

## **Photon Unity Networking**

v1.81

**Main Page** 

**Related Pages** 

Modules

Classes

Files

## **General Documentation**

Brief overview of Photon, subscriptions, hosting options and how to start.

#### **Table of Contents**

- <sup>↓</sup> Photon

  - <sup>↓</sup>Remote Procedure Calls
  - ↓ Instantiating
     Networked Objects

## **Photon**

Unlike Unity's built-in networking, PUN always connects to a dedicated server which provides rooms, matchmaking and in-room communication for players. Behind the scenes **Photon** Unity Networking uses more than one server: Several "Game Servers" run the actual rooms (matches) while a "Master Server" keeps track of rooms and match players.

You have two options for the server side.

## **Exit Games Cloud**

The Exit Games Cloud is a service which provides hosted and load balanced **Photon** Servers for you, fully managed by Exit Games. Free trials are available and <u>subscription costs for commercial use</u> are competitively low.

The service runs a fixed logic, so you can't implement your own serverside game logic. Instead, the clients need to be authoritative.

Clients are separated by "application id", which relates to your game title and a "game version". With that, your players won't clash with those of another developer or older game iterations.

## **Subscriptions bought in Asset Store**

Follow these steps, if you bought a package with **Photon** Cloud Subscription in the Asset Store:

- Register a Photon Cloud Account: exitgames.com/en/Account/SignUp
- Create an App and get your AppID from the <u>Dashboard</u>
- Send a Mail to: <a href="mailto:developer@exitgames.com">developer@exitgames.com</a>
- With:
  - Your Name and Company (if applicable)
  - Invoice/Purchase ID from the Asset Store
  - Photon Cloud ApplD

## **Photon Server SDK**

As alternative to the **Photon** Cloud service, you can run your own server and develop server side logic on top of our "Load Balancing" C# solution. This gives you full control of the server logic.

The Photon Server SDK can be downloaded on: <a href="https://www.exitgames.com/en/OnPremise/Download">www.exitgames.com/en/OnPremise/Download</a>

Starting the Server: <u>doc.exitgames.com/en/onpremise/current/getting-started/photon-server-in-5min</u>

# **Photon Unity Networking - First steps**

When you import PUN, the "Wizard" window will popup. Either enter your email address to register for the cloud, skip this step to enter the Appld of an existing account or switch to "self hosted" **Photon** to enter your server's address.

This creates a configuration for either the cloud service or your own **Photon** server in the project: PhotonServerSettings.

PUN consists of quite a few files, however there's only one that truly matters: **PhotonNetwork**. This class contains all functions and variables needed. If you ever have custom requirements, you can always modify the source files - this plugin is just an implementation of **Photon** after all.

To use PUN from UnityScript, move both folders "PhotonNetwork" and "UtilityScripts" to the Assets\ folder.

To show you how this API works, here are a few examples right away.

## **Master Server And Lobby**

PUN always uses a master server and one or more game servers. The master server manages currently running games on the various game servers and will provide a game server address when you join or create a room. PUN (the client) automatically switches to that game server.

Individual matches are known as Rooms. They are independent of each other and identified by name. Rooms are grouped into one or multiple lobbies. Lobbies are an optional part in matchmaking. If you don't use custom lobbies explicitly, PUN will use a single lobby for all rooms.

By default, PUN will join the default lobby after connecting. This lobby sends a list of existing rooms to the client, so the player can pick a room (by name or some properties listed). Access the current list by using **PhotonNetwork.GetRoomList()**. The lists is updated in intervals to keep traffic low.

Clients don't have to join a lobby to join or create rooms. If you don't want to show a list of rooms in your client, set

PhotonNetwork.autoJoinLobby = false before you connect and your clients will skip the lobby.

You can use more than one lobby to organize room-lists as needed for your game. **PhotonNetwork.JoinLobby** is the method to join a specific lobby. You can make them up on the client side - the server will keep track of them. As long as name and type are the same, the **TypedLobby** will be the same for all clients, too.

A client is always just in one lobby and while being in a lobby, creating a room will relate to this lobby, too. Multiple lobbies mean the clients get shorter rooms lists, which is good. There is no limit to the rooms lists.

A parameter in JoinRoom, JoinRandomRoom and CreateRoom enables you to select a lobby without joining it.

Players won't notice each other in the Lobby and can't send data (to prevent issues when it's getting crowded).

The servers are all run on dedicated machines - there is no such thing as player-hosted 'servers'. You don't have to bother remembering about the server organization though, as the API all hides this for you.

```
PhotonNetwork.ConnectUsingSettings("v1.0");
```

The code above is required to make use of any **PhotonNetwork** features. It sets your client's game version and uses the setup-wizard's config (stored in: PhotonServerSettings). The wizard can also be used when you host **Photon** yourself. Alternatively, use Connect() and you can ignore the PhotonServerSettings file.

## Versioning

The **Photon** Cloud uses your appID to separate your players from everyone else's.

Within one Appld, you can deliberately separate clients/players by the "Game Version" string, which is set in the "connect" methods (as parameter).

Note: As we can't guarantee that different **Photon** Unity Networking versions are compatible with each other, we add the PUN version to your game's version. This literally adds "\_" +

PhotonNetwork.versionPuN to your Game Version string.

## **Creating and Joining Games**

Next, you'll want to join or create a room. The following code showcases some required functions:

A list of currently running games is provided by the master server's lobby. It can be joined like other rooms but only provides and updates the list of rooms. The **PhotonNetwork** plugin will automatically join the lobby after connecting. When you're joining a room, the list will no longer update.

To display the list of rooms (in a lobby):

```
foreach (RoomInfo game in
        PhotonNetwork.GetRoomList())
{
    GUILayout.Label(game.name + " " +
        game.playerCount + "/" + game.maxPlayers);
}
```

Alternatively, the game can use random matchmaking: It will try to join

any room and fail if none has room for another player. In that case: Create a room without name and wait until other players join it randomly.

## **Advanced Matchmaking & Room Properties**

Fully random matchmaking is not always something players enjoy. Sometimes you just want to play a certain map or just two versus two.

In **Photon** Cloud and Loadbalancing, you can set arbitrary room properties and filter for those in JoinRandom.

## **Room Properties and the Lobby**

**Room** properties are synced to all players in the room and can be useful to keep track of the current map, round, starttime, etc. They are handled as Hashtable with string keys. Preferably short keys.

You can forward selected properties to the lobby, too. This makes them available for listing them and for random matchmaking, too. Not all room properties are interesting in the lobby, so you define the set of properties for the lobby on room creation.

Note that "ai" is not a key in the room-properties yet. It won't show up in the lobby until it's set in the game via **Room.SetCustomProperties()**. When you change the values for "map" or "ai", they will be updated in the lobby with a short delay, too.

Keep the list short to make sure performance doesn't suffer from loading the list.

## **Filtering Room Properties in Join Random**

In JoinRandom, you could pass a Hashtable with expected room properties and max player value. These work as filters when the server selects a "fitting" room for you.

```
Hashtable expectedCustomRoomProperties = new
    Hashtable() { "map", 1 } ;
JoinRandomRoom(expectedCustomRoomProperties, 4);
```

If you pass more filter properties, chances are lower that a room matches them. Better limit the options.

Make sure you never filter for properties that are not known to the lobby (see above).

## **MonoBehaviour Callbacks**

PUN uses several callbacks to let your game know about state changes like "connected" or "joined a game". All you have to do is implement the fitting method in any MonoBehaviour and it gets called when the event happens.

To get a good overview of available callbacks, take a look at the class **Photon.PunBehaviour**. If you make your script a <u>PunBehaviour</u> (instead of a MonoBehaviour), you can override individual callbacks easily. If you begin to type "override", your coding IDE should provide you a list of callbacks, so they are easy to find while coding, too.

This covers the basics of setting up game rooms. Next up is actual communication in games.

## Sending messages in rooms

Inside a room you are able to send network messages to other connected players. Furthermore you are able to send buffered messages that will also be sent to players that connect in the future (for spawning your player for instance).

Sending messages can be done using two methods. Either RPCs or by using the **PhotonView** property OnSerializePhotonView. There is more network interaction though. You can listen for callbacks for certain network events (e.g. OnPhotonInstantiate, OnPhotonPlayerConnected) and you can trigger some of these events (**PhotonNetwork.Instantiate**). Don't worry if you're confused by the last paragraph, next up we'll explain for each of these subjects.

## **Using Groups in PUN**

Groups are not synchronized when they are changed on any **PhotonView**. It's up to the developer to keep photonviews in the same groups on all clients, if that's needed. Using different group numbers for the same photonview on several clients will cause some inconsistent behaviour.

Some network messages are checked for their receiver group at the receiver side only, namely:

- RPCS that are targeted to a single player (or MasterClient)
- RPCS that are buffered (AllBuffered/OthersBuffered).
- This includes **PhotonNetwork.Instantiate** (as it is buffered).

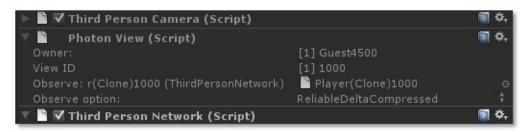
Technical reason for this: the photon server only supports interestgroups for messages that are not cached and are not targetted at sepcific actor(s). This might change in the future.

### **PhotonView**

**PhotonView** is a script component that is used to send messages (RPCs and OnSerializePhotonView). You need to attach the **PhotonView** to your games gameobjects. Note that the **PhotonView** is very similar to Unity's NetworkView.

At all times, you need at least one **PhotonView** in your game in order to send messages and optionally instantiate/allocate other PhotonViews.

To add a **PhotonView** to a gameobject, simply select a gameobject and use: "Components/Miscellaneous/Photon View".



**Photon View** 

#### **Observe Transform**

If you attach a Transform to a PhotonView's Observe property, you can choose to sync Position, Rotation and Scale or a combination of those across the players. This can be a great help for prototyping or smaller games. Note: A change to any observed value will send out all observed values - not just the single value that's changed. Also, updates are not smoothed or interpolated.

#### **Observe MonoBehaviour**

A **PhotonView** can be set to observe a MonoBehaviour. In this case, the script's OnPhotonSerializeView method will be called. This method is called for writing an object's state and for reading it, depending on whether the script is controlled by the local player.

The simple code below shows how to add character state synchronization with just a few lines of code more:

```
void OnPhotonSerializeView(PhotonStream stream,
    PhotonMessageInfo info)
{
 if (stream.isWriting)
 //We own this player: send the others our data
    stream.SendNext((int)controllerScript._char
    acterState);
       stream.SendNext(transform.position);
       stream.SendNext(transform.rotation);
 else
   {
 //Network player, receive data
       controllerScript._characterState =
    (CharacterState)(int)stream.ReceiveNext();
       correctPlayerPos =
    (Vector3)stream.ReceiveNext();
       correctPlayerRot =
    (Quaternion)stream. ReceiveNext();
   }
}
```

If you send something "ReliableDeltaCompressed", make sure to always write data to the stream in the same order. If you write no data to the **PhotonStream**, the update is not sent. This can be useful in pauses. Now on, to yet another way to communicate: RPCs.

### **Remote Procedure Calls**

Remote Procedure Calls (RPCs) are exactly what the name implies: methods that can be called on remote clients in the same room. To enable remote calls for a method of a MonoBehaviour, you must apply the attribute: [PunRPC]. A PhotonView instance is needed on the same GameObject, to call the marked functions.

```
[PunRPC]
void ChatMessage(string a, string b)
{
    Debug.Log("ChatMessage " + a + " " + b);
}
```

To call the method from any script, you need access to a **PhotonView** object. If your script derives from **Photon.MonoBehaviour**, it has a photonView field. Any regular MonoBehaviour or GameObject can use: PhotonView.Get(this) to get access to its **PhotonView** component and then call RPCs on it.

```
PhotonView photonView = PhotonView.Get(this);
photonView.RPC("ChatMessage", PhotonTargets.All,
    "jup", "and jup!");
```

So, instead of directly calling the target method, you call RPC() on a **PhotonView**. Provide the name of the method to call, which players should call the method and then provide a list of parameters.

Careful: The parameters list used in RPC() has to match the number of expected parameters! If the receiving client can't find a matching method, it will log an error. There is one exception to this rule: The last parameter of a RPC method can be of type **PhotonMessageInfo**, which will provide some context for each call.

## Timing for RPCs and Loading Levels

RPCs are called on specific PhotonViews and always target the matching one on the remote client. If the remote client does not know the fitting **PhotonView**, the RPC is lost.

A typical cause for lost RPCs is when clients load and set up levels. One client is faster or in the room for a longer time and sends important RPCs for objects that are not yet loaded on the other clients. The same happens when RPCs are buffered.

The solution is to pause the message queue, during scene loading. This code shows how how you can do it:

```
private IEnumerator MoveToGameScene()
{
  // Temporary disable processing of futher
    network messages
  PhotonNetwork.isMessageQueueRunning = false;
  Application.LoadLevel(levelName);
}
```

Alternatively you can use **PhotonNetwork.LoadLevel**. It temporarily disables the message queue as well.

Disabling the message queue will delay incoming and outgoing messages until the queue is unlocked. Obviously, it's very important to unlock the queue when you're ready to go on.

RPCs that belonged to the previously loaded scene but still arrived will now be discarded. But you should be able to define a break between both scenes by RPC.

## **Various topics**

## **Differences to Unity Networking**

#### 1. Host model

- Unity networking is server-client based (NOT P2P!). Servers are run via a Unity client (so via one of the players)
- Photon is server-client based as well, but has a dedicated server; No more dropped connections due to hosts leaving.

#### 2. Connectivity

- Unity networking works with NAT punchthrough to try to improve connectivity: since players host the network servers, the connection often fails due to firewalls/routers etc.
   Connectivity can never be guaranteed, there is a low success rate.
- Photon has a dedicated server, there is no need for NAT punchthrough or other concepts. Connectivity is a guaranteed 100%. If, in the rare case, a connection fails it must be due to a very strict client side network (a business VPN for example).

#### 3. Performance

 Photon beats Unity networking performance wise. We do not have the figures to prove this yet but the library has been optimized for years now. Furthermore, since the Unity servers are player hosted latency/ping is usually worse; you rely on the connection of the player acting as server. These connections are never any better then the connection of your dedicated **Photon** server.

#### 4. Price

Like the Unity Networking solution, the Photon Unity
Networking plugin is free as well. You can subscribe to use
Photon Cloud hosting service for your game. Alternatively,
you can rent your own servers and run Photon on them. The
free license enables up to 100 concurrent players. Other
licenses cost a one-time fee (as you do the hosting) and lift
the concurrent user limits.

#### 5. Features & maintenance

 Unity does not seem to give much priority to their Networking implementation. There are rarely feature improvements and bugfixes are as seldom. The **Photon** solution is actively maintained and parts of it are available with source code. Furthermore, **Photon** already offers more features than Unity, such as the built-in load balancing and offline mode.

#### 6. Master Server

 The Master Server for Photon is a bit different from the Master Server for plain Unity Networking: In our case, it's a Photon Server that lists room-names of currently played games in so called "lobbies". Like Unity's Master, it will forward clients to the Game Server(s), where the actual gameplay is done.

## **Instantiating Networked Objects**

In about every game you need to instantiate one or more player objects for every player. There are various options to do so which are listed below.

#### PhotonNetwork.Instantiate

PUN can automatically take care of spawning an object by passing a starting position, rotation and a prefab name to the **PhotonNetwork.Instantiate** method. Requirement: The prefab should be available directly under a Resources/ folder so that the prefab can be loaded at run time. Watch out with webplayers: Everything in the resources folder will be streamed at the very first scene per default. Under the webplayer settings you can specify the first level that uses assets from the Resources folder by using the "First streamed level". If you set this to your first game scene, your preloader and mainmenu will not be slowed down if they don't use the Resources folder assets.

```
void SpawnMyPlayerEverywhere()
{
    PhotonNetwork.Instantiate("MyPrefabName",
        new Vector3(0,0,0), Quaternion.identity,
        0);
    //The last argument is an optional group
        number, feel free to ignore it for now.
}
```

## Gain more control: Manually instantiate

If don't want to rely on the Resources folders to instantiate objects over the network you'll have to manually Instantiate objects as shown in the example at the end of this section. The main reason for wanting to instantiate manually is gaining control over what is downloaded when for streaming webplayers. The details about streaming and the Resources folder in Unity can be found here.

If you spawn manually, you will have to assign a PhotonViewID yourself, these viewID's are the key to routing network messages to the correct gameobject/scripts. The player who wants to own and spawn a new object should allocate a new viewID using **PhotonNetwork.AllocateViewID()**;. This PhotonViewID should then be send to all other players using a **PhotonView** that has already been set up (for example an existing scene **PhotonView**). You will have to keep in mind that this RPC needs to be buffered so that any clients that connect later will also receive the spawn instructions. Then the RPC message that is used to spawn the object will need a reference to your desired prefab and instantiate this using Unity's GameObject.Instantiate. Finally you will need to set setup the PhotonViews attached to this prefab by assigning all PhotonViews a PhotonViewID.

```
void SpawnMyPlayerEverywhere()
{
 //Manually allocate PhotonViewID
    PhotonViewID id1 =
    PhotonNetwork.AllocateViewID();
    photonView.RPC("SpawnOnNetwork",
    PhotonTargets.AllBuffered,
     transform.position,
        transform.rotation, id1,
    PhotonNetwork.player);
}
public Transform playerPrefab; //set this in the
     inspector
[PunRPC]
void SpawnOnNetwork(Vector3 pos, Quaternion rot,
     PhotonViewID id1, <a href="PhotonPlayer">PhotonPlayer</a> np)
{
```

```
Transform newPlayer =
    Instantiate(playerPrefab, pos, rot) as
    Transform;

//Set the PhotonView
PhotonView[] nViews =
    go.GetComponentsInChildren<PhotonView>();
    nViews[0].viewID = id1;
}
```

If you want to use asset bundles to load your network objects from, all you have to do is add your own assetbundle loading code and replace the "playerPrefab" from the example with the prefab from your asset bundle.

## Offline mode

Offline mode is a feature to be able to re-use your multiplayer code in singleplayer game modes as well.

Mike Hergaarden: At M2H we had to rebuild our games several times as game portals usually require you to remove multiplayer functionality completely. Furthermore, being able to use the same code for single and multiplayer saves a lot of work on itself.

The most common features that you'll want to be able to use in singleplayer are sending RPCs and using

**PhotonNetwork.Instantiate**. The main goal of offline mode is to disable nullreferences and other errors when using **PhotonNetwork** functionality while not connected. You would still need to keep track of the fact that you're running a singleplayer game, to set up the game etc. However, while running the game, all code should be reusable.

You need to manually enable offline mode, as **PhotonNetwork** needs to be able to distinguish erroneous from intended behaviour. Enabling this feature is very easy:

PhotonNetwork.offlineMode = true;

You can now reuse certain multiplayer methods without generating any connections and errors. Furthermore there is no noticeable overhead. Below follows a list of **PhotonNetwork** functions and variables and their results during offline mode:

PhotonNetwork.player The player ID is always -1
PhotonNetwork.playerName Works as expected.
PhotonNetwork.playerList Contains only the local player
PhotonNetwork.otherPlayers Always empty PhotonNetwork.time
returns Environment.TickCount or a more precise timer if enabled;
PhotonNetwork.isMasterClient Always true
PhotonNetwork.AllocateViewID() Works as expected.

PhotonNetwork.Instantiate Works as expected
PhotonNetwork.Destroy Works as expected.
PhotonNetwork.RemoveRPCs/RemoveRPCsInGroup/SetReceivingEn
While these make no sense in Singleplayer, they will not hurt either.
PhotonView.RPC Works as expected.

Note that using other methods than the ones above can yield unexpected results and some will simply do nothing. E.g. **PhotonNetwork.room** will, obviously, return null. If you intend on starting a game in singleplayer, but move it to multiplayer at a later stage, you might want to consider hosting a 1 player game instead; this will preserve buffered RPCs and Instantiation calls, whereas offline mode Instantiations will not automatically carry over after Connecting.

Either set **PhotonNetwork.offlineMode** = false; or Simply call Connect() to stop offline mode.

## Limitations

### Views and players

For performance reasons, the **PhotonNetwork** API supports up to 1000 PhotonViews per player and a maximum of 2,147,483 players (note that this is WAY higher than your hardware can support!). You can easily allow for more PhotonViews per player, at the cost of maximum players. This works as follows: PhotonViews send out a viewID for every network message. This viewID is an integer and it is composed of the player ID and the player's view ID. The maximum size of an int is 2,147,483,647, divided by our MAX VIEW IDS(1000) that allows for over 2 million players, each having 1000 view IDs. As you can see, you can easily increase the player count by reducing the MAX VIEW IDS. The other way around, you can give all players more VIEW IDS at the cost of less maximum players. It is important to note that most games will never need more than a few view ID's per player (one or two for the character..and that's usually it). If you need much more then you might be doing something wrong! It is extremely inefficient to assign a **PhotonView** and ID for every bullet that your weapon fires, instead keep track of your fire bullets via the player or weapon's PhotonView.

There is room for improving your bandwidth performance by reducing the int to a short (value range: -32,768 to 32,768). By setting MAX\_VIEW\_IDS to 32 you can then still support 1023 players Search for "//LIMITS NETWORKVIEWS&PLAYERS" for all occurrences of the int viewID. Furthermore, currently the API is not using uint/ushort but only the positive range of the numbers. This is done for simplicity and the usage of viewIDs is not a crucial performance issue for most situations.

### **Groups and Scoping**

The **PhotonNetwork** plugin does not support network groups fully. See above: "Using Groups in PUN".

Unity's "scope" feature is not implemented.

## **Feedback**

We are interested in your feedback, as this solution is an ongoing project for us. Let us know if something was too hidden, missing or not working. To let us know, post in our Forum: forum.exitgames.com

## F.A.Q.

## Can I use multiple PhotonViews per GameObject? Why?

Yes this is perfectly fine. You will need multiple PhotonViews if you need to observe 2 or more targets; You can only observe one per **PhotonView**. For your RPC's you'll only ever need one **PhotonView** and this can be the same **PhotonView** that is already observing something. RPC's never clash with an observed target.

## Can I use UnityScript / Javascript?

To use PUN from UnityScript, move both folders "PhotonNetwork" and "UtilityScripts" to the Assets\ folder. Now PUN compiles before UnityScript and that makes it available from regular UnityScript code.

# Converting your Unity networking project to Photon

Converting your Unity networking project to **Photon** can be done in one day. Just to be sure, make a backup of your project, as our automated converter will change your scripts. After this is done, run the converter from the **Photon** editor window (Window -> **Photon** Unity Networking -> Converter -> Start). The automatic conversion takes between 30 seconds to 10 minutes, depending on the size of your project and your computers performance. This automatic conversion takes care of the following:

- All NetworkViews are replaced with PhotonViews and the exact same settings. This is applied for all scenes and all prefabs.
- All scripts (JS/BOO/C#) are scanned for Network API calls, and they are replaced with **PhotonNetwork** calls.

There are some minor differences, therefore you will need to manually fix a few script conversion bugs. After conversion, you will most likely see some compile errors. You'll have to fix these first. Most common compile errors:

#### PhotonNetwork.RemoveRPCs(player);

PhotonNetwork.DestroyPlayerObjects(player); These do not exist, and can be safely removed. **Photon** automatically cleans up players when they leave (even though you can disable this and take care of cleanup yourself if you want to) ..CloseConnection takes '2' arguments...

Remove the second, boolean, argument from this call.

