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]`
- Topologies **Topology** `[get]`

  Indicates if a Server/Client or Shared Mode (relay server) topology is being used.

6.97.1 Detailed Description

Main simulation class.

6.97.2 Member Function Documentation

6.97.2.1 GetAreaOfInterestGizmoData()

```
void GetAreaOfInterestGizmoData (
            List<(Vector3 center, Vector3 size, int playerCount, int objectCount)> result )
```

Clears the passed results collection, and adds all current AOI cell data. Each element in the List represents one AOI cell.

Parameters

result	

6.97.2.2 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.

6.97.2.3 Update()

```
int Update (
            double dt )
```

Forwards the Simulation based on the Delta Time.

Parameters

dt	Delta Time used to forward the simulation

Returns

How many Ticks executed on this Update

6.97.3 Property Documentation

6.97.3.1 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.

6.97.3.2 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.

6.97.3.3 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.

6.98 SimulationBehaviour Class Reference

Base class for a Fusion aware Behaviour (derived from UnityEngine.MonoBehavour). Objects derived from this object can be associated with a NetworkRunner and Simulation. If a parent NetworkObject is found, this component will also be associated with that network entity.

Inherits Behaviour.

Inherited by HitboxManager, NetworkBehaviour, NetworkPhysicsSimulation2D, and NetworkPhysicsSimulation3D.

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.

Public Member Functions inherited from Behaviour

- 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()

Properties

- bool **CanReceiveRenderCallback** `[get]`
- bool **CanReceiveSimulationCallback** `[get]`
- NetworkObject **Object** `[get]`

  The NetworkObject this component is associated with.
- NetworkRunner **Runner** `[get]`

  The NetworkRunner this component is associated with.

Additional Inherited Members

Static Public Member Functions inherited from Behaviour

- static void **DestroyBehaviour** (Behaviour behaviour)

  Wrapper for Unity's GameObject.Destroy()

6.98.1 Detailed Description

Base class for a Fusion aware Behaviour (derived from UnityEngine.MonoBehavour). Objects derived from this object can be associated with a NetworkRunner and Simulation. If a parent NetworkObject is found, this component will also be associated with that network entity.

6.98.2 Member Function Documentation

6.98.2.1 FixedUpdateNetwork()

```
virtual void FixedUpdateNetwork ( )  [virtual]
```

Fusion FixedUpdate timing callback.

Reimplemented in NetworkBehaviour.

6.98.2.2 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 NetworkMecanimAnimator, and NetworkTransform.

6.99 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.

6.99.1 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 = Simulation↩
Modes.Server | SimulationModes.Host)]
```

6.100 SimulationConfig Class Reference

Project configuration settings specific to how the Simulation class behaves.

Public Types

- enum DataConsistency
- enum InputTransferModes

Public Attributes

- bool **HostMigration**

  If, in host mode, we should allow host migration if the current host leaves.
- int **InputDataWordCount**
- 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**

  Features to enabled to replication such as area of interest, etc.
- int **SceneInfoWordCount** = 16
- TickRate.Selection **TickRateSelection** = Fusion.TickRate.Default
- Topologies **Topology**

  The topology used.

Properties

- bool **AreaOfInterestEnabled** `[get]`
- int **InputTotalWordCount** `[get]`
- bool **SchedulingEnabled** `[get]`
- bool **SchedulingWithoutAOI** `[get]`

6.100.1 Detailed Description

Project configuration settings specific to how the Simulation class behaves.

6.100.2 Member Enumeration Documentation

6.100.2.1 DataConsistency

```
enum DataConsistency
```

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).

6.100.2.2 InputTransferModes

```
enum InputTransferModes
```

Enumerator

Redundancy	Send delta compressed and redundant input, used for most games.
LatestState	Only send latest input state, useful for VR, etc.

6.101 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)

6.101.1 Detailed Description

Stores the runtime configuration of the simulation.

6.102 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 **IsIPv6** `[get]`

  Signal if the NetAddress represents an IPv6 Address.
- bool **IsRelayAddr** `[get]`

  Signal if the NetAddress is a Relayed connection.
- bool **IsValid** `[get]`

  Signal if this NetAddress is not default/empty.

6.102.1 Detailed Description

Represents a Network Address, which includes a IP and Port This can contains either a IPv4 or a IPv6 address.

6.102.2 Member Function Documentation

6.102.2.1 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

port	Port used to build the NetAddress

Returns

New NetAddress reference

6.102.2.2 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

port	Port used to build the NetAddress

Returns

New NetAddress reference

6.102.2.3 CreateFromIpPort()