PhotonNetwork.GetPing(player); GetPing does not take any arguments, you can only request the ping to the photon server, not ping to other players. myPlayerClass.transform.photonView.XXX error You will need to convert code like this to:

myPlayerClass.transform.GetComponent<PhotonView>().XXX Inside of scripts, you can use photonView to get the attached **PhotonView** component. However, you cannot call this on an external transform directly. RegisterServer There's no more need to register your games

to a masterserver, **Photon** does this automatically.

You should be able to fix all compile errors in 5-30 minutes. Most errors will originate from main menu/GUI code, related to IPs/Ports/Lobby GUI.

This is where **Photon** differs most from Unity's solution:

There is only one **Photon** server and you connect using the room names. Therefore all references to IPs/ports can be removed from your code (usually GUI code). **PhotonNetwork.JoinRoom(string room)** only takes a room argument, you'll need to remove your old IP/port/NAT arguments. If you have been using the "Ultimate Unity networking project" by M2H, you should remove the MultiplayerFunctions class.

Lastly, all old MasterServer calls can be removed. You never need to register servers, and fetching the room list is as easy as calling **PhotonNetwork.GetRoomList()**. This list is always up to date (no need to fetch/poll etc). Rewriting the room listing can be most work, if your GUI needs to be redone, it might be simpler to write the GUI from scratch.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



## Photon Unity Networking

v1.81

**Main Page** 

**Related Pages** 

Modules

Classes

**Files** 

## **Network Simulation GUI**

Simple GUI element to control the built-in network condition simulation.

The Photon client library can simulate network conditions for lag (message delay) and loss, which can be a good tool for developer when testing with a local server or on near perfect network conditions.

To use it, add the component PhotonNetSimSettingsGui to an enabled GameObject in your scene. At runtime, the top left of the screen shows the current roundtrip time (RTT) and the controls for network simulation:

- RTT: The roundtrip time is the average of milliseconds until a message was acknowledged by the server. The variance value (behind the +/-) shows how stable the rtt is (a lower value being better).
- "Sim" toggle: Enables and disables the simulation. A sudden, big change of network conditions might result in disconnects.
- "Lag" slider: Adds a fixed delay to all outgoing and incoming messages. In milliseconds.
- "Jit" slider: Adds a random delay of "up to X milliseconds" per message.
- "Loss" slider: Drops the set percentage of messages. You can expect less than 2% drop in the internet today.



## Photon Unity Networking

v1.81

**Main Page** 

**Related Pages** 

Modules

Classes

**Files** 

## **Network Statistics GUI**

The PhotonStatsGui is a simple GUI component to track and show network-metrics at runtime.

## **Usage**

Just add the **PhotonStatsGui** component to any active GameObject in the hierarchy. A window appears (at runtime) and shows the message count.

A few toggles let you configure the window:

- buttons: Show buttons for "stats on", "reset stats" and "to log"
- traffic: Show lower level network traffic (bytes per direction)
- health: Show timing of sending, dispatches and their longest gaps

## **Message Statistics**

The top most values showns are counter for "messages". Any operation, response and event are counted. Shown are the total count of outgoing, incoming and the sum of those messages as total and as average for the timespan that is tracked.

#### **Traffic Statistics**

These are the byte and packet counters. Anything that leaves or arrives via network is counted here. Even if there are few messages, they could be huge by accident and still cause less powerful clients to

drop connection. You also see that there are packages sent when you don't send messages. They keeps the connection alive.

#### **Health Statistics**

The block beginning with "longest delta between" is about the performance of your client. We measure how much time passed between consecutive calls of send and dispatch. Usually they should be called ten times per second. If these values go beyond one second, you should check why Update() calls are delayed.

#### **Button "Reset"**

This resets the stats but keeps tracking them. This is useful to track message counts for different situations.

## **Button "To Log"**

Pressing this simply logs the current stat values. This can be useful to have a overview how things evolved or just as reference.

## **Button "Stats On" (Enabling Traffic Stats)**

The Photon library can track various network statistics but usually this feature is turned off. The PhotonStatsGui will enable the tracking and show those values.

The "stats on" toggle in the Gui controls if traffic stats are collected at all. The "Traffic Stats On" checkbox in the Inspector is the same value.



## Photon Unity Networking

v1.81

**Main Page** 

**Related Pages** 

Modules

Classes

**Files** 

## **Public API Module**

The Public API module rounds up the most commonly used classes of PUN.

These classes are grouped into a "module" to make it easier to find the important stuff in PUN. Classes like **PhotonNetwork** and **Photon.PunBehaviour** are good entry points to learn how to code with PUN.

Opposed to that, there are several classes that are for internal use by the PUN framework. Even some of the internally used classes are public. This is for ease of use and in parts a result of how Unity works.

**Open the Public API module** 

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



## Photon Unity Networking

v1.81

Main Page Related Pages	Modules	Classes	Files
-------------------------	---------	---------	-------

## **Modules**

Here is a list of all modules:

Public API	Groups the most important classes that you need to understand early on
Optional Gui Elements	Useful GUI elements for PUN

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page Related Pages Modules Classes Files

Classes | Enumerations | Functions

Public API

Groups the most important classes that you need to understand early on.  $\underline{\mathsf{More}}...$ 

#### Classes

#### interface IPunObservable

Defines the OnPhotonSerializeView method to make it easy to implement correctly for observable scripts. More...

#### interface IPunCallbacks

This interface is used as definition of all callback methods of PUN, except OnPhotonSerializeView. Preferably, implement them individually. <u>More...</u>

#### class Photon.PunBehaviour

This class provides a .photonView and all callbacks/events that PUN can call. Override the events/methods you want to use. More...

#### struct **PhotonMessageInfo**

Container class for info about a particular message, RPC or update. More...

#### class PhotonStream

This container is used in **OnPhotonSerializeView()** to either provide incoming data of a **PhotonView** or for you to provide it. <u>More...</u>

#### class **PhotonNetwork**

The main class to use the **PhotonNetwork** plugin. This class is static. More...

#### class **PhotonPlayer**

Summarizes a "player" within a room, identified (in that room) by actorID. More...

#### class **PhotonView**

PUN's NetworkView replacement class for networking. Use it like a NetworkView. More...

#### class Room

This class resembles a room that PUN joins (or joined). The properties are settable as opposed to those of a **RoomInfo** and you can close or hide "your" room. More...

#### class RoomInfo

A simplified room with just the info required to list and join, used for the room listing in the lobby. The properties are not settable (open, MaxPlayers, etc). More...

#### **Enumerations**

PhotonNetworkingMessage { PhotonNetworkingMessage.OnConnectedToPhoton. PhotonNetworkingMessage.OnLeftRoom. PhotonNetworkingMessage.OnMasterClientSwitched. PhotonNetworkingMessage.OnPhotonCreateRoomFailed, PhotonNetworkingMessage.OnPhotonJoinRoomFailed, PhotonNetworkingMessage.OnCreatedRoom, PhotonNetworkingMessage.OnJoinedLobby, PhotonNetworkingMessage.OnLeftLobby, PhotonNetworkingMessage.OnDisconnectedFromPhoton. PhotonNetworkingMessage.OnConnectionFail. PhotonNetworkingMessage.OnFailedToConnectToPhoton. PhotonNetworkingMessage.OnReceivedRoomListUpdate, PhotonNetworkingMessage.OnJoinedRoom, PhotonNetworkingMessage.OnPhotonPlayerConnected, PhotonNetworkingMessage.OnPhotonPlayerDisconnected, enum PhotonNetworkingMessage.OnPhotonRandomJoinFailed, PhotonNetworkingMessage.OnConnectedToMaster. PhotonNetworkingMessage.OnPhotonSerializeView. PhotonNetworkingMessage.OnPhotonInstantiate. PhotonNetworkingMessage.OnPhotonMaxCccuReached. PhotonNetworkingMessage.OnPhotonCustomRoomPrope PhotonNetworkingMessage.OnPhotonPlayerPropertiesCha PhotonNetworkingMessage.OnUpdatedFriendList, PhotonNetworkingMessage.OnCustomAuthenticationFailed PhotonNetworkingMessage.OnCustomAuthenticationRes PhotonNetworkingMessage.OnWebRpcResponse. PhotonNetworkingMessage.OnOwnershipReguest. PhotonNetworkingMessage.OnLobbyStatisticsUpdate. PhotonNetworkingMessage.OnPhotonPlayerActivityChan PhotonNetworkingMessage.OnOwnershipTransfered This enum defines the set of MonoMessages **Photon** Unity Net using as callbacks. Implemented by PunBehaviour. More...

enum

PhotonLogLevel { PhotonLogLevel.ErrorsOnly, PhotonLogLevel.Informational, PhotonLogLevel.Full }

Used to define the level of logging output created by the PUN cl log errors, info (some more) or full. <u>More...</u>

#### PhotonTargets { PhotonTargets.All, PhotonTargets.Others, PhotonTargets. PhotonTargets.AllBuffered. enum PhotonTargets.OthersBuffered, PhotonTargets.AllViaServe PhotonTargets.AllBufferedViaServer Enum of "target" options for RPCs. These define which remote your RPC call. More... ClientState { ClientState.Uninitialized, ClientState.PeerCreated, ClientSt ClientState.Authenticated. ClientState.JoinedLobby, ClientState.DisconnectingFromM ClientState.ConnectingToGameserver, ClientState.ConnectedToGameserver. ClientState.Joining, ClientState.Joined, ClientState.Leavin ClientState.DisconnectingFromGameserver, enum ClientState.ConnectingToMasterserver, ClientState.QueuedComingFromGameserver, ClientState.D ClientState.Disconnected. ClientState.ConnectedToMaster, ClientState.ConnectingTo ClientState.ConnectedToNameServer, ClientState.DisconnectingFromNameServer, ClientState.Authenticating Detailed connection / networking peer state. PUN implements a loadbalancing and authentication workflow "behind the scenes". states will automatically advance to some follow up state. Those

#### DisconnectCause {

DisconnectCause.DisconnectByServerUserLimit = StatusCode.DisconnectByServerUserLimit,
DisconnectCause.ExceptionOnConnect = StatusCode.ExceptionOnConnect,
DisconnectCause.DisconnectByServerTimeout =

commented with "(will-change)". More...

StatusCode.DisconnectByServer, DisconnectCause.DisconnectByServerLogic = StatusCode.DisconnectByServerLogic, **DisconnectCause.Exception** = StatusCode.Exception, enum DisconnectCause.InvalidAuthentication = ErrorCode.Invalid/ **DisconnectCause.MaxCcuReached** = ErrorCode.MaxCcuRea **DisconnectCause.InvalidRegion** = ErrorCode.InvalidRegion, DisconnectCause.SecurityExceptionOnConnect = StatusCode.SecurityExceptionOnConnect, DisconnectCause.DisconnectByClientTimeout = StatusCode.TimeoutDisconnect, DisconnectCause.InternalReceiveException = StatusCode.ExceptionOnReceive, **DisconnectCause.AuthenticationTicketExpired** = 32753 Summarizes the cause for a disconnect, Used in: OnConnection OnFailedToConnectToPhoton, More...

### **Functions**

#### void

# IPunObservable.OnPhotonSerializeView (PhotonStream stream, PhotonMessageInfo info)

Called by PUN several times per second, so that your script can write and read synchronization data for the **PhotonView**. More...

### **Detailed Description**

Groups the most important classes that you need to understand early on.

### **Enumeration Type Documentation**

#### enum ClientState

Detailed connection / networking peer state. PUN implements a loadbalancing and authentication workflow "behind the scenes", so some states will automatically advance to some follow up state. Those states are commented with "(will-change)".

Enumerator	
Uninitialized	Not running. Only set before initialization and first use.
PeerCreated	Created and available to connect.
Queued	Not used at the moment.
Authenticated	The application is authenticated. PUN usually joins the lobby now.  (will-change) Unless AutoJoinLobby is false.
JoinedLobby	Client is in the lobby of the Master Server and gets room listings.  Use Join, Create or JoinRandom to get into a room to play.
DisconnectingFromMasterserver	Disconnecting. (will-change)
ConnectingToGameserver	Connecting to game server (to join/create a room and play).

	(will-change)
ConnectedToGameserver	Similar to Connected state but on game server. Still in process to join/create room.
	(will-change)
Joining	In process to join/create room (on game server).
	(will-change)
Joined	Final state of a room join/create sequence. This client can now exchange events / call RPCs with other clients.
Leaving	Leaving a room.
	(will-change)
DisconnectingFromGameserver	Workflow is leaving the game server and will re-connect to the master server.  (will-change)
ConnectingToMasterserver	Workflow is connected to master server and will establish encryption and authenticate your app.  (will-change)
QueuedComingFromGameserver	Same Queued but coming from game server.  (will-change)
Disconnecting	PUN is disconnecting. This leads to Disconnected.
Diagram and I	(will-change)
Disconnected	No connection is setup, ready

	to connect. Similar to PeerCreated.
ConnectedToMaster	Final state for connecting to master without joining the lobby (AutoJoinLobby is false).
ConnectingToNameServer	Client connects to the NameServer. This process includes low level connecting and setting up encryption. When done, state becomes ConnectedToNameServer.
ConnectedToNameServer	Client is connected to the NameServer and established enctryption already. You should call OpGetRegions or ConnectToRegionMaster.
DisconnectingFromNameServer	When disconnecting from a <b>Photon</b> NameServer. (will-change)
Authenticating	When connecting to a <b>Photon</b> Server, this state is intermediate before you can call any operations.
	(will-change)

#### enum DisconnectCause

Summarizes the cause for a disconnect. Used in: OnConnectionFail and OnFailedToConnectToPhoton.

Extracted from the status codes from ExitGames.Client.Photon.StatusCode.

#### See also

PhotonNetworkingMessage

Enumerator	
DisconnectByServerUserLimit	Server actively disconnected this client. Possible cause: The server's user limit was hit and client was forced to disconnect (on connect).
ExceptionOnConnect	Connection could not be established. Possible cause: Local server not running.
DisconnectByServerTimeout	Timeout disconnect by server (which decided an ACK was missing for too long).
DisconnectByServerLogic	Server actively disconnected this client. Possible cause: Server's send buffer full (too much data for client).
Exception	Some exception caused the connection to close.
InvalidAuthentication	(32767) The <b>Photon</b> Cloud rejected the sent Appld. Check your Dashboard and make sure the Appld you use is complete and correct.
MaxCcuReached	(32757) Authorization on the <b>Photon</b> Cloud failed because the concurrent users (CCU) limit of the app's subscription is reached.
InvalidRegion	(32756) Authorization on the <b>Photon</b> Cloud failed because the app's subscription does not allow to use a particular region's server.
SecurityExceptionOnConnect	The security settings for client or server don't allow a connection (see remarks).
	A common cause for this is that browser clients read a

	"crossdomain" file from the server. If that file is unavailable or not configured to let the client connect, this exception is thrown.  Photon usually provides this crossdomain file for Unity. If it fails, read: <a href="http://doc.exitgames.com/photon-server/PolicyApp">http://doc.exitgames.com/photon-server/PolicyApp</a>
DisconnectByClientTimeout	Timeout disconnect by client (which decided an ACK was missing for too long).
InternalReceiveException	Exception in the receive-loop. Possible cause: Socket failure.
AuthenticationTicketExpired	(32753) The Authentication ticket expired. Handle this by connecting again (which includes an authenticate to get a fresh ticket).

#### enum PhotonLogLevel

Used to define the level of logging output created by the PUN classes. Either log errors, info (some more) or full.

Enumerator	
ErrorsOnly	Show only errors. Minimal output. Note: Some might be "runtime errors" which you have to expect.
Informational	Logs some of the workflow, calls and results.
Full	Every available log call gets into the console/log. Only use for debugging.

#### enum PhotonNetworkingMessage

This enum defines the set of MonoMessages Photon Unity Networking

Much like "Update()" in Unity, PUN will call methods in specific situation operations complete (example: when joining a room).

All those methods are defined and described in this enum and impleme implement them as override).

Each entry is the name of such a method and the description tells you v

Make sure to read the remarks per entry as some methods have option

Enumerator	
OnConnectedToPhoton	Called when the initial conserver. <b>OnJoinedLobby</b> ready.
	This callback is only use (technically). Most often, OnFailedToConnectTo
	OnJoinedLobby() or Oi
	When this is called, the I your Appld, the user, etc from the masterserver to
	Example: void OnConne
OnLeftRoom	Called when the local us
	When leaving a room, P can use lobbies and join OnConnectedToMaste
	Example: void OnLeftRo
OnMasterClientSwitched	Called after switching to
	This is not called when t in the player list when th
	Example: void OnMaste
OnPhotonCreateRoomFailed	Called when a CreateRo

	ErrorCode and messag
	Most likely because the faster than you). PUN lo PhotonLogLevel.Informa
	Example: void <b>OnPhoto</b>
	Example: void OnPhotor codeAndMsg[0] is short
OnPhotonJoinRoomFailed	Called when a <b>JoinRoo</b> and message.
	Most likely error is that tl client was faster than yo is >= PhotonLogLevel.In
	Example: void <b>OnPhoto</b>
	Example: void OnPhotor codeAndMsg[0] is short
OnCreatedRoom	Called when this client c called as well.
	This callback is only call <b>PhotonNetwork.Create</b>
	As any client might close the creator of a room do
	If you need specific roon OnMasterClientSwitch room's state.
	Example: void OnCreate
OnJoinedLobby	Called on entering a lobl will call <b>OnReceivedRo</b>
	Note: When PhotonNet OnConnectedToMaster

	available.
	While in the lobby, the royou can't modify). The roonReceivedRoomListl
	Example: void <b>OnJoine</b> (
OnLeftLobby	Called after leaving a lok
	When you leave a lobby refer to the default lobby
	Example: void OnLeftLc
OnDisconnectedFromPhoton	Called after disconnectir
	In some cases, other cal is called. Examples: On OnFailedToConnectTo
	Example: void <b>OnDisco</b>
OnConnectionFail	Called when something followed by a call to <b>On!</b>
	If the server could not be OnFailedToConnectToPl provided as StatusCode
	Example: void OnConne
OnFailedToConnectToPhoton	Called if a connect call to established, followed by
	OnConnectionFail only ç established in the first pl
	Example: void OnFailed
OnReceivedRoomListUpdate	Called for any update of (PhotonNetwork.inside
	PUN provides the list of Each item is a <b>RoomInf</b>

	defined those as lobby-li
	Not all types of lobbies p and specialized for serve
	Example: void <b>OnRecei</b>
OnJoinedRoom	Called when entering a r (including the Master Cli
	This method is common to be started "actively", y button-press or a timer.
	When this is called, you room via <b>PhotonNetwo</b> l already available as <b>Roc</b> find out if enough players
	Example: void OnJoine
OnPhotonPlayerConnected	Called when a remote pl added to the playerlist at
	If your game starts with a to check the <b>Room.play</b>
	Example: void OnPhotor
OnPhotonPlayerDisconnected	Called when a remote pl removed from the player
	When your client calls P the remaining clients. W this callback gets execut
	Example: void OnPhotor
OnPhotonRandomJoinFailed	Called after a JoinRando and message.
	Most likely all rooms are lobbies (via JoinLobby o rooms. PUN logs some i

	PhotonLogLevel.Informa
	Example: void <b>OnPhoto</b>
	Example: void OnPhotor codeAndMsg[0] is short
OnConnectedToMaster	Called after the connectionly when <b>PhotonNetw</b>
	If you set <b>PhotonNetwo</b> called instead of this.
	You can join rooms and lobby is used in that cas unless you join a lobby \(\circ\)
	Example: void <b>OnConne</b>
OnPhotonSerializeView	Implement to customize every 'network-update' v
	This method will be calle of a <b>PhotonView</b> . <b>Photo</b> method is called. <b>Photo</b> sent by this client.
	Implementing this methor regularly synchronizes. 'your data is used by rec
	Unlike other callbacks, C assigned to a <b>PhotonVi</b>
	To make use of this metl "writing" mode" on the cl (PhotonStream.isWriting that just receive that the
	If you skip writing any vacarefully, this can conseroom/second).
	I

	Note that OnPhotonSerial sender does not send ar Update()".
	Example: void OnPhotor PhotonMessageInfo info
OnPhotonInstantiate	Called on all scripts on a using <b>PhotonNetwork.I</b>
	PhotonMessageInfo pa when (based off PhotonI
	Example: void OnPhotor
OnPhotonMaxCccuReached	Because the concurrent rejected by the server ar
	When this happens, the in OnPhotonMaxCcuRea raise the CCU limits with subscription (when using when the CCU limit was (webpage).
	Example: void <b>OnPhoto</b>
OnPhotonCustomRoomPropertiesChanged	Called when a room's cu contains all that was set
	Since v1.25 this method Changing properties mu causes this callback loca
	Example: void OnPhotor propertiesThatChanged)
OnPhotonPlayerPropertiesChanged	Called when custom play properties are passed as
	Since v1.25 this method which contains two entri- [0] is the affected <b>Photo</b> [1] is the Hashtable of pr

	We are using a object[] ( (which has only one opti
	Changing properties mu which causes this callba
	Example:
	void OnPhotonPlayer PhotonPlayer pl Hashtable props // }
OnUpdatedFriendList	Called when the server
	The friends list is availat state and the room a use
	Example: void OnUpdat
OnCustomAuthenticationFailed	Called when the custom
	Custom Authentication c authentication is success OnJoinedLobby() or Or
	During development of $\varepsilon$ the server side. In those
	Unless you setup a customashboard), this won't b
	Example: void OnCustor
OnCustomAuthenticationResponse	Called when your Custo
	Custom Authentication s response. When present Dictionary. While the key either string or a number

	type is the one you expe
	Example: void OnCustor data) { }
	https://doc.photonengineauthentication
OnWebRpcResponse	Called by PUN when the PhotonNetwork.WebRP(
	Important: The response web-service. The conter can create a WebRespo webResponse = new <b>W</b> <sub>1</sub>
	Please note: Class Oper "used": using <b>ExitGame</b> other classes)
	The OperationResponse Service not configured" now have RPC path/nan
	Example: void OnWebR
OnOwnershipRequest	Called when another pla current owner).
	The parameter viewAnd
	PhotonView view = viev
	PhotonPlayer requestin
	void OnOwnershipRequ
OnLobbyStatisticsUpdate	Called when the Master PhotonNetwork.Lobby
	This callback has two pr before this client connec Server, which is providin

OnPhotonPlayerActivityChanged	Called when a remote <b>P</b> is PlayerTtl is greater the		
	Use <b>0. If true, the playe</b> current activity state		
	Example: void OnPhotor		
	This callback has precor		
OnOwnershipTransfered	Called when a <b>PhotonV</b>		
	The parameter viewAnd		
	PhotonView view = viev		
	PhotonPlayer newOwn		
	PhotonPlayer oldOwne		
	void OnOwnershipTrans		

### enum PhotonTargets

Enum of "target" options for RPCs. These define which remote clients get your RPC call.

Enumerator					
All	Sends the RPC to everyone else and executes it immediately on this client. Player who join later will not execute this RPC.				
Others	Sends the RPC to everyone else. This client does not execute the RPC. Player who join later will not execute this RPC.				
MasterClient	Sends the RPC to MasterClient only. Careful: The MasterClient might disconnect before it executes the RPC and that might cause dropped RPCs.				

AllBuffered	Sends the RPC to everyone else and executes it immediately on this client. New players get the RPC when they join as it's buffered (until this client leaves).
OthersBuffered	Sends the RPC to everyone. This client does not execute the RPC. New players get the RPC when they join as it's buffered (until this client leaves).
AllViaServer	Sends the RPC to everyone (including this client) through the server.  This client executes the RPC like any other when it received it from the server. Benefit: The server's order of sending the RPCs is the same on all clients.
AllBufferedViaServer	Sends the RPC to everyone (including this client) through the server and buffers it for players joining later.  This client executes the RPC like any other when it received it from the server. Benefit: The server's order of sending the RPCs is the same on all clients.

#### **Function Documentation**

# void IPunObservable.OnPhotonSerializeView ( PhotonStream stre PhotonMessageInfo info )

Called by PUN several times per second, so that your script can write a read synchronization data for the **PhotonView**.

This method will be called in scripts that are assigned as Observed component of a **PhotonView**.

**PhotonNetwork.sendRateOnSerialize** affects how often this method i called.

**PhotonNetwork.sendRate** affects how often packages are sent by this client.

Implementing this method, you can customize which data a **PhotonVie** regularly synchronizes. Your code defines what is being sent (content) a how your data is used by receiving clients.

Unlike other callbacks, *OnPhotonSerializeView only gets called when it assigned to a PhotonView* as PhotonView.observed script.

To make use of this method, the **PhotonStream** is essential. It will be in "writing" mode" on the client that controls a PhotonView (PhotonStream.isWriting == true) and in "reading mode" on the remote clients that just receive that the controlling client sends.

If you skip writing any value into the stream, PUN will skip the update. Used carefully, this can conserve bandwidth and messages (which have limit per room/second).

Note that OnPhotonSerializeView is not called on remote clients when t sender does not send any update. This can't be used as "x-times per second Update()".

# Implemented in PhotonAnimatorView, PhotonTransformView, PhotonRigidbody2DView, and PhotonRigidbodyView.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



### Photon Unity Networking

v1.81

Main Page	ge Related Pa		Pages	Modules		Classes	Files
Class List	C	Class Index Cla		erarchy	Clas	ss Members	

Public Member Functions | List of all members

# IPunObservable Interface Reference

Public API

Defines the OnPhotonSerializeView method to make it easy to implement correctly for observable scripts. <u>More...</u>

Inherited by PhotonAnimatorView, PhotonRigidbody2DView, PhotonRigidbodyView, and PhotonTransformView.

### **Public Member Functions**

#### void

# OnPhotonSerializeView (PhotonStream stream, PhotonMessageInfo info)

Called by PUN several times per second, so that your script can write and read synchronization data for the **PhotonView**. More...

### **Detailed Description**

Defines the OnPhotonSerializeView method to make it easy to implement correctly for observable scripts.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



### **Photon Unity Networking**

v1.81

Main Page	Page Related F		Pages	Modules		Classes	Files
Class List	C	Class Index	Class Hierarchy		Clas	ss Members	

Public Member Functions | List of all members

# IPunCallbacks Interface Reference

Public API

This interface is used as definition of all callback methods of PUN, except OnPhotonSerializeView. Preferably, implement them individually. <u>More...</u>

Inherited by Photon.PunBehaviour.

#### **Public Member Functions**

#### void OnConnectedToPhoton ()

Called when the initial connection got established but before you can use the server. **OnJoinedLobby()** or **OnConnectedToMaster()** are called when PUN is ready. More...

#### void OnLeftRoom ()

Called when the local user/client left a room. More...

#### void OnMasterClientSwitched (PhotonPlayer newMasterClient)

Called after switching to a new MasterClient when the current one leaves. More...

#### void OnPhotonCreateRoomFailed (object[] codeAndMsg)

Called when a **CreateRoom()** call failed. The parameter provides **ErrorCode** and message (as array). <u>More...</u>

#### void OnPhotonJoinRoomFailed (object[] codeAndMsg)

Called when a **JoinRoom()** call failed. The parameter provides **ErrorCode** and message (as array). <u>More...</u>

#### void OnCreatedRoom ()

Called when this client created a room and entered it. **OnJoinedRoom()** will be called as well. <u>More...</u>

#### void OnJoinedLobby ()

Called on entering a lobby on the Master Server. The actual room-list updates will call **OnReceivedRoomListUpdate()**. <u>More...</u>

#### void OnLeftLobby ()

Called after leaving a lobby. More...

#### void OnFailedToConnectToPhoton (DisconnectCause cause)

Called if a connect call to the **Photon** server failed before the connection was established, followed by a call to **OnDisconnectedFromPhoton()**. More...

#### void OnConnectionFail (DisconnectCause cause)

Called when something causes the connection to fail (after it was established), followed by a call to **OnDisconnectedFromPhoton()**. More...

#### void OnDisconnectedFromPhoton ()

Called after disconnecting from the **Photon** server. More...

#### void OnPhotonInstantiate (PhotonMessageInfo info)

Called on all scripts on a GameObject (and children) that have been Instantiated using **PhotonNetwork.Instantiate**. More...

#### void OnReceivedRoomListUpdate ()

Called for any update of the room-listing while in a lobby (**PhotonNetwork.insideLobby**) on the Master Server. More...

#### void OnJoinedRoom ()

Called when entering a room (by creating or joining it). Called on all clients (including the Master Client). More...

#### void **OnPhotonPlayerConnected (PhotonPlayer** newPlayer)

Called when a remote player entered the room. This **PhotonPlayer** is already added to the playerlist at this time. More...

# void OnPhotonPlayerDisconnected (PhotonPlayer otherPlayer) Called when a remote player left the room. This PhotonPlayer is already removed from the playerlist at this time. More...

### void OnPhotonRandomJoinFailed (object[] codeAndMsg)

Called when a JoinRandom() call failed. The parameter provides **ErrorCode** and message. <u>More...</u>

#### void OnConnectedToMaster ()

Called after the connection to the master is established and authenticated but only when **PhotonNetwork.autoJoinLobby** is false. More...

#### void OnPhotonMaxCccuReached ()

Because the concurrent user limit was (temporarily) reached, this client is rejected by the server and disconnecting. More...

### void OnPhotonCustomRoomPropertiesChanged (Hashtable propertiesThatChanged)

Called when a room's custom properties changed. The propertiesThatChanged contains all that was set via **Room.SetCustomProperties**. More...

### void OnPhotonPlayerPropertiesChanged (object[] playerAndUpdatedProps)

Called when custom player-properties are changed. Player and the changed properties are passed as object []. More...

#### void OnUpdatedFriendList ()

Called when the server sent the response to a FindFriends request and updated **PhotonNetwork.Friends**. More...

#### void **OnCustomAuthenticationFailed** (string debugMessage)

Called when the custom authentication failed. Followed by disconnect! More...

### void OnCustomAuthenticationResponse (Dictionary< string, object > data)

Called when your Custom Authentication service responds with additional data. More...

#### void **OnWebRpcResponse** (OperationResponse response)

Called by PUN when the response to a WebRPC is available. See PhotonNetwork.WebRPC. <u>More...</u>

#### void OnOwnershipRequest (object[] viewAndPlayer)

Called when another player requests ownership of a **PhotonView** from you (the current owner). More...

#### void OnLobbyStatisticsUpdate ()

Called when the Master Server sent an update for the Lobby Statistics, updating **PhotonNetwork.LobbyStatistics**. More...

- void OnPhotonPlayerActivityChanged (PhotonPlayer otherPlayer)
  Called when a remote Photon Player activity changed. This will
  be called ONLY if PlayerTtl is greater than 0. More...
- void **OnOwnershipTransfered** (object[] viewAndPlayers)
  Called when ownership of a **PhotonView** is transfered to another player. More...

### **Detailed Description**

This interface is used as definition of all callback methods of PUN, except OnPhotonSerializeView. Preferably, implement them individually.

This interface is available for completeness, more than for actually implementing it in a game. You can implement each method individually in any MonoMehaviour, without implementing **IPunCallbacks**.

PUN calls all callbacks by name. Don't use implement callbacks with fully qualified name. Example:

**IPunCallbacks.OnConnectedToPhoton** won't get called by Unity's SendMessage().

PUN will call these methods on any script that implements them, analog to Unity's events and callbacks. The situation that triggers the call is described per method.

OnPhotonSerializeView is NOT called like these callbacks! It's usage frequency is much higher and it is implemented in: **IPunObservable**.

#### **Member Function Documentation**

#### void IPunCallbacks.OnConnectedToMaster()

Called after the connection to the master is established and authenticated but only when **PhotonNetwork.autoJoinLobby** is false.

If you set **PhotonNetwork.autoJoinLobby** to true, **OnJoinedLobby()** will be called instead of this.

You can join rooms and create them even without being in a lobby. The default lobby is used in that case. The list of available rooms won't become available unless you join a lobby via PhotonNetwork.joinLobby.

Implemented in Photon.PunBehaviour.

#### void IPunCallbacks.OnConnectedToPhoton()

Called when the initial connection got established but before you can use the server. **OnJoinedLobby()** or **OnConnectedToMaster()** are called when PUN is ready.

This callback is only useful to detect if the server can be reached at all (technically). Most often, it's enough to implement **OnFailedToConnectToPhoton()** and **OnDisconnectedFromPhoton()**.

**OnJoinedLobby()** or **OnConnectedToMaster()** are called when PUN is ready.

When this is called, the low level connection is established and PUN will send your Appld, the user, etc in the background. This is not called for transitions from the masterserver to game servers.

Implemented in **Photon.PunBehaviour**.

### void IPunCallbacks.OnConnectionFail (DisconnectCause cause)

Called when something causes the connection to fail (after it was established), followed by a call to **OnDisconnectedFromPhoton()**.

If the server could not be reached in the first place, OnFailedToConnectToPhoton is called instead. The reason for the error is provided as DisconnectCause.

Implemented in Photon.PunBehaviour.

#### void IPunCallbacks.OnCreatedRoom ( )

Called when this client created a room and entered it. **OnJoinedRoom()** will be called as well.

This callback is only called on the client which created a room (see **PhotonNetwork.CreateRoom**).

As any client might close (or drop connection) anytime, there is a chance that the creator of a room does not execute OnCreatedRoom.

If you need specific room properties or a "start signal", it is safer to implement **OnMasterClientSwitched()** and to make the new MasterClient check the room's state.

Implemented in **Photon.PunBehaviour**.

### void IPunCallbacks.OnCustomAuthenticationFailed (string debugMess

Called when the custom authentication failed. Followed by disconnect!

Custom Authentication can fail due to user-input, bad tokens/secrets. If authentication is successful, this method is not called. Implement **OnJoinedLobby()** or **OnConnectedToMaster()** (as usual).

During development of a game, it might also fail due to wrong configura on the server side. In those cases, logging the debugMessage is very important.

Unless you setup a custom authentication service for your app (in the <u>Dashboard</u>), this won't be called!

#### **Parameters**

**debugMessage** Contains a debug message why authentication fa This has to be fixed during development time.

Implemented in Photon.PunBehaviour.

## void IPunCallbacks.OnCustomAuthenticationResponse ( Dictionary< st

Called when your Custom Authentication service responds with addition

Custom Authentication services can include some custom data in their present, that data is made available in this callback as Dictionary. While data have to be strings, the values can be either string or a number (in make extra sure, that the value type is the one you expect. Numbers be int64.

Example: void OnCustomAuthenticationResponse(Dictionary<string, ok https://doc.photonengine.com/en/realtime/current/reference/custom-aut Implemented in **Photon.PunBehaviour**.

#### void IPunCallbacks.OnDisconnectedFromPhoton ( )

Called after disconnecting from the **Photon** server.

In some cases, other callbacks are called before OnDisconnectedFromPhoton is called. Examples: OnConnectionFail() and OnFailedToConnectToPhoton().

Implemented in Photon.PunBehaviour.

#### void

#### IPunCallbacks.OnFailedToConnectToPhoton ( DisconnectCause c

Called if a connect call to the **Photon** server failed before the connectic was established, followed by a call to **OnDisconnectedFromPhoton()**.

This is called when no connection could be established at all. It differs f OnConnectionFail, which is called when an existing connection fails.

Implemented in Photon.PunBehaviour.

#### void IPunCallbacks.OnJoinedLobby ()

Called on entering a lobby on the Master Server. The actual roomlist updates will call **OnReceivedRoomListUpdate()**.

Note: When **PhotonNetwork.autoJoinLobby** is false, **OnConnectedToMaster()** will be called and the room list won't become available.

While in the lobby, the roomlist is automatically updated in fixed intervals (which you can't modify). The room list gets available when **OnReceivedRoomListUpdate()** gets called after **OnJoinedLobby()**.

Implemented in Photon.PunBehaviour.

#### void IPunCallbacks.OnJoinedRoom ( )

Called when entering a room (by creating or joining it). Called on all

clients (including the Master Client).

This method is commonly used to instantiate player characters. If a match has to be started "actively", you can call an **PunRPC** triggered by a user's button-press or a timer.

When this is called, you can usually already access the existing players in the room via **PhotonNetwork.playerList**. Also, all custom properties should be already available as **Room.customProperties**. Check **Room.playerCount** to find out if enough players are in the room to start playing.

Implemented in Photon.PunBehaviour.

#### void IPunCallbacks.OnLeftLobby ()

Called after leaving a lobby.

When you leave a lobby, **CreateRoom** and **JoinRandomRoom** automatically refer to the default lobby.

Implemented in Photon.PunBehaviour.

#### void IPunCallbacks.OnLeftRoom ( )

Called when the local user/client left a room.

When leaving a room, PUN brings you back to the Master Server. Before you can use lobbies and join or create rooms, OnJoinedLobby() or OnConnectedToMaster() will get called again.

Implemented in Photon.PunBehaviour.

#### void IPunCallbacks.OnLobbyStatisticsUpdate()

Called when the Master Server sent an update for the Lobby Statistics, updating **PhotonNetwork.LobbyStatistics**.

This callback has two preconditions: EnableLobbyStatistics must be set to true, before this client connects. And the client has to be connected to the Master Server, which is providing the info about lobbies.

Implemented in Photon.PunBehaviour.

## void IPunCallbacks.OnMasterClientSwitched ( PhotonPlayer newMaste

Called after switching to a new MasterClient when the current one leave

This is not called when this client enters a room. The former MasterClie still in the player list when this method get called.

Implemented in Photon.PunBehaviour.

## void IPunCallbacks.OnOwnershipRequest (object[] viewAndPlayer)

Called when another player requests ownership of a **PhotonView** from you (the current owner).

The parameter viewAndPlayer contains:

PhotonView view = viewAndPlayer[0] as PhotonView;

PhotonPlayer requestingPlayer = viewAndPlayer[1] as PhotonPlayer;

#### **Parameters**

**viewAndPlayer** The **PhotonView** is viewAndPlayer[0] and the requesting player is viewAndPlayer[1].

Implemented in Photon.PunBehaviour.

#### void

#### IPunCallbacks.OnOwnershipTransfered (object[] viewAndPlayers

Called when ownership of a **PhotonView** is transferred to another player.

The parameter viewAndPlayers contains:

PhotonView view = viewAndPlayers[0] as PhotonView;

PhotonPlayer newOwner = viewAndPlayers[1] as PhotonPlayer;

PhotonPlayer oldOwner = viewAndPlayers[2] as PhotonPlayer;

void OnOwnershipTransfered(object[] viewAndPlayers) {} //

Implemented in Photon.PunBehaviour.

#### void

#### IPunCallbacks.OnPhotonCreateRoomFailed (object[] codeAndMs

Called when a **CreateRoom()** call failed. The parameter provides **ErrorCode** and message (as array).

Most likely because the room name is already in use (some other client was faster than you). PUN logs some info if the **PhotonNetwork.logLevel** is >= PhotonLogLevel.Informational.

#### **Parameters**

**codeAndMsg** codeAndMsg[0] is short **ErrorCode** and codeAndMsg[1] is a string debug msg.

Implemented in Photon.PunBehaviour.

#### void

#### IPunCallbacks.OnPhotonCustomRoomPropertiesChanged (Hashta

Called when a room's custom properties changed. The propertiesThatC set via **Room.SetCustomProperties**.

Since v1.25 this method has one parameter: Hashtable propertiesThat( Changing properties must be done by **Room.SetCustomProperties**, w locally, too.

#### **Parameters**

propertiesThatChanged

Implemented in Photon.PunBehaviour.

#### void

IPunCallbacks.OnPhotonInstantiate (PhotonMessageInfo info)

Called on all scripts on a GameObject (and children) that have been Instantiated using **PhotonNetwork.Instantiate**.

**PhotonMessageInfo** parameter provides info about who created the object and when (based off PhotonNetworking.time).

Implemented in Photon.PunBehaviour.

#### void

IPunCallbacks.OnPhotonJoinRoomFailed ( object[] codeAndMsg )

Called when a **JoinRoom()** call failed. The parameter provides **ErrorCode** and message (as array).

Most likely error is that the room does not exist or the room is full (some other client was faster than you). PUN logs some info if the **PhotonNetwork.logLevel** is >= PhotonLogLevel.Informational.

#### **Parameters**

codeAndMsg codeAndMsg[0] is short ErrorCode and

#### codeAndMsg[1] is string debug msg.

Implemented in Photon.PunBehaviour.

#### void IPunCallbacks.OnPhotonMaxCccuReached ( )

Because the concurrent user limit was (temporarily) reached, this client is rejected by the server and disconnecting.

When this happens, the user might try again later. You can't create or join rooms in OnPhotonMaxCcuReached(), cause the client will be disconnecting. You can raise the CCU limits with a new license (when you host yourself) or extended subscription (when using the **Photon** Cloud). The **Photon** Cloud will mail you when the CCU limit was reached. This is also visible in the Dashboard (webpage).

Implemented in Photon.PunBehaviour.

## void IPunCallbacks.OnPhotonPlayerActivityChanged (PhotonPlayer ot

Called when a remote **Photon** Player activity changed. This will be called PlayerTtl is greater than 0.

Use **0.** If true, the player is not gett...">PhotonPlayer.IsInactive to cl player's current activity state.

Example: void **OnPhotonPlayerActivityChanged(PhotonPlayer othe** {...}

This callback has precondition: PlayerTtl must be greater than 0.

Implemented in Photon.PunBehaviour.

## void IPunCallbacks.OnPhotonPlayerConnected (PhotonPlayer newPlay

Called when a remote player entered the room. This **PhotonPlayer** is already added to the playerlist at this time.

If your game starts with a certain number of players, this callback can be useful to check the **Room.playerCount** and find out if you can start.

Implemented in Photon.PunBehaviour.

# void IPunCallbacks.OnPhotonPlayerDisconnected (PhotonPlayer other

Called when a remote player left the room. This **PhotonPlayer** is alread removed from the playerlist at this time.

When your client calls PhotonNetwork.leaveRoom, PUN will call this me on the remaining clients. When a remote client drops connection or get closed, this callback gets executed. after a timeout of several seconds.

Implemented in Photon.PunBehaviour.

# void IPunCallbacks.OnPhotonPlayerPropertiesChanged (object[] playe

Called when custom player-properties are changed. Player and the chapassed as object[].

Since v1.25 this method has one parameter: object[] playerAndUpdatec contains two entries.

[0] is the affected **PhotonPlayer**.

[1] is the Hashtable of properties that changed.

We are using a object[] due to limitations of Unity's GameObject.SendN only one optional parameter).

Changing properties must be done by **PhotonPlayer.SetCustomPrope** this callback locally, too.

Example:

```
void OnPhotonPlayerPropertiesChanged(object[] playerAndUpdatedProp
PhotonPlayer player = playerAndUpdatedProps[0] as PhotonPlayer
Hashtable props = playerAndUpdatedProps[1] as Hashtable;
//...
}
```

#### **Parameters**

playerAndUpdatedProps Contains PhotonPlayer and the proper See remarks.

Implemented in Photon.PunBehaviour.

#### void

#### IPunCallbacks.OnPhotonRandomJoinFailed (object[] codeAndMs

Called when a JoinRandom() call failed. The parameter provides **ErrorCode** and message.

Most likely all rooms are full or no rooms are available.

When using multiple lobbies (via JoinLobby or **TypedLobby**), another lobby might have more/fitting rooms.

PUN logs some info if the **PhotonNetwork.logLevel** is >= PhotonLogLevel.Informational.

#### **Parameters**

**codeAndMsg** codeAndMsg[0] is short **ErrorCode**. codeAndMsg[1 is string debug msg.

Implemented in Photon.PunBehaviour.

#### void IPunCallbacks.OnReceivedRoomListUpdate ( )

Called for any update of the room-listing while in a lobby (**PhotonNetwork.insideLobby**) on the Master Server.

PUN provides the list of rooms by **PhotonNetwork.GetRoomList()**. Each item is a **RoomInfo** which might include custom properties

(provided you defined those as lobby-listed when creating a room).

Not all types of lobbies provide a listing of rooms to the client. Some are silent and specialized for server-side matchmaking.

Implemented in Photon.PunBehaviour.

#### void IPunCallbacks.OnUpdatedFriendList()

Called when the server sent the response to a FindFriends request and updated **PhotonNetwork.Friends**.

The friends list is available as **PhotonNetwork.Friends**, listing name, online state and the room a user is in (if any).

Implemented in Photon.PunBehaviour.

# void IPunCallbacks.OnWebRpcResponse (OperationResponse response re

Called by PUN when the response to a WebRPC is available. See PhotonNetwork.WebRPC.

Important: The response.ReturnCode is 0 if **Photon** was able to reach your web-service.

The content of the response is what your web-service sent. You can cre a **WebRpcResponse** from it.

Example: WebRpcResponse webResponse = new WebRpcResponse(operationResponse);

Please note: Class OperationResponse is in a namespace which needs be "used":

using **ExitGames.Client.Photon**; // includes OperationResponse (and other classes)

The OperationResponse.ReturnCode by **Photon** is:

0 for "OK"

- -3 for "Web-Service not configured" (see Dashboard / WebHooks)
- -5 for "Web-Service does now have RPC path/name" (at least for Azı

Implemented in **Photon.PunBehaviour**.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



**Public API** 

# Photon Unity Networking

v1.81

Main Page	e Related	Related Pages		lles	Classes	Files					
Class List	Class Index	Class Hi	erarchy	Class Members							
Photon PunBehaviour											
Photon.PunBehaviour Class Reference											

This class provides a .photonView and all callbacks/events that PUN

can call. Override the events/methods you want to use. More...

Inherits Photon. MonoBehaviour, and IPunCallbacks.

#### **Public Member Functions**

#### virtual void OnConnectedToPhoton ()

Called when the initial connection got established but before you can use the server. **OnJoinedLobby()** or **OnConnectedToMaster()** are called when PUN is ready. More...

#### virtual void OnLeftRoom ()

Called when the local user/client left a room. More...

## virtual void OnMasterClientSwitched (PhotonPlayer newMasterClient)

Called after switching to a new MasterClient when the current one leaves. More...

#### virtual void **OnPhotonCreateRoomFailed** (object[] codeAndMsg)

Called when a **CreateRoom()** call failed. The parameter provides **ErrorCode** and message (as array). <u>More...</u>

#### virtual void OnPhotonJoinRoomFailed (object[] codeAndMsg)

Called when a **JoinRoom()** call failed. The parameter provides **ErrorCode** and message (as array). <u>More...</u>

#### virtual void OnCreatedRoom ()

Called when this client created a room and entered it. **OnJoinedRoom()** will be called as well. <u>More...</u>

### virtual void OnJoinedLobby ()

Called on entering a lobby on the Master Server. The actual room-list updates will call **OnReceivedRoomListUpdate()**. More...

#### virtual void OnLeftLobby ()

Called after leaving a lobby. More...

#### virtual void

#### OnFailedToConnectToPhoton (DisconnectCause cause)

Called if a connect call to the **Photon** server failed before the connection was established, followed by a call to OnDisconnectedFromPhoton(). More...

#### virtual void OnDisconnectedFromPhoton ()

Called after disconnecting from the **Photon** server. More...

#### virtual void OnConnectionFail (DisconnectCause cause)

Called when something causes the connection to fail (after it was established), followed by a call to OnDisconnectedFromPhoton(). More...