```
static NetAddress CreateFromIpPort (
             string ip,
             ushort port )  [static]
```

Create a new NetAddress based on the IP and Port passed as argument.

Parameters

ip	String representation of an IP, either IPv4 or IPv6
port	Port used to build the NetAddress

Returns

New NetAddress reference

Exceptions

ArgumentException	If IP is empty/null or an invalid IP, or port $<$ 0
AssertException	If unable to parse IP

6.102.2.4 FromActorId()

```
static NetAddress FromActorId (
             int actorId )  [static]
```

Build a new NetAddress based on an ActorId.

Parameters

actor↩ Id	ActorId used to build the NetAddress

Returns

Relay NetAddress that references the ActorId

ActorId must be 0 or greated

6.102.2.5 LocalhostIPv4()

```
static NetAddress LocalhostIPv4 (
             ushort port = 0 )  [static]
```

Create a new NetAddress on the LocalHost address with the desired Port.

Parameters

port	Port used to build the NetAddress

Returns

New NetAddress reference

6.102.2.6 LocalhostIPv6()

```
static NetAddress LocalhostIPv6 (
            ushort port = 0 )  [static]
```

Create a new NetAddress on the LocalHost IPv6 Address with the desired Port.

Parameters

port	Port used to build the NetAddress

Returns

New NetAddress reference

6.103 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**

6.103.1 Detailed Description

Represents a linked list of Fusion.Sockets.NetBitBuffer

6.103.2 Member Function Documentation

6.103.2.1 AddFirst()

```
void AddFirst (
            NetBitBuffer * item )
```

Add a Fusion.Sockets.NetBitBuffer at the beginning of the List.

Parameters

item	NetBitBuffer to add to the list

6.103.2.2 AddLast()

```
void AddLast (
            NetBitBuffer * item )
```

Add a Fusion.Sockets.NetBitBuffer at the end of the list.

Parameters

item	NetBitBuffer to add to the list

6.103.2.3 IsInList()

```
bool IsInList (
            NetBitBuffer * item )
```

Check if a specific Fusion.Sockets.NetBitBuffer is in the list.

Parameters

item	NetBitBuffer to check

Returns

True if the list contains the item, false otherwise

6.103.2.4 Remove()

```
void Remove (
            NetBitBuffer * item )
```

Remove a specific Fusion.Sockets.NetBitBuffer from the list.

Parameters

item	NetBitBuffer to remove

6.103.2.5 RemoveHead()

```
NetBitBuffer * RemoveHead ( )
```

Removes the first element of the list.

Returns

NetBitBuffer reference

6.104 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**

6.104.1 Detailed Description

Accepted Command, sent by the server when a remote client connection is accepted.

6.105 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

6.105.1 Detailed Description

Connect Command used to signal a remote server that a client is trying to connect to it.

6.106 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

6.106.1 Detailed Description

Disconnect Command, it can be used by either side of the connection.

6.107 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

6.107.1 Detailed Description

Network Command Header Describe its type and usual settings for all commands.

6.107.2 Member Function Documentation

6.107.2.1 Create()

```
static NetCommandHeader Create (
            NetCommands command )  [static]
```

Create a new NetCommandHeader based on a NetCommands type.

Parameters

command	Type of Command that should be created

Returns

New NetCommandHeader reference based on the Command Type

6.108 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

6.108.1 Detailed Description

Refuse Command, sent by the server when the connection was refused. This happens when the server has reached its max connection capacity.

6.109 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 **DefaultMtu**

    default Maximum Transmission Unit
- 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

6.109.1 Detailed Description

General configuration used to drive the behavior of the Socket library.

6.109.2 Member Data Documentation

6.109.2.1 ConnectionShutdownTime

```
double ConnectionShutdownTime
```

Timeout in Seconds to allow a disconnected Connection to be released from the Group Mapping.

NetPeerGroup.UpdateShutdown

6.110 StunClient.TestIPs Class Reference

List of public DNS Servers.

Static Public Attributes

- static IPEndPoint **TEST_NET_IPV4** = new IPEndPoint(IPAddress.Parse(¨203.0.113.0¨), 65530)
- static IPEndPoint **TEST_NET_IPV6** = new IPEndPoint(IPAddress.Parse(¨2001:db8::¨), 65530)

6.110.1 Detailed Description

List of public DNS Servers.

6.111 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.
- 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 this peer, when running in Server/Host Mode.
- NetworkSceneInfo? Scene

    Scene that will be set as the starting Scene when running in Server/Host Mode.
- INetworkSceneManager SceneManager

    See INetworkSceneManager.
- string SessionName

    Photon Cloud Session Name used either to Create or Join 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.

    See also

    https://doc.photonengine.com/fusion/current/connection-and-authentication/tcp-and-udp-port-numbers

6.111.1 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`

6.111.2 Member Data Documentation

6.111.2.1 Address

```
NetAddress? Address
```

Peer Binding Address.

Default: NetAddress.Any(ushort)

6.111.2.2 AuthValues

```
AuthenticationValues AuthValues
```

Custom Authentication Data.

Default: null (default authentication values)

6.111.2.3 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/networ`

6.111.2.4 ConnectionToken

```
byte [ ] ConnectionToken
```

Connection token sent by client to server. Not used in shared mode.

Default: null (empty connection token)

6.111.2.5 CustomCallbackInterfaces

```
Type [ ] CustomCallbackInterfaces
```

User defined callback interfaces we will provide O(1) constant time lookup for.

Default: null

6.111.2.6 CustomLobbyName

```
string CustomLobbyName
```

Session Custom Lobby to be published in.

Default: null (default Lobby for each Session Type, LobbyClientServer or LobbyShared)

6.111.2.7 CustomPhotonAppSettings

```
FusionAppSettings CustomPhotonAppSettings
```

Custom Photon Application Settings.

Default: null (Global PhotonAppSettings)

6.111.2.8 CustomPublicAddress

```
NetAddress?  CustomPublicAddress
```

Custom Public Reflexive Address.

Default: null

6.111.2.9 CustomSTUNServer

```
string CustomSTUNServer
```

Specify a Custom STUN Server used to Resolve the peer Reflexive Addresses.

Default: null (no custom STUN Server)

6.111.2.10 DisableNATPunchthrough

```
bool DisableNATPunchthrough
```

Flag to disable the NAT Punchthrough implementation and connect only via Relay.

Default: false

6.111.2.11 EnableClientSessionCreation

```
bool?  EnableClientSessionCreation
```

Enables the Session creation when starting a Client with an specific Session Name.

Default: false (clients *can not* create new Sessions)

6.111.2.12 HostMigrationResume

```
Action<NetworkRunner> HostMigrationResume
```

Callback invoked when the new Host is migrating from the old Host state.

Default: null

6.111.2.13 HostMigrationToken

```
HostMigrationToken HostMigrationToken
```

Host Migration Token used when restarting the Fusion Simulation.

Default: null

6.111.2.14 IsOpen

```
bool? IsOpen
```

Session should be created Open or Closed to accept joins.

Default: true

6.111.2.15 IsVisible

```
bool? IsVisible
```

Session should be Visible or not in the Session Lobby list.

Default: true

6.111.2.16 MatchmakingMode

```
MatchmakingMode? MatchmakingMode
```

Session Join Matchmaking Mode when joining a Session. For more information, check Fusion.Photon.Realtime.↩ MatchmakingMode

Default: MatchmakingMode.FillRoom

6.111.2.17 ObjectProvider

```
INetworkObjectProvider ObjectProvider
```

Object pool to use.

Default: null

6.111.2.18 OnGameStarted

```
Action<NetworkRunner> OnGameStarted
```

Callback that is invoked when the Fusion has fully started.

Default: null

6.111.2.19 PlayerCount

```
int? PlayerCount
```

Number of players allowed to connect to this peer, when running in Server/Host Mode.

Default: DefaultPlayers from the Global NetworkProjectConfig

6.111.2.20 Scene

```
NetworkSceneInfo? Scene
```

Scene that will be set as the starting Scene when running in Server/Host Mode.

Default: null (no scene set)

6.111.2.21 SceneManager