#### virtual void **OnPhotonInstantiate** (**PhotonMessageInfo** info)

Called on all scripts on a GameObject (and children) that have been Instantiated using

PhotonNetwork.Instantiate. More...

#### virtual void OnReceivedRoomListUpdate ()

Called for any update of the room-listing while in a lobby (PhotonNetwork.insideLobby) on the Master Server. More...

#### virtual void OnJoinedRoom ()

Called when entering a room (by creating or joining it). Called on all clients (including the Master Client). More...

### virtual void **OnPhotonPlayerConnected** (**PhotonPlayer** newPlayer)

Called when a remote player entered the room. This **PhotonPlayer** is already added to the playerlist at this time. More...

#### OnPhotonPlayerDisconnected (PhotonPlayer virtual void otherPlayer)

Called when a remote player left the room. This

**PhotonPlayer** is already removed from the playerlist at this time. More...

### virtual void **OnPhotonRandomJoinFailed** (object[] codeAndMsg)

Called when a JoinRandom() call failed. The parameter provides **ErrorCode** and message. <u>More...</u>

#### virtual void OnConnectedToMaster ()

Called after the connection to the master is established and authenticated but only when **PhotonNetwork.autoJoinLobby** is false. More...

#### virtual void OnPhotonMaxCccuReached ()

Because the concurrent user limit was (temporarily) reached, this client is rejected by the server and disconnecting. More...

### virtual void OnPhotonCustomRoomPropertiesChanged (Hashtable propertiesThatChanged)

Called when a room's custom properties changed. The propertiesThatChanged contains all that was set via **Room.SetCustomProperties**. More...

## virtual void OnPhotonPlayerPropertiesChanged (object[] playerAndUpdatedProps)

Called when custom player-properties are changed. Player and the changed properties are passed as object[]. More...

#### virtual void OnUpdatedFriendList ()

Called when the server sent the response to a FindFriends request and updated **PhotonNetwork.Friends**. More...

### virtual void **OnCustomAuthenticationFailed** (string debugMessage)

Called when the custom authentication failed. Followed by disconnect! More...

#### virtual void

OnCustomAuthenticationResponse (Dictionary< string, object > data)

Called when your Custom Authentication service responds with additional data. More...

virtual void **OnWebRpcResponse** (OperationResponse response)

Called by PUN when the response to a WebRPC is available. See PhotonNetwork.WebRPC. More...

virtual void **OnOwnershipRequest** (object[] viewAndPlayer)

Called when another player requests ownership of a PhotonView from you (the current owner). More...

virtual void OnLobbyStatisticsUpdate ()

Called when the Master Server sent an update for the Lobby Statistics, updating

PhotonNetwork.LobbyStatistics. More...

#### virtual void

OnPhotonPlayerActivityChanged (PhotonPlayer otherPlayer)

Called when a remote **Photon** Player activity changed. This will be called ONLY if PlayerTtl is greater than 0. More...

virtual void **OnOwnershipTransfered** (object[] viewAndPlayers)

Called when ownership of a **PhotonView** is transferred to another player. More...

### **Additional Inherited Members**

### ▶ Properties inherited from Photon.MonoBehaviour

### PhotonView photonView [get]

A cached reference to a **PhotonView** on this GameObject. <u>More...</u>

### new PhotonView networkView [get]

This property is only here to notify developers when they use the outdated value. More...

### **Detailed Description**

This class provides a .photonView and all callbacks/events that PUN can call. Override the events/methods you want to use.

By extending this class, you can implement individual methods as override.

Visual Studio and MonoDevelop should provide the list of methods when you begin typing "override". Your implementation does not have to call "base.method()".

This class implements **IPunCallbacks**, which is used as definition of all PUN callbacks. Don't implement **IPunCallbacks** in your classes. Instead, implent **PunBehaviour** or individual methods.

### **Member Function Documentation**

#### virtual void Photon.PunBehaviour.OnConnectedToMaster





Called after the connection to the master is established and authenticated but only when **PhotonNetwork.autoJoinLobby** is false.

If you set PhotonNetwork.autoJoinLobby to true, OnJoinedLobby() will be called instead of this.

You can join rooms and create them even without being in a lobby. The default lobby is used in that case. The list of available rooms won't become available unless you join a lobby via PhotonNetwork.joinLobby.

Implements IPunCallbacks.

#### virtual void Photon.PunBehaviour.OnConnectedToPhoton





Called when the initial connection got established but before you can use the server. OnJoinedLobby() or OnConnectedToMaster() are called when PUN is ready.

This callback is only useful to detect if the server can be reached at all (technically). Most often, it's enough to implement OnFailedToConnectToPhoton() and OnDisconnectedFromPhoton().

OnJoinedLobby() or OnConnectedToMaster() are called when PUN is ready.

When this is called, the low level connection is established and PUN

will send your Appld, the user, etc in the background. This is not called for transitions from the masterserver to game servers.

Implements IPunCallbacks.

# virtual void Photon.PunBehaviour.OnConnectionFail ( DisconnectCause cause

Called when something causes the connection to fail (after it was estab followed by a call to **OnDisconnectedFromPhoton()**.

If the server could not be reached in the first place, OnFailedToConnectToPhoton is called instead. The reason for the error provided as DisconnectCause.

Implements IPunCallbacks.

#### virtual void Photon.PunBehaviour.OnCreatedRoom ( )

virtual

Called when this client created a room and entered it. **OnJoinedRoom()** will be called as well.

This callback is only called on the client which created a room (see **PhotonNetwork.CreateRoom**).

As any client might close (or drop connection) anytime, there is a chance that the creator of a room does not execute OnCreatedRoom.

If you need specific room properties or a "start signal", it is safer to implement **OnMasterClientSwitched()** and to make the new MasterClient check the room's state.

Implements IPunCallbacks.

#### virtual void

Photon.PunBehaviour.OnCustomAuthenticationFailed (string deb

Called when the custom authentication failed. Followed by disconnect!

Custom Authentication can fail due to user-input, bad tokens/secrets. If successful, this method is not called. Implement OnJoinedLobby() or OnConnectedToMaster() (as usual).

During development of a game, it might also fail due to wrong configura side. In those cases, logging the debugMessage is very important.

Unless you setup a custom authentication service for your app (in the  $\underline{\Gamma}$ won't be called!

#### **Parameters**

debugMessage Contains a debug message why authentication fa fixed during development time.

Implements IPunCallbacks.

#### virtual void Photon.PunBehaviour.OnCustomAuthenticationResponse (Diction

Called when your Custom Authentication service responds with addition

Custom Authentication services can include some custom data in their is made available in this callback as Dictionary. While the keys of your ( can be either string or a number (in Json). You need to make extra sure expect. Numbers become (currently) int64.

Example: void OnCustomAuthenticationResponse(Dictionary<string, ol https://doc.photonengine.com/en/realtime/current/reference/custom-aut Implements IPunCallbacks.

virtual void Photon.PunBehaviour.OnDisconnectedFromPhoton





Called after disconnecting from the **Photon** server.

In some cases, other callbacks are called before OnDisconnectedFromPhoton is called. Examples: OnConnectionFail() and OnFailedToConnectToPhoton().

Implements IPunCallbacks.

## virtual void Photon.PunBehaviour.OnFailedToConnectToPhoton ( DisconnectC

Called if a connect call to the **Photon** server failed before the connectic followed by a call to **OnDisconnectedFromPhoton()**.

This is called when no connection could be established at all. It differs f OnConnectionFail, which is called when an existing connection fails.

Implements IPunCallbacks.

### virtual void Photon.PunBehaviour.OnJoinedLobby ()

virtual

Called on entering a lobby on the Master Server. The actual roomlist updates will call **OnReceivedRoomListUpdate()**.

Note: When **PhotonNetwork.autoJoinLobby** is false, **OnConnectedToMaster()** will be called and the room list won't become available.

While in the lobby, the roomlist is automatically updated in fixed intervals (which you can't modify). The room list gets available when **OnReceivedRoomListUpdate()** gets called after **OnJoinedLobby()**.

Implements IPunCallbacks.

virtual void Photon.PunBehaviour.OnJoinedRoom ( )

virtual

Called when entering a room (by creating or joining it). Called on all clients (including the Master Client).

This method is commonly used to instantiate player characters. If a match has to be started "actively", you can call an **PunRPC** triggered by a user's button-press or a timer.

When this is called, you can usually already access the existing players in the room via **PhotonNetwork.playerList**. Also, all custom properties should be already available as **Room.customProperties**. Check **Room.playerCount** to find out if enough players are in the room to start playing.

Implements IPunCallbacks.

#### virtual void Photon.PunBehaviour.OnLeftLobby ( )

virtual

Called after leaving a lobby.

When you leave a lobby, **CreateRoom** and **JoinRandomRoom** automatically refer to the default lobby.

Implements IPunCallbacks.

#### virtual void Photon.PunBehaviour.OnLeftRoom ( )

virtual

Called when the local user/client left a room.

When leaving a room, PUN brings you back to the Master Server. Before you can use lobbies and join or create rooms, OnJoinedLobby() or OnConnectedToMaster() will get called again.

Implements IPunCallbacks.

#### virtual void

#### Photon.PunBehaviour.OnLobbyStatisticsUpdate



( ) virtual

Called when the Master Server sent an update for the Lobby Statistics, updating **PhotonNetwork.LobbyStatistics**.

This callback has two preconditions: EnableLobbyStatistics must be set to true, before this client connects. And the client has to be connected to the Master Server, which is providing the info about lobbies.

Implements IPunCallbacks.

#### virtual void

Photon.PunBehaviour.OnMasterClientSwitched (PhotonPlayer ne

Called after switching to a new MasterClient when the current one leave

This is not called when this client enters a room. The former MasterClie list when this method get called.

Implements IPunCallbacks.

#### virtual void

Photon.PunBehaviour.OnOwnershipRequest (object[] viewAndPla

Called when another player requests ownership of a **PhotonView** from current owner).

The parameter viewAndPlayer contains:

**PhotonView** view = viewAndPlayer[0] as **PhotonView**;

**PhotonPlayer** requestingPlayer = viewAndPlayer[1] as **PhotonPlayer**;

#### **Parameters**

viewAndPlayer The PhotonView is viewAndPlayer[0] and the rec player is viewAndPlayer[1].

Implements IPunCallbacks.

# virtual void Photon.PunBehaviour.OnOwnershipTransfered ( object[] viewAnd

Called when ownership of a **PhotonView** is transferred to another playe

The parameter viewAndPlayers contains:

PhotonView view = viewAndPlayers[0] as PhotonView;

PhotonPlayer newOwner = viewAndPlayers[1] as PhotonPlayer;

PhotonPlayer oldOwner = viewAndPlayers[2] as PhotonPlayer;

void OnOwnershipTransfered(object[] viewAndPlayers) {} //

Implements IPunCallbacks.

# virtual void Photon.PunBehaviour.OnPhotonCreateRoomFailed (object[] code

Called when a **CreateRoom()** call failed. The parameter provides **Error** message (as array).

Most likely because the room name is already in use (some other client you). PUN logs some info if the **PhotonNetwork.logLevel** is >= PhotonLogLevel.Informational.

#### **Parameters**

**codeAndMsg** codeAndMsg[0] is a short **ErrorCode** and codeAnd string debug msg.

Implements IPunCallbacks.

#### virtual void

#### Photon.PunBehaviour.OnPhotonCustomRoomPropertiesChanged

Called when a room's custom properties changed. The propertiesThatC Room.SetCustomProperties.

Since v1.25 this method has one parameter: Hashtable propertiesThat( Changing properties must be done by Room.SetCustomProperties, w

#### **Parameters**

propertiesThatChanged

Implements IPunCallbacks.

## virtual void Photon.PunBehaviour.OnPhotonInstantiate ( PhotonMessageInfo

Called on all scripts on a GameObject (and children) that have been Ingusing **PhotonNetwork.Instantiate**.

**PhotonMessageInfo** parameter provides info about who created the ol when (based off PhotonNetworking.time).

Implements IPunCallbacks.

#### virtual void

Photon.PunBehaviour.OnPhotonJoinRoomFailed (object[] codeA

Called when a **JoinRoom()** call failed. The parameter provides **ErrorC** message (as array).

Most likely error is that the room does not exist or the room is full (some was faster than you). PUN logs some info if the **PhotonNetwork.logLe** PhotonLogLevel.Informational.

#### **Parameters**

**codeAndMsg** codeAndMsg[0] is short **ErrorCode**. codeAndMsg[1 debug msg.

#### virtual void Photon.PunBehaviour.OnPhotonMaxCccuReached



Because the concurrent user limit was (temporarily) reached, this client is rejected by the server and disconnecting.

When this happens, the user might try again later. You can't create or join rooms in OnPhotonMaxCcuReached(), cause the client will be disconnecting. You can raise the CCU limits with a new license (when you host yourself) or extended subscription (when using the Photon Cloud). The Photon Cloud will mail you when the CCU limit was reached. This is also visible in the Dashboard (webpage).

Implements IPunCallbacks.

#### virtual void Photon.PunBehaviour.OnPhotonPlayerActivityChanged (PhotonPlayerActivityChanged)

Called when a remote **Photon** Player activity changed. This will be called greater than 0.

Use 0. If true, the player is not gett...">PhotonPlayer.IsInactive to cl activity state.

Example: void OnPhotonPlayerActivityChanged(PhotonPlayer othe

This callback has precondition: PlayerTtl must be greater than 0.

Implements IPunCallbacks.

### virtual void Photon.PunBehaviour.OnPhotonPlayerConnected (PhotonPlayer

Called when a remote player entered the room. This **PhotonPlayer** is a

the playerlist at this time.

If your game starts with a certain number of players, this callback can be the **Room.playerCount** and find out if you can start.

Implements IPunCallbacks.

# virtual void Photon.PunBehaviour.OnPhotonPlayerDisconnected ( PhotonPlayerDisconnected ( PhotonPlayerD

Called when a remote player left the room. This **PhotonPlayer** is alread playerlist at this time.

When your client calls PhotonNetwork.leaveRoom, PUN will call this medients. When a remote client drops connection or gets closed, this call after a timeout of several seconds.

Implements IPunCallbacks.

### virtual void Photon.PunBehaviour.OnPhotonPlayerPropertiesChanged ( object

Called when custom player-properties are changed. Player and the chaobject[].

Since v1.25 this method has one parameter: object[] playerAndUpdatec [0] is the affected **PhotonPlayer**.

[1] is the Hashtable of properties that changed.

We are using a object[] due to limitations of Unity's GameObject.SendN optional parameter).

Changing properties must be done by **PhotonPlayer.SetCustomPrope** locally, too.

#### Example:

 $\verb"void OnPhotonPlayerPropertiesChanged(object[] playerAndUpdatedPropertiesChanged(object[] playerAndUpdatedPr$ 

```
PhotonPlayer player = playerAndUpdatedProps[0] as PhotonPlayer
    Hashtable props = playerAndUpdatedProps[1] as Hashtable;
    //...
}
```

#### **Parameters**

playerAndUpdatedProps Contains PhotonPlayer and the proper Implements IPunCallbacks.

#### virtual void Photon.PunBehaviour.OnPhotonRandomJoinFailed (object[] code

Called when a JoinRandom() call failed. The parameter provides Error message.

Most likely all rooms are full or no rooms are available.

When using multiple lobbies (via JoinLobby or TypedLobby), another le more/fitting rooms.

PUN logs some info if the **PhotonNetwork.logLevel** is >= PhotonLogLevel.Informational.

#### **Parameters**

codeAndMsg codeAndMsg[0] is short ErrorCode. codeAndMsg[1 msq.

Implements IPunCallbacks.

### virtual void Photon.PunBehaviour.OnReceivedRoomListUpdate





Called for any update of the room-listing while in a lobby (PhotonNetwork.insideLobby) on the Master Server.

PUN provides the list of rooms by PhotonNetwork.GetRoomList(). Each item is a **RoomInfo** which might include custom properties (provided you defined those as lobby-listed when creating a room).

Not all types of lobbies provide a listing of rooms to the client. Some are silent and specialized for server-side matchmaking.

Implements IPunCallbacks.

#### virtual void Photon.PunBehaviour.OnUpdatedFriendList





Called when the server sent the response to a FindFriends request and updated PhotonNetwork.Friends.

The friends list is available as **PhotonNetwork.Friends**, listing name, online state and the room a user is in (if any).

Implements IPunCallbacks.

#### virtual void Photon.PunBehaviour.OnWebRpcResponse (OperationResponse

Called by PUN when the response to a WebRPC is available. See PhotonNetwork.WebRPC.

Important: The response.ReturnCode is 0 if **Photon** was able to reach The content of the response is what your web-service sent. You can cre WebResponse instance from it. Example: WebRpcResponse webRes WebRpcResponse(operationResponse);

Please note: Class OperationResponse is in a namespace which needs using ExitGames.Client.Photon; // includes OperationResponse (and

The OperationResponse.ReturnCode by **Photon** is:

- 0 for "OK"
- -3 for "Web-Service not configured" (see Dashboard / WebHooks)
- -5 for "Web-Service does now have RPC path/name" (at least for Azı

Implements IPunCallbacks.



# **Photon Unity Networking**

v1.81

Main Page		Related Pages		Modules		Classes	Files
Class List	Class Index		Class Hierarchy		Clas	ss Members	

### PhotonMessageInfo Struct Reference

**Public API** 

<u>Public Member Functions</u> | <u>Public Attributes</u> | <u>Properties</u> | <u>List of all members</u>

Container class for info about a particular message, RPC or update. More...

### **Public Member Functions**

PhotonMessageInfo (PhotonPlayer player, int timestamp, PhotonView view)

override string ToString ()

### **Public Attributes**

readonly PhotonPlayer sender

readonly PhotonView photonView

### Properties

double timestamp [get]

### **Detailed Description**

Container class for info about a particular message, RPC or update.

# Constructor & Destructor Documentation

```
PhotonMessageInfo.PhotonMessageInfo ( PhotonPlayer player, int timestam PhotonView view )
```

# **Member Function Documentation**

override string PhotonMessageInfo.ToString ( )

# Member Data Documentation

readonly PhotonView PhotonMessageInfo.photonView

readonly PhotonPlayer PhotonMessageInfo.sender

# **Property Documentation**

## double PhotonMessageInfo.timestamp



Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# **Photon Unity Networking**

v1.81

Main Page Related		Pages	Modu	les	Classes	Files	
Class List	Class Index		Class Hi	erarchy	Clas	ss Members	

# PhotonStream Class Reference

**Public API** 

<u>Public Member Functions | Properties |</u>
<u>List of all members</u>

This container is used in **OnPhotonSerializeView()** to either provide incoming data of a **PhotonView** or for you to provide it. <u>More...</u>

### **Public Member Functions**

#### PhotonStream (bool write, object[] incomingData)

Creates a stream and initializes it. Used by PUN internally. More...

#### void SetReadStream (object[] incomingData, byte pos=0)

#### object ReceiveNext ()

Read next piece of data from the stream when is Reading is true. More...

#### object PeekNext ()

Read next piece of data from the stream without advancing the "current" item. More...

#### void SendNext (object obj)

Add another piece of data to send it when is Writing is true. More...

### object[] ToArray ()

Turns the stream into a new object[]. More...

### void **Serialize** (ref bool myBool)

Will read or write the value, depending on the stream's isWriting value. More...

#### void Serialize (ref int myInt)

Will read or write the value, depending on the stream's isWriting value. More...

### void **Serialize** (ref string value)

Will read or write the value, depending on the stream's isWriting value. More...

#### void **Serialize** (ref char value)

Will read or write the value, depending on the stream's isWriting value. More...

#### void **Serialize** (ref short value)

Will read or write the value, depending on the stream's isWriting value. More...

#### void Serialize (ref float obj)

Will read or write the value, depending on the stream's isWriting value. More...

#### void Serialize (ref PhotonPlayer obj)

Will read or write the value, depending on the stream's isWriting value. More...

#### void **Serialize** (ref Vector3 obj)

Will read or write the value, depending on the stream's isWriting value. More...

#### void Serialize (ref Vector2 obj)

Will read or write the value, depending on the stream's isWriting value. More...

#### void Serialize (ref Quaternion obj)

Will read or write the value, depending on the stream's isWriting value. More...

# **Properties**

### bool isWriting [get]

If true, this client should add data to the stream to send it. More...

### bool isReading [get]

If true, this client should read data send by another client. More...

### int Count [get]

Count of items in the stream. More...

# **Detailed Description**

This container is used in **OnPhotonSerializeView()** to either provide incoming data of a **PhotonView** or for you to provide it.

The isWriting property will be true if this client is the "owner" of the **PhotonView** (and thus the GameObject). Add data to the stream and it's sent via the server to the other players in a room. On the receiving side, isWriting is false and the data should be read.

Send as few data as possible to keep connection quality up. An empty **PhotonStream** will not be sent.

Use either **Serialize()** for reading and writing or **SendNext()** and **ReceiveNext()**. The latter two are just explicit read and write methods but do about the same work as **Serialize()**. It's a matter of preference which methods you use.

See also

PhotonNetworkingMessage

# Constructor & Destructor Documentation

```
PhotonStream (bool write, object[] incomingData )
```

Creates a stream and initializes it. Used by PUN internally.

### **Member Function Documentation**

#### object PhotonStream.PeekNext ( )

Read next piece of data from the stream without advancing the "current" item.

#### object PhotonStream.ReceiveNext ()

Read next piece of data from the stream when isReading is true.

### void PhotonStream.SendNext ( object obj )

Add another piece of data to send it when is Writing is true.

### void PhotonStream.Serialize (ref bool myBool)

Will read or write the value, depending on the stream's isWriting value.

#### void PhotonStream.Serialize ( ref int myInt )

Will read or write the value, depending on the stream's isWriting value.

### void PhotonStream.Serialize (ref string value)

Will read or write the value, depending on the stream's isWriting

value.

#### void PhotonStream.Serialize (ref char value)

Will read or write the value, depending on the stream's isWriting value.

#### void PhotonStream.Serialize (ref short value)

Will read or write the value, depending on the stream's isWriting value.

#### void PhotonStream.Serialize (ref float obj)

Will read or write the value, depending on the stream's isWriting value.

### void PhotonStream.Serialize (ref PhotonPlayer obj)

Will read or write the value, depending on the stream's isWriting value.

#### void PhotonStream.Serialize (ref Vector3 obj)

Will read or write the value, depending on the stream's isWriting value.

### void PhotonStream.Serialize (ref Vector2 obj)

Will read or write the value, depending on the stream's isWriting value.

# void PhotonStream.Serialize (ref Quaternion obj)

Will read or write the value, depending on the stream's isWriting value.

# object [] PhotonStream.ToArray ()

Turns the stream into a new object[].

# **Property Documentation**

#### int PhotonStream.Count

get

Count of items in the stream.

## bool PhotonStream.isReading

get

If true, this client should read data send by another client.

## bool PhotonStream.isWriting

get

If true, this client should add data to the stream to send it.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page	Related	Pages Modu		lles	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	
DhatanNaturark Class				Public Member Functions   Static Public Member Functions		
PhotonNetwork Class						•
Reference			<u>Pub</u>	Public Attributes   Static Public Attributes		
					Properties   List of	of all members
Public API						

The main class to use the  ${\bf PhotonNetwork}$  plugin. This class is static.  ${\bf More...}$ 

# **Public Member Functions**

**EventCallback** (byte eventCode, object content, int senderId)
Defines the delegate usable in OnEventCall. More... delegate void

Static Public Member Functions				
static void	SwitchToProtocol (ConnectionProtocol cp) While offline, the network protocol can be switched (which affects the ports you can use to connect). More			
static bool	ConnectUsingSettings (string gameVersion) Connect to Photon as configured in the editor (saved in PhotonServerSettings file). More			
static bool	ConnectToMaster (string masterServerAddress, int port, string appID, string gameVersion) Connect to a Photon Master Server by address, port, appID and game(client) version. More			
static bool	Reconnect () Can be used to reconnect to the master server after a disconnect. More			
static bool	ReconnectAndRejoin () When the client lost connection during gameplay, this method attempts to reconnect and rejoin the room. More			
static bool	ConnectToBestCloudServer (string gameVersion) Connect to the Photon Cloud region with the lowest ping (on platforms			

that support	Unity's	Ping).	More
--------------	---------	--------	------

### ConnectToRegion static bool (CloudRegionCode region, string gameVersion)

Connects to the **Photon** Cloud region of choice. More...

### static void

### OverrideBestCloudServer (CloudRegionCode region)

Overwrites the region that is used for ConnectToBestCloudServer(string gameVersion). More...

#### static void RefreshCloudServerRating ()

Pings all cloud servers again to find the one with best ping (currently). More...

### static void NetworkStatisticsReset ()

Resets the traffic stats and reenables them. More...

### static string NetworkStatisticsToString ()

Only available when NetworkStatisticsEnabled was used to gather some stats. More...

### static void InitializeSecurity ()

Used for compatibility with Unity networking only. Encryption is automatically initialized while connecting. More...

#### static void **Disconnect** ()

Makes this client disconnect from the photon server, a process that leaves any room and calls

	OnDisconnectedFromPhoton on completion. More
static bool	FindFriends (string[] friendsToFind) Requests the rooms and online status for a list of friends and saves the result in PhotonNetwork.Friends. More
static bool	CreateRoom (string roomName) Creates a room with given name but fails if this room(name) is existing already. Creates random name for roomName null. More
static bool	CreateRoom (string roomName, RoomOptions roomOptions, TypedLobby typedLobby) Creates a room but fails if this room is existing already. Can only be called on Master Server. More
static bool	CreateRoom (string roomName, RoomOptions roomOptions, TypedLobby typedLobby, string[] expectedUsers) Creates a room but fails if this room is existing already. Can only be called on Master Server. More
static bool	JoinRoom (string roomName) Join room by roomname and on success calls OnJoinedRoom(). This is not affected by lobbies. More
static bool	JoinRoom (string roomName, string[] expectedUsers)

Join room by roomname and on success calls OnJoinedRoom(). This is not affected by lobbies. More...

## static bool

JoinOrCreateRoom (string roomName, RoomOptions roomOptions, TypedLobby typedLobby)

Lets you either join a named room or create it on the fly - you don't have to know if someone created the room already. More...

### static bool

JoinOrCreateRoom (string roomName, RoomOptions roomOptions, TypedLobby typedLobby, string[] expectedUsers) Lets you either join a named room or create it on the fly - you don't have to know if someone created the room already. More...

static bool JoinRandomRoom ()

Joins any available room of the currently used lobby and fails if none is available. More...

JoinRandomRoom (Hashtable static bool expectedCustomRoomProperties, byte expectedMaxPlayers)

Attempts to join an open room with fitting, custom properties but fails if none is currently available. More...

JoinRandomRoom (Hashtable expectedCustomRoomProperties, byte expectedMaxPlayers, static bool **MatchmakingMode** matchingType,

### **TypedLobby** typedLobby, string sqlLobbyFilter, string[] expectedUsers=null)

Attempts to join an open room with fitting, custom properties but fails if none is currently available. More...

### static bool **ReJoinRoom** (string roomName)

Can be used to return to a room after a disconnect and reconnect. More...

#### static bool **JoinLobby** ()

On MasterServer this joins the default lobby which list rooms currently in use. More...

#### static bool

#### JoinLobby (TypedLobby typedLobby)

On a Master Server you can join a lobby to get lists of available rooms. More...

### static bool LeaveLobby ()

Leave a lobby to stop getting updates about available rooms. More...

### static bool LeaveRoom ()

Leave the current room and return to the Master Server where you can join or create rooms (see remarks). More...

#### static bool

### **GetCustomRoomList (TypedLobby** typedLobby, string sqlLobbyFilter)

Fetches a custom list of games from the server, matching a SQL-like

"where" clause, then triggers OnReceivedRoomListUpdate callback. More...

#### static **RoomInfo**[]

#### GetRoomList ()

Gets currently known rooms as RoomInfo array. This is available and updated while in a lobby (check insideLobby). More...

#### static void

### **SetPlayerCustomProperties** (Hashtable customProperties)

Sets this (local) player's properties and synchronizes them to the other players (don't modify them directly). More...

#### static void

## RemovePlayerCustomProperties (string[] customPropertiesToDelete)

Locally removes Custom Properties of "this" player. Important: This does not synchronize the change! Useful when you switch rooms. More...

### RaiseEvent (byte eventCode, object static bool eventContent, bool sendReliable, RaiseEventOptions options)

Sends fully customizable events in a room. Events consist of at least an EventCode (0..199) and can have content. More...

#### static int AllocateViewID ()

Allocates a viewID that's valid for the current/local player. More...

### static int AllocateSceneViewID ()

Enables the Master Client to allocate

a viewID that is valid for scene objects. More...

#### static void **UnAllocateViewID** (int viewID) Unregister a viewID (of manually instantiated and destroyed

networked objects). More...

### **Instantiate** (string prefabName, static GameObject Vector3 position, Quaternion rotation, int group)

Instantiate a prefab over the network. This prefab needs to be located in the root of a "Resources" folder. More...

**Instantiate** (string prefabName, static GameObject Vector3 position, Quaternion rotation, int group, object∏ data) Instantiate a prefab over the

network. This prefab needs to be located in the root of a "Resources" folder. More...

### static GameObject

InstantiateSceneObject (string prefabName, Vector3 position, Quaternion rotation, int group, object∏ data)

Instantiate a scene-owned prefab over the network. The PhotonViews will be controllable by the MasterClient. This prefab needs to be located in the root of a "Resources" folder. More...

### static int GetPing ()

The current roundtrip time to the photon server. More...

#### static void FetchServerTimestamp ()

Refreshes the server timestamp (async operation, takes a roundtrip). More...

#### static void SendOutgoingCommands ()

Can be used to immediately send the RPCs and Instantiates just called, so they are on their way to the other players. More...

#### **CloseConnection (PhotonPlayer** static bool kickPlayer)

Request a client to disconnect (KICK). Only the master client can do this More...

#### static bool

#### SetMasterClient (PhotonPlayer masterClientPlayer)

Asks the server to assign another player as Master Client of your current room. More...

### static void **Destroy** (**PhotonView** targetView)

Network-Destroy the GameObject associated with the **PhotonView**. unless the **PhotonView** is static or not under this client's control. More...

### static void **Destroy** (GameObject targetGo)

Network-Destroy the GameObject, unless it is static or not under this client's control. More...

### static void

## **DestroyPlayerObjects** (PhotonPlayer targetPlayer)

Network-Destroy all GameObjects,

PhotonViews and their RPCs of targetPlayer. Can only be called on local player (for "self") or Master Client (for anyone). More...

### static void

#### **DestroyPlayerObjects** (int targetPlayerId)

Network-Destroy all GameObjects, PhotonViews and their RPCs of this player (by ID). Can only be called on local player (for "self") or Master Client (for anyone). More...

#### static void **DestroyAll** ()

Network-Destroy all GameObjects. PhotonViews and their RPCs in the room. Removes anything buffered from the server. Can only be called by Master Client (for anyone). More...

### static void

#### RemoveRPCs (PhotonPlayer targetPlayer)

Remove all buffered RPCs from server that were sent by targetPlayer. Can only be called on local player (for "self") or Master Client (for anyone). More...

#### static void

### RemoveRPCs (PhotonView targetPhotonView)

Remove all buffered RPCs from server that were sent via targetPhotonView. The Master Client and the owner of the targetPhotonView may call this. More...

### RemoveRPCsInGroup (int

#### static void targetGroup)

Remove all buffered RPCs from server that were sent in the targetGroup, if this is the Master Client or if this controls the individual **PhotonView**. More...

#### static void

# CacheSendMonoMessageTargets (Type type)

Populates

SendMonoMessageTargets with currently existing GameObjects that have a Component of type. More...

# static HashSet< GameObject >

# FindGameObjectsWithComponent (Type type)

Finds the GameObjects with Components of a specific type (using FindObjectsOfType). More...

### static void

# **SetReceivingEnabled** (int group, bool enabled)

Enable/disable receiving on given group (applied to PhotonViews)
More...

### static void

# SetReceivingEnabled (int[] enableGroups, int[] disableGroups)

Enable/disable receiving on given groups (applied to PhotonViews) More...

### static void

# **SetSendingEnabled** (int group, bool enabled)

Enable/disable sending on given group (applied to PhotonViews) More...

#### static void

SetSendingEnabled (int[] enableGroups, int[] disableGroups) Enable/disable sending on given groups (applied to PhotonViews) More...

static void **SetLevelPrefix** (short prefix) Sets level prefix for PhotonViews instantiated later on. Don't set it if

you need only one! More...

#### static void **LoadLevel** (int levelNumber)

Wraps loading a level to pause the network mesage-queue. Optionally syncs the loaded level in a room. More...

#### static void **LoadLevel** (string levelName)

Wraps loading a level to pause the network mesage-queue. Optionally syncs the loaded level in a room. More...

### static bool

### WebRpc (string name, object parameters)

This operation makes **Photon** call your custom web-service by name (path) with the given parameters. More...

# **Public Attributes**

const string **versionPUN** = "1.81"

Version number of PUN. Also used in GameVersion to separate client version from each other. <u>More...</u>

# Static Public Attributes

static readonly int	MAX_VIEW_IDS = 1000 The maximum number of assigned Phose the General Documentation topic limitation. More
	minication. <u>More</u>
static <b>ServerSettings</b>	PhotonServerSettings = (ServerSettings)Resources.Load(Photypeof(ServerSettings)) Serialized server settings, written by th ConnectUsingSettings. More
static hool	InstantiatoInBoomOnly - true
Static bool	InstantiateInRoomOnly = true If true, Instantiate methods will check if are not. More
static <b>PhotonLogLevel</b>	Network log level. Controls how verbos
static float	precisionForVectorSynchronization The minimum difference that a Vector2 rotation) needs to change before we se OnSerialize/ObservingComponent. Mo
static float	The minimum angle that a rotation nee a <b>PhotonView</b> 's OnSerialize/Observing
static float	precisionForFloatSynchronization = The minimum difference between floats PhotonView's OnSerialize/ObservingC
static bool	UseRpcMonoBehaviourCache
	While enabled, the MonoBehaviours or avoiding costly GetComponents <mono< th=""></mono<>

static bool	UsePrefabCache = true While enabled (true), Instantiate uses I keep game objects in memory (improvi prefab). More
static Dictionary< string, GameObject >	PrefabCache = new Dictionary <string, for<="" gameobjects="" keeps="" references="" th="" to=""></string,>
	memory instead of loading the Resource
static HashCat < CamaOhiast >	CondMonoMonograTorgato
static HashSet< GameObject >	SendMonoMessageTargets If not null, this is the (exclusive) list of ( PUN SendMonoMessage(). More
static Type	SendMonoMessageTargetType = typ Defines which classes can contain PUI More
static bool	StartRpcsAsCoroutine = true Can be used to skip starting RPCs as of performance issue. More
static int	maxConnections Only used in Unity Networking. In PUN PhotonNetwork.CreateRoom. More
static float	BackgroundTimeout = 60.0f Defines how many seconds PUN keep OnApplicationPause(true) call. Default:
static <b>EventCallback</b>	OnEventCall Register your RaiseEvent handling me

## **Properties**

#### static string gameVersion [get, set]

Version string for your this build. Can be used to separate incompatible clients. Sent during connect. More...

#### static string ServerAddress [get]

Currently used server address (no matter if master or game server). More...

#### static bool connected [get]

False until you connected to **Photon** initially. True in offline mode, while connected to any server and even while switching servers. <u>More...</u>

#### static bool connecting [get]

True when you called ConnectUsingSettings (or similar) until the low level connection to **Photon** gets established. More...

### static bool connectedAndReady [get]

A refined version of connected which is true only if your connection to the server is ready to accept operations like join, leave, etc. <u>More...</u>

### static ConnectionState connectionState [get]

Simplified connection state More...

#### static ClientState connectionStateDetailed [get]

Detailed connection state (ignorant

of PUN, so it can be "disconnected" while switching servers). More...

#### static ServerConnection Server [get]

The server (type) this client is currently connected or connecting to. More...

#### static Authentication Values Auth Values [get, set]

A user's authentication values used during connect. More...

#### static Room room [get]

Get the room we're currently in. Null if we aren't in any room. More...

### static **PhotonPlayer**

#### player [get]

The local **PhotonPlayer**. Always available and represents this player. CustomProperties can be set before entering a room and will be synced as well. More...

### static **PhotonPlayer**

#### masterClient [get]

The Master Client of the current room or null (outside of rooms). More...

#### static string

#### playerName [get, set]

Set to synchronize the player's nickname with everyone in the room(s) you enter. This sets PhotonNetwork.player.NickName. More...

### static **PhotonPlayer**[]

#### playerList [get]

The list of players in the current

room, including the local player. More...

#### static **PhotonPlayer**[]

#### otherPlayers [get]

The list of players in the current room, excluding the local player. More...

#### static List< FriendInfo >

#### Friends [get, set]

Read-only list of friends, their online status and the room they are in. Null until initialized by a FindFriends call. More...

#### static int FriendsListAge [get]

Age of friend list info (in milliseconds). It's 0 until a friend list is fetched. More...

#### static IPunPrefabPool PrefabPool [get, set]

An Object Pool can be used to keep and reuse instantiated object instances. It replaced Unity's default Instantiate and Destroy methods. More...

#### static bool offlineMode [get, set]

Offline mode can be set to re-use your multiplayer code in singleplayer game modes. When this is on PhotonNetwork will not create any connections and there is near to no overhead. Mostly usefull for reusing RPC's and

PhotonNetwork.Instantiate More...

### static bool

#### automaticallySyncScene [get,

#### set]

Defines if all clients in a room should load the same level as the Master Client (if that used

PhotonNetwork.LoadLevel). More...

### static bool

### autoCleanUpPlayerObjects [get, set]

This setting defines per room, if network-instantiated GameObjects (with **PhotonView**) get cleaned up when the creator of it leaves. More...

#### static bool autoJoinLobby [get, set]

Set in PhotonServerSettings asset. Defines if the **PhotonNetwork** should join the "lobby" when connected to the Master server. More...

# static bool EnableLobbyStatistics [get, set] Set in PhotonServerSettings asset.

Enable to get a list of active lobbies from the Master Server, More...

### static List< TypedLobbyInfo >

### LobbyStatistics [get, set]

If turned on, the Master Server will provide information about active lobbies for this application. More...

### static bool insideLobby [get]

True while this client is in a lobby. More...

### static **TypedLobby**

lobby [get, set]

The lobby that will be used when

PUN joins a lobby or creates a game. More...

#### static int sendRate [get, set]

Defines how many times per second **PhotonNetwork** should send a package. If you change this, do not forget to also change 'sendRateOnSerialize'. More...

#### static int sendRateOnSerialize [get, set]

Defines how many times per second OnPhotonSerialize should be called on PhotonViews. More...

### static bool

# isMessageQueueRunning [get, set]

Can be used to pause dispatching of incoming evtents (RPCs, Instantiates and anything else incoming). More...

### static int

# unreliableCommandsLimit [get, set]

Used once per dispatch to limit unreliable commands per channel (so after a pause, many channels can still cause a lot of unreliable commands) More...

### static double time [get]

**Photon** network time, synched with the server. More...

### static int **ServerTimestamp** [get]

The current server's millisecond timestamp. More...

#### static bool isMasterClient [get]

Are we the master client? More...

#### static bool inRoom [get]

Is true while being in a room (connectionStateDetailed == ClientState.Joined). More...

### static bool isNonMasterClientInRoom [get]

True if we are in a room (client) and NOT the room's masterclient More...

#### static int countOfPlayersOnMaster [get]

The count of players currently looking for a room (available on MasterServer in 5sec intervals). More...

#### static int countOfPlayersInRooms [get]

Count of users currently playing your app in some room (sent every 5sec by Master Server). Use playerList.Count to get the count of players in the room you're in! More...

#### static int countOfPlayers [get]

The count of players currently using this application (available on MasterServer in 5sec intervals). More...

# static int countOfRooms [get]

The count of rooms currently in use (available on MasterServer in 5sec intervals). More...

#### static bool

#### **NetworkStatisticsEnabled** [get, set1

Enables or disables the collection of statistics about this client's traffic. More...

#### static int ResentReliableCommands [get]

Count of commands that got repeated (due to local repeat-timing before an ACK was received). More...

#### static bool CrcCheckEnabled [get, set]

Crc checks can be useful to detect and avoid issues with broken datagrams. Can be enabled while not connected. More...

#### static int PacketLossByCrcCheck [get]

If CrcCheckEnabled, this counts the incoming packages that don't have a valid CRC checksum and got rejected. More...

#### static int

#### MaxResendsBeforeDisconnect [get, set]

Defines the number of times a reliable message can be resent before not getting an ACK for it will trigger a disconnect. Default: 5. More...

#### static int QuickResends [get, set]

In case of network loss, reliable messages can be repeated quickly up to 3 times. More...

## **Detailed Description**

The main class to use the **PhotonNetwork** plugin. This class is static.

## **Member Function Documentation**

#### static int PhotonNetwork.AllocateSceneViewID ( )

static

Enables the Master Client to allocate a viewID that is valid for scene objects.

#### Returns

A viewID that can be used for a new **PhotonView** or -1 in case of an error.

#### static int PhotonNetwork.AllocateViewID ( )

static

Allocates a viewID that's valid for the current/local player.

#### Returns

A viewID that can be used for a new **PhotonView**.

#### static void

PhotonNetwork.CacheSendMonoMessageTargets (Type type) st

Populates SendMonoMessageTargets with currently existing GameObjects that have a Component of type.

#### **Parameters**

type If null, this will use SendMonoMessageTargets as component type (MonoBehaviour by default).

#### static bool

PhotonNetwork.CloseConnection (PhotonPlayer kickPlayer)

Request a client to disconnect (KICK). Only the master client can do this

Only the target player gets this event. That player will disconnect automatically, which is what the others will notice, too.

#### **Parameters**

kickPlayer The PhotonPlayer to kick.

## static bool PhotonNetwork.ConnectToBestCloudServer ( string gameVersion

Connect to the **Photon** Cloud region with the lowest ping (on platforms support Unity's Ping).

Will save the result of pinging all cloud servers in PlayerPrefs. Calling the first time can take +-2 seconds. The ping result can be overridden via **PhotonNetwork.OverrideBestCloudServer(..)** This call can take up to seconds if it is the first time you are using this, all cloud servers will be a to check for the best region.

The PUN Setup Wizard stores your appID in a settings file and applies server address/port. To connect to the **Photon** Cloud, a valid AppId mu in the settings file (shown in the **Photon** Cloud Dashboard). <a href="https://www.photonengine.com/dashboard">https://www.photonengine.com/dashboard</a>

Connecting to the **Photon** Cloud might fail due to:

- Invalid Appld (calls: **OnFailedToConnectToPhoton()**. check exact value)
- Network issues (calls: OnFailedToConnectToPhoton())
- Invalid region (calls: OnConnectionFail() with DisconnectCause.InvalidRegion)
- Subscription CCU limit reached (calls: OnConnectionFail() with DisconnectCause.MaxCcuReached. also calls: OnPhotonMaxCccuReached())

More about the connection limitations: <a href="http://doc.exitgames.com/en/pur">http://doc.exitgames.com/en/pur</a>

#### **Parameters**

**gameVersion** This client's version number. Users are separated fr each other by gameversion (which allows you to ma breaking changes).

#### Returns

If this client is going to connect to cloud server based on ping. Ever true, this does not guarantee a connection but the attempt is being made.

```
static bool
PhotonNetwork.ConnectToMaster ( string masterServerAddress, int port, string appID, string gameVersion )
```

Connect to a **Photon** Master Server by address, port, appID and game(client) version.

To connect to the **Photon** Cloud, a valid Appld must be in the settings f (shown in the **Photon** Cloud Dashboard). https://www.photonengine.com/dashboard

Connecting to the **Photon** Cloud might fail due to:

- Invalid Appld (calls: OnFailedToConnectToPhoton(). check exact Appld value)
- Network issues (calls: OnFailedToConnectToPhoton())
- Invalid region (calls: **OnConnectionFail()** with DisconnectCause.InvalidRegion)
- Subscription CCU limit reached (calls: OnConnectionFail() with DisconnectCause.MaxCcuReached. also calls: OnPhotonMaxCccuReached())

More about the connection limitations: <a href="http://doc.exitgames.com/en/pur">http://doc.exitgames.com/en/pur</a>

#### **Parameters**

masterServerAddress The server's address (either your own or

Photon Cloud address).

The server's port to connect to. port

Your application ID (Photon Cloud provide applD

you with a GUID for your game).

This client's version number. Users are gameVersion

separated by gameversion (which allows v

to make breaking changes).

#### static bool

PhotonNetwork.ConnectToRegion (CloudRegionCode region, gameVersic string

Connects to the **Photon** Cloud region of choice.

#### static bool

PhotonNetwork.ConnectUsingSettings (string gameVersion)

Connect to **Photon** as configured in the editor (saved in PhotonServerSettings file).

This method will disable offlineMode (which won't destroy any instantiated GOs) and it will set isMessageQueueRunning to true.

Your server configuration is created by the PUN Wizard and contains the Appld and region for **Photon** Cloud games and the server address if you host **Photon** yourself. These settings usually don't change often.

To ignore the config file and connect anywhere call: PhotonNetwork.ConnectToMaster.

To connect to the **Photon** Cloud, a valid Appld must be in the settings file (shown in the **Photon** Cloud Dashboard). https://www.photonengine.com/dashboard

Connecting to the **Photon** Cloud might fail due to:

- Invalid Appld (calls: OnFailedToConnectToPhoton(). check exact Appld value)
- Network issues (calls: OnFailedToConnectToPhoton())
- Invalid region (calls: OnConnectionFail() with DisconnectCause.InvalidRegion)
- Subscription CCU limit reached (calls: OnConnectionFail() with DisconnectCause.MaxCcuReached. also calls: OnPhotonMaxCccuReached())

More about the connection limitations: <a href="http://doc.exitgames.com/en/pur">http://doc.exitgames.com/en/pur</a>

#### **Parameters**

**gameVersion** This client's version number. Users are separated from each other by gameversion (which allows you to make breaking changes).

## static bool PhotonNetwork.CreateRoom

(string roomName)

static

Creates a room with given name but fails if this room(name) is existing already. Creates random name for roomName null.

If you don't want to create a unique room-name, pass null or "" as name and the server will assign a roomName (a GUID as string).

The created room is automatically placed in the currently used lobby (if any) or the default-lobby if you didn't explicitly join one.

Call this only on the master server. Internally, the master will respond with a server-address (and roomName, if needed). Both are used internally to switch to the assigned game server and roomName.

**PhotonNetwork.autoCleanUpPlayerObjects** will become this room's AutoCleanUp property and that's used by all clients that join this room.

#### **Parameters**

**roomName** Unique name of the room to create.

#### Returns

If the operation got queued and will be sent.

```
static bool
PhotonNetwork.CreateRoom ( string roomName,
RoomOptions roomOptions,
TypedLobby typedLobby
)
```

Creates a room but fails if this room is existing already. Can only be called on Master Server.