```
INetworkSceneManager SceneManager
```

See INetworkSceneManager.

Default: null

More about Scene Loading: https://doc.photonengine.com/en-us/fusion/current/manual/scene-load

6.111.2.22 SessionName

```
string SessionName
```

Photon Cloud Session Name used either to Create or Join a Session.

Default: null (random session matching)

6.111.2.23 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)

6.111.2.24 StartGameCancellationToken

```
CancellationToken StartGameCancellationToken
```

Optional CancellationToken used to cancel the NetworkRunner start up process and shutdown.

Defaults: null

6.111.2.25 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: false

6.111.2.26 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.

See also

https://doc.photonengine.com/fusion/current/connection-and-authentication/tcp-and-udp-port-numbers

Default: false (uses ports 27000, 27001 and 27002)

6.112 StartGameResult Class Reference

Represents the result of starting the Fusion Simulation.

Public Member Functions

- override string ToString ()

  StartGameResult to String.

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.

6.112.1 Detailed Description

Represents the result of starting the Fusion Simulation.

6.112.2 Member Function Documentation

6.112.2.1 ToString()

```
override string ToString ( )
```

StartGameResult to String.

Returns

6.113 StatsMetaAttribute Class Reference

This stat goes on field elements of classes/structs and is used by FieldsMask.

Inherits DisplayNameAttribute.

Public Member Functions

- **StatsMetaAttribute** (bool defaultEnabled=false, int decimals=3, float multiplier=1, float warnThreshold=float.↩ PositiveInfinity, float errorThreshold=float.PositiveInfinity, StatAveraging averaging=StatAveraging.Per↩ Sample)
- **StatsMetaAttribute** (string shortName, bool defaultEnabled=false, int decimals=3, float multiplier=1, float warnThreshold=float.PositiveInfinity, float errorThreshold=float.PositiveInfinity, StatAveraging averaging=Stat↩ Averaging.PerSample)
- **StatsMetaAttribute** (string shortName, string longName, bool defaultEnabled=false, int decimals=3, float multiplier=1, float warnThreshold=float.PositiveInfinity, float errorThreshold=float.PositiveInfinity, Stat↩ Averaging averaging=StatAveraging.PerSample)

Public Attributes

- readonly StatAveraging **Averaging**
- readonly int **Decimals**
- readonly bool **DefaultEnabled**
- readonly float **ErrorThreshold**
- readonly float **Multiplier**
- readonly string **ShortName**
- readonly float **WarnThreshold**

6.113.1 Detailed Description

This stat goes on field elements of classes/structs and is used by FieldsMask.

6.114 UnitAttribute Class Reference

Unit Attribute class. Used to mark a field with the respective Units

Inherits DecoratingPropertyAttribute.

Public Member Functions

- **UnitAttribute** (Units units)

Properties

- Units **Unit** `[get]`

6.114.1 Detailed Description

Unit Attribute class. Used to mark a field with the respective Units

6.115 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 DoIfAttributeBase.

Public Member Functions

- **WarnIfAttribute** (string propertyPath, bool compareToValue, string message, CompareOperator compare=CompareOperator.Equal)
- **WarnIfAttribute** (string propertyPath, double compareToValue, string message, CompareOperator compare=CompareOperator.Equal)
- **WarnIfAttribute** (string propertyPath, long compareToValue, string message, CompareOperator compare=CompareOperator.Equal)
- **WarnIfAttribute** (string propertyPath, string message)

Public Attributes

- bool **AsBox**
- string **Message**

    The default warning text, when a warning is shown.

Public Attributes inherited from DoIfAttributeBase

- double **_doubleValue**
- bool **_isDouble**
- long **_longValue**
- CompareOperator **Compare**
- string **ConditionMember**
- bool **ErrorOnConditionMemberNotFound** = true

Additional Inherited Members

Protected Member Functions inherited from DoIfAttributeBase

- **DoIfAttributeBase** (string propertyPath, double compareToValue, CompareOperator compare)
- **DoIfAttributeBase** (string propertyPath, long compareToValue, CompareOperator compare)

6.115.1 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).

Value of condition method is converted to a long. Null = 0, False = 0, True = 1, Unity Object = InstanceId

Index