When successful, this calls the callbacks OnCreatedRoom and OnJoinedRoom (the latter, cause you join as first player). If the room can't be created (because it exists already), OnPhotonCreateRoomFailed gets called.

If you don't want to create a unique room-name, pass null or "" as name and the server will assign a roomName (a GUID as string).

Rooms can be created in any number of lobbies. Those don't have to exist before you create a room in them (they get auto-created on demand). Lobbies can be useful to split room lists on the server-side already. That can help keep the room lists short and manageable. If you set a typedLobby parameter, the room will be created in that lobby (no matter if you are active in any). If you don't set a typedLobby, the room is automatically placed in the currently active lobby (if any) or the default-lobby.

Call this only on the master server. Internally, the master will respond with a server-address (and roomName, if needed). Both are used internally to switch to the assigned game server and roomName.

**PhotonNetwork.autoCleanUpPlayerObjects** will become this room's autoCleanUp property and that's used by all clients that join this room.

#### **Parameters**

roomName Unique name of the room to create. Pass null or

"" to make the server generate a name.

roomOptions Common options for the room like MaxPlayers,

initial custom room properties and similar. See

RoomOptions type...

typedLobby If null, the room is automatically created in the

currently used lobby (which is "default" when

you didn't join one explicitly).

#### **Returns**

If the operation got queued and will be sent.

```
static bool
PhotonNetwork.CreateRoom ( string roomName,
RoomOptions roomOptions,
TypedLobby typedLobby,
string[] expectedUsers
)
```

Creates a room but fails if this room is existing already. Can only be called on Master Server.

When successful, this calls the callbacks OnCreatedRoom and OnJoinedRoom (the latter, cause you join as first player). If the room can't be created (because it exists already), OnPhotonCreateRoomFailed gets called.

If you don't want to create a unique room-name, pass null or "" as name and the server will assign a roomName (a GUID as string).

Rooms can be created in any number of lobbies. Those don't have to exist before you create a room in them (they get auto-created on demand). Lobbies can be useful to split room lists on the server-side already. That can help keep the room lists short and manageable. If you set a typedLobby parameter, the room will be created in that lobby (no matter if you are active in any). If you don't set a typedLobby, the

room is automatically placed in the currently active lobby (if any) or the default-lobby.

Call this only on the master server. Internally, the master will respond with a server-address (and roomName, if needed). Both are used internally to switch to the assigned game server and roomName.

PhotonNetwork.autoCleanUpPlayerObjects will become this room's autoCleanUp property and that's used by all clients that join this room.

You can define an array of expectedUsers, to block player slots in the room for these users. The corresponding feature in **Photon** is called "Slot Reservation" and can be found in the doc pages.

#### **Parameters**

**roomName** Unique name of the room to create. Pass null or

"" to make the server generate a name.

**roomOptions** Common options for the room like MaxPlayers,

initial custom room properties and similar. See

RoomOptions type..

**typedLobby** If null, the room is automatically created in the

currently used lobby (which is "default" when

you didn't join one explicitly).

expectedUsers Optional list of users (by UserId) who are

expected to join this game and who you want to

block a slot for.

#### Returns

If the operation got queued and will be sent.

#### static void

PhotonNetwork.Destroy

(PhotonView targetView)



Network-Destroy the GameObject associated with the **PhotonView**, unless the **PhotonView** is static or not under this client's control.

Destroying a networked GameObject while in a Room includes:

Removal of the Instantiate call from the server's room buffer.

- Removing RPCs buffered for PhotonViews that got created indirectly with the PhotonNetwork.Instantiate call.
- Sending a message to other clients to remove the GameObject also (affected by network lag).

Usually, when you leave a room, the GOs get destroyed automatically. If you have to destroy a GO while not in a room, the Destroy is only done locally.

Destroying networked objects works only if they got created with **PhotonNetwork.Instantiate()**. Objects loaded with a scene are ignored, no matter if they have **PhotonView** components.

The GameObject must be under this client's control:

- Instantiated and owned by this client.
- Instantiated objects of players who left the room are controlled by the Master Client.
- Scene-owned game objects are controlled by the Master Client.
- GameObject can be destroyed while client is not in a room.

#### Returns

Nothing. Check error debug log for any issues.

### static void PhotonNetwork.Destroy

(GameObject targetGo)



Network-Destroy the GameObject, unless it is static or not under this client's control.

Destroying a networked GameObject includes:

- Removal of the Instantiate call from the server's room buffer.
- Removing RPCs buffered for PhotonViews that got created indirectly with the **PhotonNetwork.Instantiate** call.
- Sending a message to other clients to remove the GameObject also (affected by network lag).

Usually, when you leave a room, the GOs get destroyed automatically. If you have to destroy a GO while not in a room, the

Destroy is only done locally.

Destroying networked objects works only if they got created with **PhotonNetwork.Instantiate()**. Objects loaded with a scene are ignored, no matter if they have **PhotonView** components.

The GameObject must be under this client's control:

- Instantiated and owned by this client.
- Instantiated objects of players who left the room are controlled by the Master Client.
- Scene-owned game objects are controlled by the Master Client.
- GameObject can be destroyed while client is not in a room.

#### **Returns**

Nothing. Check error debug log for any issues.

#### static void PhotonNetwork.DestroyAll ( )

static

Network-Destroy all GameObjects, PhotonViews and their RPCs in the room. Removes anything buffered from the server. Can only be called by Master Client (for anyone).

Can only be called by Master Client (for anyone). Unlike the Destroy methods, this will remove anything from the server's room buffer. If your game buffers anything beyond Instantiate and RPC calls, that will be cleaned as well from server.

Destroying all includes:

- Remove anything from the server's room buffer (Instantiate, RPCs, anything buffered).
- Sending a message to other clients to destroy everything locally, too (affected by network lag).

Destroying networked objects works only if they got created with **PhotonNetwork.Instantiate()**. Objects loaded with a scene are ignored, no matter if they have **PhotonView** components.

#### Returns

Nothing. Check error debug log for any issues.

#### static void

### PhotonNetwork.DestroyPlayerObjects (PhotonPlayer targetPlayer

Network-Destroy all GameObjects, PhotonViews and their RPCs of targetPlayer. Can only be called on local player (for "self") or Master Cli anyone).

Destroying a networked GameObject includes:

- Removal of the Instantiate call from the server's room buffer.
- Removing RPCs buffered for PhotonViews that got created indirect the PhotonNetwork.Instantiate call.
- Sending a message to other clients to remove the GameObject als (affected by network lag).

Destroying networked objects works only if they got created with **PhotonNetwork.Instantiate()**. Objects loaded with a scene are ignored matter if they have **PhotonView** components.

#### Returns

Nothing. Check error debug log for any issues.

#### static void

PhotonNetwork.DestroyPlayerObjects (int targetPlayerId)



Network-Destroy all GameObjects, PhotonViews and their RPCs of this player (by ID). Can only be called on local player (for "self") or Master Client (for anyone).

Destroying a networked GameObject includes:

- Removal of the Instantiate call from the server's room buffer.
- Removing RPCs buffered for PhotonViews that got created indirectly with the PhotonNetwork.Instantiate call.
- Sending a message to other clients to remove the GameObject also (affected by network lag).

Destroying networked objects works only if they got created with **PhotonNetwork.Instantiate()**. Objects loaded with a scene are ignored, no matter if they have **PhotonView** components.

#### **Returns**

Nothing. Check error debug log for any issues.

#### static void PhotonNetwork.Disconnect ( )

static

Makes this client disconnect from the photon server, a process that leaves any room and calls OnDisconnectedFromPhoton on completion.

When you disconnect, the client will send a "disconnecting" message to the server. This speeds up leave/disconnect messages for players in the same room as you (otherwise the server would timeout this client's connection). When used in offlineMode, the state-change and event-call OnDisconnectedFromPhoton are immediate. Offline mode is set to false as well. Once disconnected, the client can connect again. Use ConnectUsingSettings.

```
delegate void PhotonNetwork.EventCallback ( byte eventCode, object content, int senderId )
```

Defines the delegate usable in OnEventCall.

Any eventCode < 200 will be forwarded to your delegate(s).

#### **Parameters**

eventCode The code assigend to the incoming event.

**content** The content the sender put into the event.

**senderId** The ID of the player who sent the event. It might be

0, if the "room" sent the event.

#### static void PhotonNetwork.FetchServerTimestamp ( )

static

Refreshes the server timestamp (async operation, takes a roundtrip).

Can be useful if a bad connection made the timestamp unusable or imprecise.

#### static bool PhotonNetwork.FindFriends

(string[] friendsToFind) static



Requests the rooms and online status for a list of friends and saves the result in **PhotonNetwork.Friends**.

Works only on Master Server to find the rooms played by a selected list of users.

The result will be stored in **PhotonNetwork.Friends** when available. That list is initialized on first use of OpFindFriends (before that, it is null). To refresh the list, call FindFriends again (in 5 seconds or 10 or 20).

Users identify themselves by setting a unique userId in the PhotonNetwork.AuthValues. See remarks of **AuthenticationValues** for info about how this is set and used.

The list of friends must be fetched from some other source (not provided by **Photon**).

Internal: The server response includes 2 arrays of info (each index matching a friend from the request):

ParameterCode.FindFriendsResponseOnlineList = bool[] of online states ParameterCode.FindFriendsResponseRoomIdList = string[] of room names (empty string if not in a room)

#### **Parameters**

**friendsToFind** Array of friend (make sure to use unique playerName or AuthValues).

#### Returns

If the operation could be sent (requires connection, only one request is allowed at any time). Always false in offline mode.

## static HashSet<GameObject> PhotonNetwork.FindGameObjectsWithComponent ( Type type )

Finds the GameObjects with Components of a specific type (using FindObjectsOfType).

#### **Parameters**

type Type must be a Component

#### Returns

HashSet with GameObjects that have a specific type of Componer

# static bool PhotonNetwork.GetCustomRoomList (TypedLobby typedLobby, string sqlLobbyFilter )

Fetches a custom list of games from the server, matching a SQL-like "v clause, then triggers OnReceivedRoomListUpdate callback.

Operation is only available for lobbies of type SqlLobby. Note: You don'to join that lobby. This is an async request.

When done, OnReceivedRoomListUpdate gets called. Use **GetRoomL** access it.

http://doc.photonengine.com/en-us/pun/current/manuals-and-demos/matchmaking-and-lobby::sql\_lobby\_type

#### **Parameters**

**typedLobby** The lobby to query. Has to be of type SqlLobby. **sqlLobbyFilter** The sql query statement.

6

#### Returns

If the operation could be sent (has to be connected).

#### static int PhotonNetwork.GetPing ( )

static

The current roundtrip time to the photon server.

#### **Returns**

Roundtrip time (to server and back).

#### static RoomInfo [] PhotonNetwork.GetRoomList ( )

static

Gets currently known rooms as **RoomInfo** array. This is available and updated while in a lobby (check insideLobby).

This list is a cached copy of the internal rooms list so it can be accessed each frame if needed. Per **RoomInfo** you can check if the room is full by comparing playerCount and MaxPlayers before you allow a join.

The name of a room must be used to join it (via JoinRoom).

Closed rooms are also listed by lobbies but they can't be joined. While in a room, any player can set **Room.visible** and **Room.open** to hide rooms from matchmaking and close them.

#### Returns

**RoomInfo**[] of current rooms in lobby.

## static void PhotonNetwork.InitializeSecurity ( )

static

Used for compatibility with Unity networking only. Encryption is automatically initialized while connecting.

#### static GameObject

```
PhotonNetwork.Instantiate (string prefabName, Vector3 position, Quaternion rotation, int group )
```

Instantiate a prefab over the network. This prefab needs to be located in the root of a "Resources" folder.

Instead of using prefabs in the Resources folder, you can manually Instantiate and assign PhotonViews. See doc.

#### **Parameters**

**prefabName** Name of the prefab to instantiate.

position Position Vector3 to apply on instantiation.rotation Rotation Quaternion to apply on instantiation.

**group** The group for this **PhotonView**.

#### **Returns**

The new instance of a GameObject with initialized **PhotonView**.

```
static GameObject
PhotonNetwork.Instantiate

(string prefabName,
Vector3 position,
Quaternion rotation,
int group,
object[] data
)

static
```

Instantiate a prefab over the network. This prefab needs to be located in the root of a "Resources" folder.

Instead of using prefabs in the Resources folder, you can manually Instantiate and assign PhotonViews. See doc.

#### **Parameters**

**prefabName** Name of the prefab to instantiate.

position Position Vector3 to apply on instantiation.rotation Rotation Quaternion to apply on instantiation.

**group** The group for this **PhotonView**.

data Optional instantiation data. This will be saved to

it's PhotonView.instantiationData.

#### **Returns**

The new instance of a GameObject with initialized PhotonView.

static GameObject
PhotonNetwork.InstantiateSceneObject ( string prefabName, Vector3 position,
Ouaternion rotation

Quaternion rotation, int group, object[] data

)

Instantiate a scene-owned prefab over the network. The PhotonViews v be controllable by the MasterClient. This prefab needs to be located in 1 root of a "Resources" folder.

Only the master client can Instantiate scene objects. Instead of using prefabs in the Resources folder, you can manually Instantiate and assiç PhotonViews. See doc.

#### **Parameters**

**prefabName** Name of the prefab to instantiate.

position Position Vector3 to apply on instantiation.rotation Rotation Quaternion to apply on instantiation.

**group** The group for this **PhotonView**.

data Optional instantiation data. This will be saved to it's

PhotonView.instantiationData.

#### Returns

The new instance of a GameObject with initialized PhotonView.

#### static bool PhotonNetwork.JoinLobby ( )

static

On MasterServer this joins the default lobby which list rooms currently in use.

The room list is sent and refreshed by the server. You can access this cached list by **PhotonNetwork.GetRoomList()**.

Per room you should check if it's full or not before joining. **Photon** also lists rooms that are full, unless you close and hide them (room.open = false and room.visible = false).

In best case, you make your clients join random games, as described here:

http://doc.exitgames.com/en/realtime/current/reference/matchmaking-and-lobby

You can show your current players and room count without joining a lobby (but you must be on the master server). Use: countOfPlayers, countOfPlayersOnMaster, countOfPlayersInRooms and countOfRooms.

You can use more than one lobby to keep the room lists shorter. See **JoinLobby(TypedLobby lobby)**. When creating new rooms, they will be "attached" to the currently used lobby or the default lobby.

You can use JoinRandomRoom without being in a lobby! Set autoJoinLobby = false before you connect, to not join a lobby. In that case, the connect-workflow will call OnConnectedToMaster (if you implement it) when it's done.

#### static bool PhotonNetwork.JoinLobby

(TypedLobby typedLobby)



On a Master Server you can join a lobby to get lists of available rooms.

The room list is sent and refreshed by the server. You can access

this cached list by PhotonNetwork.GetRoomList().

Any client can "make up" any lobby on the fly. Splitting rooms into multiple lobbies will keep each list shorter. However, having too many lists might ruin the matchmaking experience.

In best case, you create a limited number of lobbies. For example, create a lobby per game-mode: "koth" for king of the hill and "ffa" for free for all, etc.

There is no listing of lobbies at the moment.

Sql-typed lobbies offer a different filtering model for random matchmaking. This might be more suited for skillbased-games. However, you will also need to follow the conventions for naming filterable properties in sql-lobbies! Both is explained in the matchmaking doc linked below.

In best case, you make your clients join random games, as described here:

http://confluence.exitgames.com/display/PTN/Op+JoinRandomGame

Per room you should check if it's full or not before joining. **Photon** does list rooms that are full, unless you close and hide them (room.open = false and room.visible = false).

You can show your games current players and room count without joining a lobby (but you must be on the master server). Use: countOfPlayers, countOfPlayersOnMaster, countOfPlayersInRooms and countOfRooms.

When creating new rooms, they will be "attached" to the currently used lobby or the default lobby.

You can use JoinRandomRoom without being in a lobby! Set autoJoinLobby = false before you connect, to not join a lobby. In that case, the connect-workflow will call OnConnectedToMaster (if you implement it) when it's done.

#### **Parameters**

**typedLobby** A typed lobby to join (must have name and type).

```
static bool
PhotonNetwork.JoinOrCreateRoom ( string roomName,
RoomOptions roomOptions,
TypedLobby
)
```

Lets you either join a named room or create it on the fly - you don't have know if someone created the room already.

This makes it easier for groups of players to get into the same room. Of the group exchanged a roomName, any player can call JoinOrCreateRo and it doesn't matter who actually joins or creates the room.

The parameters roomOptions and typedLobby are only used when the actually gets created by this client. You know if this client created a roor you get a callback OnCreatedRoom (before OnJoinedRoom gets callec well).

#### **Parameters**

**roomName** Name of the room to join. Must be non null.

roomOptions Options for the room, in case it does not exist yet. E

these values are ignored.

**typedLobby** Lobby you want a new room to be listed in. Ignored

the room was existing and got joined.

#### Returns

If the operation got queued and will be sent.

```
static bool
PhotonNetwork.JoinOrCreateRoom ( string roomName,
RoomOptions roomOptions,
TypedLobby typedLobby,
string[] expectedUsers
```

Lets you either join a named room or create it on the fly - you don't have know if someone created the room already.

This makes it easier for groups of players to get into the same room. Of the group exchanged a roomName, any player can call JoinOrCreateRo and it doesn't matter who actually joins or creates the room.

The parameters roomOptions and typedLobby are only used when the actually gets created by this client. You know if this client created a roor you get a callback OnCreatedRoom (before OnJoinedRoom gets callec well).

You can define an array of expectedUsers, to block player slots in the refor these users. The corresponding feature in **Photon** is called "Slot Reservation" and can be found in the doc pages.

#### **Parameters**

**roomName** Name of the room to join. Must be non null.

**roomOptions** Options for the room, in case it does not exist yet.

these values are ignored.

**typedLobby** Lobby you want a new room to be listed in. Ignore

the room was existing and got joined.

expectedUsers Optional list of users (by UserId) who are expecte

join this game and who you want to block a slot fo

#### Returns

If the operation got queued and will be sent.

#### static bool PhotonNetwork.JoinRandomRoom ( )

static

Joins any available room of the currently used lobby and fails if none is available.

Rooms can be created in arbitrary lobbies which get created on demand. You can join rooms from any lobby without actually joining the lobby. Use the JoinRandomRoom overload with **TypedLobby** parameter.

This method will only match rooms attached to one lobby! If you use many lobbies, you might have to repeat JoinRandomRoom, to find some fitting room. This method looks up a room in the currently active lobby or (if no lobby is joined) in the default lobby.

If this fails, you can still create a room (and make this available for the next who uses JoinRandomRoom). Alternatively, try again in a moment.

# static bool PhotonNetwork.JoinRandomRoom (Hashtable expectedCustomR byte expectedMaxPlaye

Attempts to join an open room with fitting, custom properties but fails if available.

Rooms can be created in arbitrary lobbies which get created on deman from any lobby without actually joining the lobby. Use the JoinRandomF **TypedLobby** parameter.

This method will only match rooms attached to one lobby! If you use making the match to repeat JoinRandomRoom, to find some fitting room. This room in the currently active lobby or (if no lobby is joined) in the default

If this fails, you can still create a room (and make this available for the r JoinRandomRoom). Alternatively, try again in a moment.

#### **Parameters**

**expectedCustomRoomProperties** Filters for rooms that match th properties (string keys and va

null.

expectedMaxPlayers

Filters for a particular maxplay accept any maxPlayer value.

#### Returns

If the operation got queued and will be sent.

static bool	
PhotonNetwork.JoinRandomRoom ( Hashtable	expected(
byte	expected
MatchmakingMode	matching <sup>*</sup>
TypedLobby	typedLob
string	sqlLobbyl
string[]	expectedl
)	

Attempts to join an open room with fitting, custom properties but fails if

Rooms can be created in arbitrary lobbies which get created on deman lobby without actually joining the lobby with this overload.

This method will only match rooms attached to one lobby! If you use make repeat JoinRandomRoom, to find some fitting room. This method looks lobby or the currently active lobby (if none specified) or in the default lo

If this fails, you can still create a room (and make this available for the r JoinRandomRoom). Alternatively, try again in a moment.

In offlineMode, a room will be created but no properties will be set and a JoinRandomRoom call are ignored. The event/callback OnJoinedRoom PhotonNetworkingMessage).

You can define an array of expectedUsers, to block player slots in the recorresponding feature in **Photon** is called "Slot Reservation" and can b

#### **Parameters**

expectedCustomRoomProperties	Filters for rooms that match the keys and values). To ignore, p
expectedMaxPlayers	Filters for a particular maxplay maxPlayer value.
matchingType	Selects one of the available m MatchmakingMode enum for c
typedLobby	The lobby in which you want t use the default lobby. This doe neither sets the lobby property

# sqlLobbyFilter expectedUsers

A filter-string for SQL-typed lo Optional list of users (by User this game and who you want t

#### Returns

If the operation got queued and will be sent.

#### static bool PhotonNetwork.JoinRoom (string roomName)

static

Join room by roomname and on success calls **OnJoinedRoom()**. This is not affected by lobbies.

On success, the method **OnJoinedRoom()** is called on any script. You can implement it to react to joining a room.

JoinRoom fails if the room is either full or no longer available (it might become empty while you attempt to join). Implement **OnPhotonJoinRoomFailed()** to get a callback in error case.

To join a room from the lobby's listing, use **RoomInfo.Name** as roomName here. Despite using multiple lobbies, a roomName is always "global" for your application and so you don't have to specify which lobby it's in. The Master Server will find the room. In the **Photon** Cloud, an application is defined by Appld, Game- and PUNversion.

PhotonNetworkingMessage.OnPhotonJoinRoomFailed PhotonNetworkingMessage.OnJoinedRoom

#### **Parameters**

**roomName** Unique name of the room to join.

#### Returns

If the operation got queued and will be sent.

static bool PhotonNetwork.JoinRoom

(string roomName, string∏ expectedUsers

static

Join room by roomname and on success calls **OnJoinedRoom()**. This is not affected by lobbies.

On success, the method **OnJoinedRoom()** is called on any script. You can implement it to react to joining a room.

JoinRoom fails if the room is either full or no longer available (it might become empty while you attempt to join). Implement **OnPhotonJoinRoomFailed()** to get a callback in error case.

To join a room from the lobby's listing, use **RoomInfo.Name** as roomName here. Despite using multiple lobbies, a roomName is always "global" for your application and so you don't have to specify which lobby it's in. The Master Server will find the room. In the **Photon** Cloud, an application is defined by Appld, Game- and PUNversion.

You can define an array of expectedUsers, to block player slots in the room for these users. The corresponding feature in **Photon** is called "Slot Reservation" and can be found in the doc pages.

PhotonNetworkingMessage.OnPhotonJoinRoomFailed PhotonNetworkingMessage.OnJoinedRoom

#### **Parameters**

**roomName** Unique name of the room to join.

**expectedUsers** Optional list of users (by UserId) who are

expected to join this game and who you want

to block a slot for.

#### Returns

If the operation got queued and will be sent.

#### static bool PhotonNetwork.LeaveLobby ( )

static

Leave a lobby to stop getting updates about available rooms.

This does not reset **PhotonNetwork.lobby**! This allows you to join this particular lobby later easily.

The values countOfPlayers, countOfPlayersOnMaster, countOfPlayersInRooms and countOfRooms are received even without being in a lobby.

You can use JoinRandomRoom without being in a lobby. Use autoJoinLobby to not join a lobby when you connect.

#### static bool PhotonNetwork.LeaveRoom ( )

static

Leave the current room and return to the Master Server where you can join or create rooms (see remarks).

This will clean up all (network) GameObjects with a **PhotonView**, unless you changed autoCleanUp to false. Returns to the Master Server.

In OfflineMode, the local "fake" room gets cleaned up and OnLeftRoom gets called immediately.

### static void PhotonNetwork.LoadLevel (int levelNumber)

static

Wraps loading a level to pause the network mesage-queue. Optionally syncs the loaded level in a room.

To sync the loaded level in a room, set **PhotonNetwork.automaticallySyncScene** to true. The Master Client of a room will then sync the loaded level with every other player in the room.

While loading levels, it makes sense to not dispatch messages received by other players. This method takes care of that by setting **PhotonNetwork.isMessageQueueRunning** = false and enabling the queue when the level was loaded.

You should make sure you don't fire RPCs before you load another

scene (which doesn't contain the same GameObjects and PhotonViews). You can call this in OnJoinedRoom.

This uses Application.LoadLevel.

#### **Parameters**

levelNumber Number of the level to load. When using level numbers, make sure they are identical on all clients.

#### static void PhotonNetwork.LoadLevel (string levelName)

static

Wraps loading a level to pause the network mesage-queue. Optionally syncs the loaded level in a room.

While loading levels, it makes sense to not dispatch messages received by other players. This method takes care of that by setting **PhotonNetwork.isMessageQueueRunning** = false and enabling the queue when the level was loaded.

To sync the loaded level in a room, set **PhotonNetwork.automaticallySyncScene** to true. The Master Client of a room will then sync the loaded level with every other player in the room.

You should make sure you don't fire RPCs before you load another scene (which doesn't contain the same GameObjects and PhotonViews). You can call this in OnJoinedRoom.

This uses Application.LoadLevel.

#### **Parameters**

**levelName** Name of the level to load. Make sure it's available to all clients in the same room.

#### static void PhotonNetwork.NetworkStatisticsReset ( )

static

Resets the traffic stats and re-enables them.

#### static string PhotonNetwork.NetworkStatisticsToString ( )

static

Only available when NetworkStatisticsEnabled was used to gather some stats.

#### Returns

A string with vital networking statistics.

#### static void

PhotonNetwork.OverrideBestCloudServer (CloudRegionCode reg

Overwrites the region that is used for ConnectToBestCloudServer(str gameVersion).

This will overwrite the result of pinging all cloud servers.

Use this to allow your users to save a manually selected region in the p preferences.

Note: You can also use **PhotonNetwork.ConnectToRegion** to (tempor connect to a specific region.

#### static bool

PhotonNetwork.RaiseEvent (byte eventCode, object eventContent, bool sendReliable, RaiseEventOptions options )

Sends fully customizable events in a room. Events consist of at least ar EventCode (0..199) and can have content.

To receive the events someone sends, register your handling method ir PhotonNetwork.OnEventCall.

Example: private void OnEventHandler(byte eventCode, object content int senderId) { Debug.Log("OnEventHandler"); }

PhotonNetwork.OnEventCall += this.OnEventHandler;

With the senderld, you can look up the **PhotonPlayer** who sent the event. It is best practice to assign a eventCode for each different type o content and action. You have to cast the content.

The eventContent is optional. To be able to send something, it must be "serializable type", something that the client can turn into a byte[] basically. Most basic types and arrays of them are supported, including Unity's Vector2, Vector3, Quaternion. Transforms or classes some projections are NOT supported! You can make your own class a "serializab type" by following the example in **CustomTypes.cs**.

The **RaiseEventOptions** have some (less intuitive) combination rules: you set targetActors (an array of **PhotonPlayer.ID** values), the receiver parameter gets ignored. When using event caching, the targetActors, receivers and interestGroup can't be used. Buffered events go to all. When using cachingOption removeFromRoomCache, the eventCode a content are actually not sent but used as filter.

#### **Parameters**

eventCode A byte identifying the type of event. You might want

to use a code per action or to signal which content

can be expected. Allowed: 0..199.

eventContent Some serializable object like string, byte, integer,

float (etc) and arrays of those. Hashtables with byte

keys are good to send variable content.

sendReliable Makes sure this event reaches all players. It gets

acknowledged, which requires bandwidth and it car

be skipped (might add lag in case of loss).

**options** Allows more complex usage of events. If null,

RaiseEventOptions.Default will be used (which is

fine).

#### Returns

False if event could not be sent

#### static bool PhotonNetwork.Reconnect ( )

static

Can be used to reconnect to the master server after a disconnect.

After losing connection, you can use this to connect a client to the region Master Server again. Cache the room name you're in and use ReJoin(roomname) to return to a game. Common use case: Press the Lock Button on a iOS device and you get disconnected immediately.

#### static bool PhotonNetwork.ReconnectAndRejoin ( )

static

When the client lost connection during gameplay, this method attempts to reconnect and rejoin the room.

This method re-connects directly to the game server which was hosting the room PUN was in before. If the room was shut down in the meantime, PUN will call OnPhotonJoinRoomFailed and return this client to the Master Server.

Check the return value, if this client will attempt a reconnect and rejoin (if the conditions are met). If ReconnectAndRejoin returns false, you can still attempt a Reconnect and ReJoin.

Similar to PhotonNetwork.ReJoin, this requires you to use unique IDs per player (the UserID).

#### **Returns**

False, if there is no known room or game server to return to. Then, this client does not attempt the ReconnectAndRejoin.

#### static void PhotonNetwork.RefreshCloudServerRating ( )

static

Pings all cloud servers again to find the one with best ping (currently).

#### static bool PhotonNetwork.ReJoinRoom

(string roomName) static

Can be used to return to a room after a disconnect and reconnect.

After losing connection, you might be able to return to a room and continue playing, if the client is reconnecting fast enough. Use Reconnect() and this method. Cache the room name you're in and use ReJoin(roomname) to return to a game.

Note: To be able to ReJoin any room, you need to use UserIDs! You also need to set RoomOptions.PlayerTtl.

Important: Instantiate() and use of RPCs is not yet supported. The ownership rules of PhotonViews prevent a seamless return to a game. Use Custom Properties and RaiseEvent with event caching instead.

Common use case: Press the Lock Button on a iOS device and you get disconnected immediately.

#### static void PhotonNetwork.RemovePlayerCustomProperties (string[] custom

Locally removes Custom Properties of "this" player. Important: This doe change! Useful when you switch rooms.

Use this method with care. It can create inconsistencies of state betwee changes the player.customProperties locally. This can be useful to clea between games (let's say they store which turn you made, kills, etc).

SetPlayerCustomProperties() syncs and can be used to set values to can be considered "removed" while in a room.

If customPropertiesToDelete is null or has 0 entries, all Custom Propert with a new Hashtable). If you specify keys to remove, those will be rem but other keys are unaffected.

#### **Parameters**

#### static void

PhotonNetwork.RemoveRPCs (PhotonPlayer targetPlayer)

static

Remove all buffered RPCs from server that were sent by targetPlayer. Can only be called on local player (for "self") or Master Client (for anyone).

This method requires either:

- This is the targetPlayer's client.
- This client is the Master Client (can remove any **PhotonPlayer**'s RPCs).

If the targetPlayer calls RPCs at the same time that this is called, network lag will determine if those get buffered or cleared like the rest.

#### **Parameters**

**targetPlayer** This player's buffered RPCs get removed from server buffer.

#### static void

PhotonNetwork.RemoveRPCs (PhotonView targetPhotonView)

Remove all buffered RPCs from server that were sent via targetPhotonView. The Master Client and the owner of the targetPhotonView may call this.

This method requires either:

- The targetPhotonView is owned by this client (Instantiated by it).
- This client is the Master Client (can remove any PhotonView's RPCs).

#### **Parameters**

#### targetPhotonView RPCs buffered for this PhotonView get remov from server buffer.

#### static void

PhotonNetwork.RemoveRPCsInGroup (int targetGroup) static



Remove all buffered RPCs from server that were sent in the targetGroup, if this is the Master Client or if this controls the individual PhotonView.

This method requires either:

- This client is the Master Client (can remove any RPCs per group).
- Any other client: each **PhotonView** is checked if it is under this client's control. Only those RPCs are removed.

#### **Parameters**

targetGroup Interest group that gets all RPCs removed.

#### static void PhotonNetwork.SendOutgoingCommands ( )

static

Can be used to immediately send the RPCs and Instantiates just called, so they are on their way to the other players.

This could be useful if you do a RPC to load a level and then load it yourself. While loading, no RPCs are sent to others, so this would delay the "load" RPC. You can send the RPC to "others", use this method, disable the message queue (by isMessageQueueRunning) and then load.

### static void PhotonNetwork.SetLevelPrefix (short prefix)

static

Sets level prefix for PhotonViews instantiated later on. Don't set it if you need only one!

Important: If you don't use multiple level prefixes, simply don't set this value. The default value is optimized out of the traffic.

This won't affect existing PhotonViews (they can't be changed yet for existing PhotonViews).

Messages sent with a different level prefix will be received but not executed. This affects RPCs, Instantiates and synchronization.

Be aware that PUN never resets this value, you'll have to do so yourself.

#### **Parameters**

prefix Max value is short.MaxValue = 32767

# static bool PhotonNetwork.SetMasterClient (PhotonPlayer masterClientPlayer

Asks the server to assign another player as Master Client of your currer

RPCs and RaiseEvent have the option to send messages only to the M Client of a room. SetMasterClient affects which client gets those messa

This method calls an operation on the server to set a new Master Client takes a roundtrip. In case of success, this client and the others get the Master Client from the server.

SetMasterClient tells the server which current Master Client should be r with the new one. It will fail, if anything switches the Master Client mom earlier. There is no callback for this error. All clients should get the new Client assigned by the server anyways.

See also: PhotonNetwork.masterClient

On v3 servers: The ReceiverGroup.MasterClient (usable in RPCs) is no affected by this (still points to lowest player.ID in room). Avoid using this value (and send to a specific player instead).

If the current Master Client leaves, PUN will detect a new one by "lowes ID". Implement OnMasterClientSwitched to get a callback in this case.

PUN-selected Master Client might assign a new one.

Make sure you don't create an endless loop of Master-assigning! When selecting a custom Master Client, all clients should point to the same pl matter who actually assigns this player.

Locally the Master Client is immediately switched, while remote clients event. This means the game is tempoarily without Master Client like wh current Master Client leaves.

When switching the Master Client manually, keep in mind that this user leave and not do it's work, just like any Master Client.

#### **Parameters**

masterClientPlayer The player to become the next Master Client.

#### **Returns**

False when this operation couldn't be done. Must be in a room (not offlineMode).

#### static void

## PhotonNetwork.SetPlayerCustomProperties (Hashtable customPr

Sets this (local) player's properties and synchronizes them to the other modify them directly).

While in a room, your properties are synced with the other players. Creation and JoinRandomRoom will all apply your player's custom proyou enter the room. The whole Hashtable will get sent. Minimize the train only updated key/values.

If the Hashtable is null, the custom properties will be cleared. Custom p never cleared automatically, so they carry over to the next room, if you them.

Don't set properties by modifying PhotonNetwork.player.customPropert

#### **Parameters**

customProperties Only string-typed keys will be used from this ha

### custom properties are all deleted.

```
static void PhotonNetwork.SetReceivingEnabled
```

(int group, bool enabled

static

Enable/disable receiving on given group (applied to PhotonViews)

#### **Parameters**

group The interest group to affect.enabled Sets if receiving from group to enabled (or not).

#### static void

PhotonNetwork.SetReceivingEnabled (int[] enableGroups, int[] disableGroups)

static

Enable/disable receiving on given groups (applied to PhotonViews)

#### **Parameters**

**enableGroups** The interest groups to enable (or null). **disableGroups** The interest groups to disable (or null).

#### static void

PhotonNetwork.SetSendingEnabled

(int group, bool enabled

static

Enable/disable sending on given group (applied to PhotonViews)

#### **Parameters**

group The interest group to affect.enabled Sets if sending to group is enabled (or not).

## 

Enable/disable sending on given groups (applied to PhotonViews)

#### **Parameters**

**enableGroups** The interest groups to enable sending on (or null).

**disableGroups** The interest groups to disable sending on (or null).

#### static void

PhotonNetwork.SwitchToProtocol (ConnectionProtocol cp)



While offline, the network protocol can be switched (which affects the ports you can use to connect).

When you switch the protocol, make sure to also switch the port for the master server. Default ports are: TCP: 4530 UDP: 5055

This could look like this:

Connect(serverAddress, <udpport|tcpport>, appID, gameVersion)

Or when you use **ConnectUsingSettings()**, the PORT in the settings can be switched like so:

PhotonNetwork.PhotonServerSettings.ServerPort = 4530;

The current protocol can be read this way:

Photon Network. networking Peer. Used Protocol

This does not work with the native socket plugin of PUN+ on mobile!

#### **Parameters**

**cp** Network protocol to use as low level connection. UDP is

default. TCP is not available on all platforms (see remarks).

## static void PhotonNetwork.UnAllocateViewID (int viewID)

static

Unregister a viewID (of manually instantiated and destroyed networked objects).

#### **Parameters**

**viewID** A viewID manually allocated by this player.

## static bool PhotonNetwork.WebRpc (string name, object parameters

This operation makes **Photon** call your custom web-service by name (path) with the given parameters.

This is a server-side feature which must be setup in the **Photon** Cloud Dashboard prior to use.

See the Turnbased Feature Overview for a short intro. http://doc.photonengine.com/en/turnbased/current/gettingstarted/feature-overview br/> The Parameters will be converted into JSon format, so make sure your parameters are compatible.

See PhotonNetworkingMessage.OnWebRpcResponse on how to get a response.

It's important to understand that the OperationResponse only tells if the WebRPC could be called. The content of the response contains any values your web-service sent and the error/success code. In case the web-service failed, an error code and a debug message are usually inside the OperationResponse.

The class **WebRpcResponse** is a helper-class that extracts the most valuable content from the WebRPC response.

Example callback implementation:

```
public void OnWebRpcResponse(OperationResponse response)
{
    WebRpcResponse webResponse = new WebRpcResponse(operationResponse)
    if (webResponse.ReturnCode != 0) { //...
    }
    switch (webResponse.Name) { //...
    }
    // and so on
}
```

## Member Data Documentation

#### float PhotonNetwork.BackgroundTimeout = 60.0f

static

Defines how many seconds PUN keeps the connection, after Unity's OnApplicationPause(true) call. Default: 60 seconds.

It's best practice to disconnect inactive apps/connections after a while but to also allow users to take calls, etc.. We think a reasonable backgroung timeout is 60 seconds.

To handle the timeout, implement: **OnDisconnectedFromPhoton()**, as usual. Your application will "notice" the background disconnect when it becomes active again (running the Update() loop).

If you need to separate this case from others, you need to track if the app was in the background (there is no special callback by PUN).

A value below 0.1 seconds will disable this timeout (careful: connections can be kept indefinitely).

Info: PUN is running a "fallback thread" to send ACKs to the server, even when Unity is not calling Update() regularly. This helps keeping the connection while loading scenes and assets and when the app is in the background.

Note: Some platforms (e.g. iOS) don't allow to keep a connection while the app is in background. In those cases, this value does not change anything, the app immediately loses connection in background.

Unity's OnApplicationPause() callback is broken in some exports (Android) of some Unity versions. Make sure OnApplicationPause() gets the callbacks you'd expect on the platform you target! Check PhotonHandler.OnApplicationPause(bool pause), to see the implementation.

## bool PhotonNetwork.InstantiateInRoomOnly = true

static

If true, Instantiate methods will check if you are in a room and fail if you are not.

Instantiating anything outside of a specific room is very likely to break things. Turn this off only if you know what you do.

## PhotonLogLevel PhotonNetwork.logLevel = PhotonLogLevel.ErrorsOnly

static

Network log level. Controls how verbose PUN is.

#### readonly int PhotonNetwork.MAX\_VIEW\_IDS = 1000

static

The maximum number of assigned PhotonViews *per player* (or scene). See the **General Documentation** topic "Limitations" on how to raise this limitation.

### int PhotonNetwork.maxConnections

static

Only used in Unity Networking. In PUN, set the number of players in **PhotonNetwork.CreateRoom**.

#### EventCallback PhotonNetwork.OnEventCall

static

Register your RaiseEvent handling methods here by using "+=".

Any eventCode < 200 will be forwarded to your delegate(s).

#### RaiseEvent

# ServerSettings PhotonNetwork.PhotonServerSettings = (ServerSettings)Resources.Load(PhotonNetwork.serverSettingsAstypeof(ServerSettings))

Serialized server settings, written by the Setup Wizard for use in ConnectUsingSettings.

## float PhotonNetwork.precisionForFloatSynchronization = 0.01f

static

The minimum difference between floats before we send it via a **PhotonView**'s OnSerialize/ObservingComponent.

#### float

## PhotonNetwork.precisionForQuaternionSynchronization = 1.0f

static

The minimum angle that a rotation needs to change before we send it via a **PhotonView**'s OnSerialize/ObservingComponent.

## float PhotonNetwork.precisionForVectorSynchronization = 0.000099f

stati<u>c</u>

The minimum difference that a Vector2 or Vector3(e.g. a transforms rotation) needs to change before we send it via a **PhotonView**'s OnSerialize/ObservingComponent.

Note that this is the sqrMagnitude. E.g. to send only after a 0.01 change on the Y-axix, we use 0.01f\*0.01f=0.0001f. As a remedy against float inaccuracy we use 0.000099f instead of 0.0001f.

Dictionary<string, GameObject>
PhotonNetwork.PrefabCache = new Dictionary<string,
GameObject>()

static

Keeps references to GameObjects for frequent instantiation (out of memory instead of loading the Resources).

You should be able to modify the cache anytime you like, except while Instantiate is used. Best do it only in the main-Thread.

## HashSet<GameObject> PhotonNetwork.SendMonoMessageTargets

static

If not null, this is the (exclusive) list of GameObjects that get called by PUN SendMonoMessage().

For all callbacks defined in PhotonNetworkingMessage, PUN will use SendMonoMessage and call FindObjectsOfType() to find all scripts and GameObjects that might want a callback by PUN.

PUN callbacks are not very frequent (in-game, property updates are most frequent) but FindObjectsOfType is time consuming and with a large number of GameObjects, performance might suffer.

Optionally, SendMonoMessageTargets can be used to supply a list of target GameObjects. This skips the FindObjectsOfType() but any GameObject that needs callbacks will have to Add itself to this list.

If null, the default behaviour is to do a SendMessage on each GameObject with a MonoBehaviour.

## Type PhotonNetwork.SendMonoMessageTargetType = typeof(MonoBehaviour)

static

Defines which classes can contain PUN Callback implementations.

This provides the option to optimize your runtime for speed. The more specific this Type is, the fewer classes will be checked with reflection for callback methods.

## **bool PhotonNetwork.StartRpcsAsCoroutine = true**

static

Can be used to skip starting RPCs as Coroutine, which can be a performance issue.

#### bool PhotonNetwork.UsePrefabCache = true

static

While enabled (true), Instantiate uses **PhotonNetwork.PrefabCache** to keep game objects in memory (improving instantiation of the same prefab).

Setting UsePrefabCache to false during runtime will not clear PrefabCache but will ignore it right away. You could clean and modify the cache yourself. Read its comments.

## bool PhotonNetwork.UseRpcMonoBehaviourCache

static

While enabled, the MonoBehaviours on which we call RPCs are cached, avoiding costly GetComponents<MonoBehaviour>() calls.

RPCs are called on the MonoBehaviours of a target **PhotonView**. Those have to be found via GetComponents.

When set this to true, the list of MonoBehaviours gets cached in each **PhotonView**. You can use photonView.RefreshRpcMonoBehaviourCache() to manually refresh a **PhotonView**'s list of MonoBehaviours on demand (when a new MonoBehaviour gets added to a networked GameObject, e.g.).

## const string PhotonNetwork.versionPUN = "1.81"

Version number of PUN. Also used in GameVersion to separate client version from each other.

## **Property Documentation**

#### Authentication Values Photon Network. Auth Values





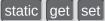


A user's authentication values used during connect.

Set these before calling Connect if you want custom authentication. These values set the userld, if and how that userld gets verified (server-side), etc...

If authentication fails for any values, PUN will call your implementation of OnCustomAuthenticationFailed(string debugMsg). See: PhotonNetworkingMessage.OnCustomAuthenticationFailed

### bool PhotonNetwork.autoCleanUpPlayerObjects







This setting defines per room, if network-instantiated GameObjects (with **PhotonView**) get cleaned up when the creator of it leaves.

This setting is done per room. It can't be changed in the room and it will override the settings of individual clients.

If room. Auto Clean Up is enabled in a room, the PUN clients will destroy a player's GameObjects on leave. This includes GameObjects manually instantiated (via RPCs, e.g.). When enabled, the server will clean RPCs, instantiated GameObjects and PhotonViews of the leaving player, too. and Players who join after someone left, won't get the events of that player anymore.

Under the hood, this setting is stored as a Custom Room Property. Enabled by default.

## bool PhotonNetwork.autoJoinLobby







Set in PhotonServerSettings asset. Defines if the **PhotonNetwork** should join the "lobby" when connected to the Master server.

If this is false, **OnConnectedToMaster()** will be called when connection to the Master is available. OnJoinedLobby() will NOT be called if this is false.

Enabled by default.

The room listing will not become available. Rooms can be created and joined (randomly) without joining the lobby (and getting sent the room list).

### bool PhotonNetwork.automaticallySyncScene







Defines if all clients in a room should load the same level as the Master Client (if that used **PhotonNetwork.LoadLevel**).

To synchronize the loaded level, the Master Client should use PhotonNetwork.LoadLevel. All clients will load the new scene when they get the update or when they join.

Internally, a Custom **Room** Property is set for the loaded scene. When a client reads that and is not in the same scene yet, it will immediately pause the Message Queue

(PhotonNetwork.isMessageQueueRunning = false) and load. When the scene finished loading, PUN will automatically re-enable the Message Queue.

#### bool PhotonNetwork.connected





False until you connected to **Photon** initially. True in offline mode, while connected to any server and even while switching servers.

## bool PhotonNetwork.connectedAndReady





A refined version of connected which is true only if your connection to the server is ready to accept operations like join, leave, etc.

## bool PhotonNetwork.connecting





True when you called ConnectUsingSettings (or similar) until the low level connection to **Photon** gets established.

#### ConnectionState PhotonNetwork.connectionState





Simplified connection state

#### ClientState PhotonNetwork.connectionStateDetailed





Detailed connection state (ignorant of PUN, so it can be "disconnected" while switching servers).

In OfflineMode, this is ClientState. Joined (after create/join) or it is ConnectedToMaster in all other cases.

## int PhotonNetwork.countOfPlayers





The count of players currently using this application (available on MasterServer in 5sec intervals).

## int PhotonNetwork.countOfPlayersInRooms





Count of users currently playing your app in some room (sent every 5sec by Master Server). Use playerList.Count to get the count of players in the room you're in!

## int PhotonNetwork.countOfPlayersOnMaster

static get

The count of players currently looking for a room (available on MasterServer in 5sec intervals).

#### int PhotonNetwork.countOfRooms

static get



The count of rooms currently in use (available on MasterServer in 5sec intervals).

While inside the lobby you can also check the count of listed rooms as: PhotonNetwork.GetRoomList().Length. Since PUN v1.25 this is only based on the statistic event **Photon** sends (counting all rooms).

#### bool PhotonNetwork.CrcCheckEnabled

static get set





Crc checks can be useful to detect and avoid issues with broken datagrams. Can be enabled while not connected.

## bool PhotonNetwork.EnableLobbyStatistics

static get set





Set in PhotonServerSettings asset. Enable to get a list of active lobbies from the Master Server.

Lobby Statistics can be useful if a game uses multiple lobbies and you want to show activity of each to players.

This value is stored in PhotonServerSettings.

PhotonNetwork.LobbyStatistics is updated when you connect to the Master Server. There is also a callback PunBehaviour.

#### List<FriendInfo> PhotonNetwork.Friends

static get set





Read-only list of friends, their online status and the room they are in. Null until initialized by a FindFriends call.

Do not modify this list! It is internally handled by FindFriends and only available to read the values. The value of FriendListAge tells you how old the data is in milliseconds.

Don't get this list more often than useful (> 10 seconds). In best case, keep the list you fetch really short. You could (e.g.) get the full list only once, then request a few updates only for friends who are online. After a while (e.g. 1 minute), you can get the full list again (to update online states).

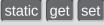
## int PhotonNetwork.FriendsListAge





Age of friend list info (in milliseconds). It's 0 until a friend list is fetched.

## string PhotonNetwork.gameVersion







Version string for your this build. Can be used to separate incompatible clients. Sent during connect.

This is only sent when you connect so that is also the place you set it usually (e.g. in ConnectUsingSettings).

#### bool PhotonNetwork.inRoom





Is true while being in a room (connectionStateDetailed == ClientState.Joined).

Many actions can only be executed in a room, like Instantiate or Leave, etc. You can join a room in offline mode, too.

## bool PhotonNetwork.insideLobby





True while this client is in a lobby.

Implement IPunCallbacks.OnReceivedRoomListUpdate() for a notification when the list of rooms becomes available or updated.

You are automatically leaving any lobby when you join a room! Lobbies only exist on the Master Server (whereas rooms are handled by Game Servers).

#### bool PhotonNetwork.isMasterClient

static get



Are we the master client?

### bool PhotonNetwork.isMessageQueueRunning







Can be used to pause dispatching of incoming evtents (RPCs, Instantiates and anything else incoming).

While IsMessageQueueRunning == false, the OnPhotonSerializeView calls are not done and nothing is sent by a client. Also, incoming messages will be queued until you re-activate the message queue.

This can be useful if you first want to load a level, then go on receiving data of PhotonViews and RPCs. The client will go on receiving and sending acknowledgements for incoming packages and your RPCs/Events. This adds "lag" and can cause issues when the pause is longer, as all incoming messages are just queued.

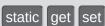
#### bool PhotonNetwork.isNonMasterClientInRoom





True if we are in a room (client) and NOT the room's masterclient

## TypedLobby PhotonNetwork.lobby







The lobby that will be used when PUN joins a lobby or creates a game.

The default lobby uses an empty string as name. PUN will enter a lobby on the Master Server if autoJoinLobby is set to true. So when you connect or leave a room, PUN automatically gets you into a lobby again.

Check **PhotonNetwork.insideLobby** if the client is in a lobby. (Master Server And Lobby)

## List<TypedLobbyInfo> PhotonNetwork.LobbyStatistics

static get set



If turned on, the Master Server will provide information about active lobbies for this application.

Lobby Statistics can be useful if a game uses multiple lobbies and you want to show activity of each to players. Per lobby, you get: name, type, room- and player-count.

PhotonNetwork.LobbyStatistics is updated when you connect to the Master Server. There is also a callback PunBehaviour.OnLobbyStatisticsUpdate, which you should implement to update your UI (e.g.).

Lobby Statistics are not turned on by default. Enable them in the PhotonServerSettings file of the project.

## PhotonPlayer PhotonNetwork.masterClient





The Master Client of the current room or null (outside of rooms).

Can be used as "authoritative" client/player to make descisions, run Al or other.

If the current Master Client leaves the room (leave/disconnect), the server will quickly assign someone else. If the current Master Client times out (closed app, lost connection, etc), messages sent to this client are effectively lost for the others! A timeout can take 10 seconds in which no Master Client is active.

Implement the method IPunCallbacks.OnMasterClientSwitched to be called when the Master Client switched.

Use **PhotonNetwork.SetMasterClient**, to switch manually to some other player / client.

With offlineMode == true, this always returns the PhotonNetwork.player.

#### int PhotonNetwork.MaxResendsBeforeDisconnect



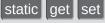




Defines the number of times a reliable message can be resent before not getting an ACK for it will trigger a disconnect. Default: 5.

Less resends mean quicker disconnects, while more can lead to much more lag without helping. Min: 3. Max: 10.

#### bool PhotonNetwork.NetworkStatisticsEnabled



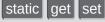




Enables or disables the collection of statistics about this client's traffic.

If you encounter issues with clients, the traffic stats are a good starting point to find solutions. Only with enabled stats, you can use **GetVitalStats** 

#### bool PhotonNetwork.offlineMode







Offline mode can be set to re-use your multiplayer code in singleplayer game modes. When this is on **PhotonNetwork** will not create any connections and there is near to no overhead. Mostly usefull for reusing RPC's and PhotonNetwork.Instantiate

## PhotonPlayer [] PhotonNetwork.otherPlayers





The list of players in the current room, excluding the local player.

This list is only valid, while the client is in a room. It automatically gets updated when someone joins or leaves.

This can be used to list all other players in a room. Each player's PhotonPlayer.customProperties are accessible (set and synchronized via **PhotonPlayer.SetCustomProperties**).

You can use a **PhotonPlayer.TagObject** to store an arbitrary object for reference. That is not synchronized via the network.

## int PhotonNetwork.PacketLossByCrcCheck





If CrcCheckEnabled, this counts the incoming packages that don't have a valid CRC checksum and got rejected.

## PhotonPlayer PhotonNetwork.player





The local **PhotonPlayer**. Always available and represents this player. CustomProperties can be set before entering a room and will be synced as well.

## PhotonPlayer [] PhotonNetwork.playerList





The list of players in the current room, including the local player.

This list is only valid, while the client is in a room. It automatically gets updated when someone joins or leaves.

This can be used to list all players in a room. Each player's PhotonPlayer.customProperties are accessible (set and

synchronized via **PhotonPlayer.SetCustomProperties**).

You can use a **PhotonPlayer.TagObject** to store an arbitrary object for reference. That is not synchronized via the network.

## string PhotonNetwork.playerName







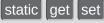
Set to synchronize the player's nickname with everyone in the room(s) you enter. This sets PhotonNetwork.player.NickName.

The playerName is just a nickname and does not have to be unique or backed up with some account.

Set the value any time (e.g. before you connect) and it will be available to everyone you play with.

Access the names of players by: **PhotonPlayer.NickName**. **PhotonNetwork.otherPlayers** is a list of other players - each contains the playerName the remote player set.

#### IPunPrefabPool PhotonNetwork.PrefabPool







An Object Pool can be used to keep and reuse instantiated object instances. It replaced Unity's default Instantiate and Destroy methods.

To use a GameObject pool, implement IPunPrefabPool and assign it here. Prefabs are identified by name.

## int PhotonNetwork.QuickResends







In case of network loss, reliable messages can be repeated quickly up to 3 times.

When reliable messages get lost more than once, subsequent repeats are delayed a bit to allow the network to recover. With this option, the repeats 2 and 3 can be sped up. This can help avoid timeouts but also it increases the speed in which gaps are

closed.

When you set this, increase

PhotonNetwork.MaxResendsBeforeDisconnect to 6 or 7.

#### int PhotonNetwork.ResentReliableCommands

static get



Count of commands that got repeated (due to local repeat-timing before an ACK was received).

If this value increases a lot, there is a good chance that a timeout disconnect will happen due to bad conditions.

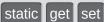
#### Room PhotonNetwork.room





Get the room we're currently in. Null if we aren't in any room.

#### int PhotonNetwork.sendRate



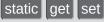




Defines how many times per second **PhotonNetwork** should send a package. If you change this, do not forget to also change 'sendRateOnSerialize'.

Less packages are less overhead but more delay. Setting the sendRate to 50 will create up to 50 packages per second (which is a lot!). Keep your target platform in mind: mobile networks are slower and less reliable.

#### int PhotonNetwork.sendRateOnSerialize





Defines how many times per second OnPhotonSerialize should be called on PhotonViews.

Choose this value in relation to **PhotonNetwork.sendRate**. OnPhotonSerialize will create updates and messages to be sent. A lower rate takes up less performance but will cause more lag.

#### ServerConnection PhotonNetwork.Server

static get



The server (type) this client is currently connected or connecting to.

Photon uses 3 different roles of servers: Name Server, Master Server and Game Server.

## string PhotonNetwork.ServerAddress





Currently used server address (no matter if master or game server).

#### int PhotonNetwork.ServerTimestamp





The current server's millisecond timestamp.

This can be useful to sync actions and events on all clients in one room. The timestamp is based on the server's Environment.TickCount.

It will overflow from a positive to a negative value every so often, so be careful to use only time-differences to check the time delta when things happen.

This is the basis for **PhotonNetwork.time**.

#### double PhotonNetwork.time





**Photon** network time, synched with the server.

v1.55

This time value depends on the server's Environment. TickCount. It is different per server but inside a **Room**, all clients should have the same value (Rooms are on one server only).

This is not a DateTime!

Use this value with care: It can start with any positive value. It will "wrap around" from 4294967.295 to 0!

#### int PhotonNetwork.unreliableCommandsLimit

static get set





Used once per dispatch to limit unreliable commands per channel (so after a pause, many channels can still cause a lot of unreliable commands)

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



## **Photon Unity Networking**

List of all members

v1.81

Main Page	e Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	/ Class Members		
				Public Member Functions		
PhotonPlayer Class				Static Public Member Functions		
Referen	_				Public Attributes	Properties

**Public API** 

Summarizes a "player" within a room, identified (in that room) by actorID. More...

Inherits IComparable< PhotonPlayer >, IComparable< int >, IEquatable< PhotonPlayer >, and IEquatable< int >.

## **Public Member Functions**

PhotonPlayer (bool isLocal, int actorID, string name)

Creates a **PhotonPlayer** instance. More...

override bool Equals (object p)

Makes PhotonPlayer comparable More...

override int GetHashCode ()

SetCustomProperties (Hashtable propertiesToSet, void Hashtable expectedValues=null, bool

webForward=false)

Updates the this player's Custom Properties with new/updated key-values. More...

PhotonPlayer Get (int id)

PhotonPlayer GetNext ()

PhotonPlayer GetNextFor (PhotonPlayer currentPlayer)

PhotonPlayer GetNextFor (int currentPlayerId)

int CompareTo (PhotonPlayer other)

int CompareTo (int other)

bool Equals (PhotonPlayer other)

bool **Equals** (int other)

override string ToString ()

Brief summary string of the **PhotonPlayer**. Includes name or player.ID and if it's the Master Client.

More...

## string ToStringFull ()

String summary of the **PhotonPlayer**: player.ID, name and all custom properties of this user. <u>More...</u>

## Static Public Member Functions

static **PhotonPlayer** Find (int **ID**)

Try to get a specific player by id. More...

## **Public Attributes**

## readonly bool | IsLocal = false

Only one player is controlled by each client. Others are not local. More...

## object TagObject

Can be used to store a reference that's useful to know "by player". More...

## **Properties**

### int **ID** [get]

This player's actorID More...

## string NickName [get, set]

Nickname of this player. More...

## string UserId [get, set]

UserId of the player, available when the room got created with **RoomOptions.PublishUserId** = true. More...

## bool IsMasterClient [get]

True if this player is the Master Client of the current room. More...

## bool Islnactive [get, set]

Players might be inactive in a room when PlayerTTL for a room is > 0. If true, the player is not getting events from this room (now) but can return later. <u>More...</u>

## Hashtable CustomProperties [get, set]

Read-only cache for custom properties of player. Set via **PhotonPlayer.SetCustomProperties**. <u>More...</u>

## Hashtable AllProperties [get]

Creates a Hashtable with all properties (custom and "well known" ones). <u>More...</u>

string name [get, set]

string userId [get, set]

bool isLocal [get]

bool isMasterClient [get]

bool islnactive [get, set]

Hashtable customProperties [get, set]

Hashtable allProperties [get]

## **Detailed Description**

Summarizes a "player" within a room, identified (in that room) by actorID.

Each player has an actorId (or ID), valid for that room. It's -1 until it's assigned by server. Each client can set it's player's custom properties with SetCustomProperties, even before being in a room. They are synced when joining a room.

## Constructor & Destructor Documentation

```
PhotonPlayer.PhotonPlayer ( bool isLocal, int actorID, string name )
```

Creates a **PhotonPlayer** instance.

#### **Parameters**

isLocal If this is the local peer's player (or a remote one).
 actorID ID or ActorNumber of this player in the current room (a shortcut to identify each player in room)
 name Name of the player (a "well known property").

## **Member Function Documentation**

int PhotonPlayer.CompareTo (PhotonPlayer other)

int PhotonPlayer.CompareTo (int other)

override bool PhotonPlayer.Equals (object p)

Makes **PhotonPlayer** comparable

bool PhotonPlayer.Equals (PhotonPlayer other)

bool PhotonPlayer.Equals (int other)

## static PhotonPlayer PhotonPlayer.Find (int ID)

stati<u>c</u>

Try to get a specific player by id.

**Parameters** 

**ID** ActorID

Returns

The player with matching actorID or null, if the actorID is not in use.

PhotonPlayer PhotonPlayer.Get (int id)

override int PhotonPlayer.GetHashCode ( )

## PhotonPlayer PhotonPlayer.GetNext ( )

```
PhotonPlayer.GetNextFor
```

(PhotonPlayer currentPlayer)

PhotonPlayer PhotonPlayer.GetNextFor (int currentPlayerId)

```
void
PhotonPlayer.SetCustomProperties (Hashtable propertiesToSet,
Hashtable expectedValues =
bool webForward = fal
)
```

Updates the this player's Custom Properties with new/updated key-valu

Custom Properties are a key-value set (Hashtable) which is available to players in a room. They can relate to the room or individual players and useful when only the current value of something is of interest. For exam The map of a room. All keys must be strings.

The **Room** and the **PhotonPlayer** class both have SetCustomPropertic methods. Also, both classes offer access to current key-values by: customProperties.

Always use SetCustomProperties to change values. To reduce network traffic, set only values that actually changed. New properties are added existing values are updated. Other values will not be changed, so only provide values that changed or are new.

To delete a named (custom) property of this room, use null as value.

Locally, SetCustomProperties will update it's cache without delay. Other clients are updated through **Photon** (the server) with a fitting operation.

## **Check and Swap**

SetCustomProperties have the option to do a server-side Check-And-S (CAS): Values only get updated if the expected values are correct. The

expectedValues can be different key/values than the propertiesToSet. S you can check some key and set another key's value (if the check succeeds).

If the client's knowledge of properties is wrong or outdated, it can't set values with CAS. This can be useful to keep players from concurrently setting values. For example: If all players try to pickup some card or iter only one should get it. With CAS, only the first SetProperties gets exect server-side and any other (sent at the same time) fails.

The server will broadcast successfully changed values and the local "cache" of customProperties only gets updated after a roundtrip (if anyt changed).

You can do a "webForward": **Photon** will send the changed properties t WebHook defined for your application.

#### OfflineMode

While **PhotonNetwork.offlineMode** is true, the expectedValues and webForward parameters are ignored. In OfflineMode, the local customProperties values are immediately updated (without the roundtri)

#### **Parameters**

**propertiesToSet** The new properties to be set.

expectedValues At least one property key/value set to check serv

side. Key and value must be correct. Ignored in

OfflineMode.

webForward Set to true, to forward the set properties to a

WebHook, defined for this app (in Dashboard).

Ignored in OfflineMode.

## override string PhotonPlayer.ToString ( )

Brief summary string of the **PhotonPlayer**. Includes name or player.ID and if it's the Master Client.

## string PhotonPlayer.ToStringFull ( )

String summary of the **PhotonPlayer**: player.ID, name and all custom properties of this user.

Use with care and not every frame! Converts the customProperties to a String on every single call.

# Member Data Documentation

## readonly bool PhotonPlayer.IsLocal = false

Only one player is controlled by each client. Others are not local.

## object PhotonPlayer.TagObject

Can be used to store a reference that's useful to know "by player".

Example: Set a player's character as Tag by assigning the GameObject on Instantiate.

# **Property Documentation**

#### Hashtable PhotonPlayer.AllProperties

get

Creates a Hashtable with all properties (custom and "well known" ones).

If used more often, this should be cached.

#### Hashtable PhotonPlayer.allProperties



#### Hashtable PhotonPlayer.CustomProperties



Read-only cache for custom properties of player. Set via PhotonPlayer.SetCustomProperties.

Don't modify the content of this Hashtable. Use SetCustomProperties and the properties of this class to modify values. When you use those, the client will sync values with the server.

## **SetCustomProperties**

## Hashtable PhotonPlayer.customProperties





## int PhotonPlayer.ID



This player's actorID

### bool PhotonPlayer.IsInactive





Players might be inactive in a room when PlayerTTL for a room is > 0. If true, the player is not getting events from this room (now) but can return later.

#### bool PhotonPlayer.isInactive



#### bool PhotonPlayer.isLocal



#### bool PhotonPlayer.IsMasterClient



True if this player is the Master Client of the current room.

See also: PhotonNetwork.masterClient.

#### bool PhotonPlayer.isMasterClient



#### string PhotonPlayer.name





#### string PhotonPlayer.NickName





Nickname of this player.

Set the **PhotonNetwork.playerName** to make the name synchronized in a room.

### string PhotonPlayer.UserId





UserId of the player, available when the room got created with RoomOptions.PublishUserId = true.

Useful for **PhotonNetwork.FindFriends** and blocking slots in a room for expected players (e.g. in PhotonNetwork.CreateRoom).

# string PhotonPlayer.userId



Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page Related		Pages Modul		lles	Classes	Files
Class List	Class Index	Class Hi	lierarchy Class Members		ss Members	
PhotonView Class Reference Public API				Public Member Functions   Static Public Member Functions   Public Attributes   Properties   List of all members		

PUN's NetworkView replacement class for networking. Use it like a NetworkView. <u>More...</u>

Inherits Photon.MonoBehaviour.

## **Public Member Functions**

### void RequestOwnership ()

Depending on the **PhotonView**'s ownershipTransfer setting, any client can request to become owner of the **PhotonView**. More...

## void TransferOwnership (PhotonPlayer newOwner)

Transfers the ownership of this **PhotonView** (and GameObject) to another player. More...

#### void **TransferOwnership** (int newOwnerld)

Transfers the ownership of this **PhotonView** (and GameObject) to another player. <u>More...</u>

# void OnMasterClientSwitched (PhotonPlayer newMasterClient)

Check ownerId assignment for sceneObjects to keep being owned by the MasterClient. More...

void SerializeView (PhotonStream stream, PhotonMessageInfo info)

void **DeserializeView (PhotonStream** stream, **PhotonMessageInfo** info)

### void RefreshRpcMonoBehaviourCache ()

Can be used to refesh the list of MonoBehaviours on this GameObject while

**PhotonNetwork.UseRpcMonoBehaviourCache** is true. More...

# void RPC (string methodName, PhotonTargets target, params object[] parameters)

Call a RPC method of this GameObject on remote clients of this room (or on all, inclunding this client). More...

void

**RpcSecure** (string methodName, **PhotonTargets** target, bool encrypt, params object[] parameters) Call a RPC method of this GameObject on remote clients of this room (or on all, inclunding this client). More...

void

RPC (string methodName, PhotonPlayer targetPlayer, params object[] parameters) Call a RPC method of this GameObject on remote clients of this room (or on all, inclunding this client). More...

RpcSecure (string methodName, PhotonPlayer void targetPlayer, bool encrypt, params object[] parameters)

> Call a RPC method of this GameObject on remote clients of this room (or on all, inclunding this client). More...

override string ToString ()

# Static Public Member Functions

static PhotonView Get (Component component)

static PhotonView Get (GameObject gameObj)

static PhotonView Find (int viewID)

## Public Attributes

Public Attributes	
int	ownerld
int	group = 0
bool	OwnerShipWasTransfered Flag to check if ownership of this photonView was set during the lifecycle. Used for checking when joining late if event with mismatched owner and sender needs addressing. More
int	prefixBackup = -1
ViewSynchronization	synchronization
OnSerializeTransform	onSerializeTransformOption = OnSerializeTransform.PositionAndRotation
OnSerializeRigidBody	onSerializeRigidBodyOption = OnSerializeRigidBody.All
OwnershipOption	<pre>ownershipTransfer = OwnershipOption.Fixed Defines if ownership of this PhotonView is fixed, can be requested or simply taken. More</pre>
List< Component >	ObservedComponents
·	·
int	instantiationId
int	currentMasterID = -1 The current master ID so that we can compare when we receive OnMasterClientSwitched() callback It's

public so that we can check it during ownerld assignments in networkPeer script TODO: Maybe we can have the networkPeer always aware of the previous MasterClient? More...

# **Properties**

int prefix [get, set]

object[] instantiationData [get, set]

This is the instantiationData that was passed when calling **PhotonNetwork.Instantiate**\* (if that was used to spawn this prefab) <u>More...</u>

int viewID [get, set]

The ID of the **PhotonView**. Identifies it in a networked game (per room). More...

bool isSceneView [get]

True if the **PhotonView** was loaded with the scene (game object) or instantiated with InstantiateSceneObject. <u>More...</u>

PhotonPlayer owner [get]

The owner of a **PhotonView** is the player who created the GameObject with that view. Objects in the scene don't have an owner. More...

int OwnerActorNr [get]

bool isOwnerActive [get]

int CreatorActorNr [get]

bool isMine [get]

True if the **PhotonView** is "mine" and can be controlled by this client. <u>More...</u>

▶ Properties inherited from Photon.MonoBehaviour

PhotonView photonView [get]

A cached reference to a **PhotonView** on this GameObject. <u>More...</u>

## new PhotonView networkView [get]

This property is only here to notify developers when they use the outdated value. More...

# **Detailed Description**

PUN's NetworkView replacement class for networking. Use it like a NetworkView.

## **Member Function Documentation**

void PhotonView.DeserializeView ( PhotonStream stream, PhotonMessageInfo info )

#### static PhotonView PhotonView.Find (int viewID)



static PhotonView PhotonView.Get

(Component component)



static PhotonView PhotonView.Get

(GameObject gameObj)



#### void

PhotonView.OnMasterClientSwitched (PhotonPlayer newMasterC

Check ownerId assignment for sceneObjects to keep being owned by tl MasterClient.

#### **Parameters**

newMasterClient New master client.

## void PhotonView.RefreshRpcMonoBehaviourCache ( )

Can be used to refesh the list of MonoBehaviours on this GameObject while **PhotonNetwork.UseRpcMonoBehaviourCache** is true.

Set **PhotonNetwork.UseRpcMonoBehaviourCache** to true to enable the caching. Uses this.GetComponents<MonoBehaviour>()

to get a list of MonoBehaviours to call RPCs on (potentially).

While **PhotonNetwork.UseRpcMonoBehaviourCache** is false, this method has no effect, because the list is refreshed when a RPC gets called.

#### void PhotonView.RequestOwnership ()

Depending on the **PhotonView**'s ownershipTransfer setting, any client can request to become owner of the **PhotonView**.

Requesting ownership can give you control over a **PhotonView**, if the ownershipTransfer setting allows that. The current owner might have to implement **IPunCallbacks.OnOwnershipRequest** to react to the ownership request.

The owner/controller of a **PhotonView** is also the client which sends position updates of the GameObject.

Call a RPC method of this GameObject on remote clients of this room (or on all, inclunding this client).

Remote Procedure Calls are an essential tool in making multiplayer games with PUN. It enables you to make every client in a room call a specific method.

RPC calls can target "All" or the "Others". Usually, the target "All" gets executed locally immediately after sending the RPC. The "\*ViaServer" options send the RPC to the server and execute it on this client when it's sent back. Of course, calls are affected by this client's lag and that of remote clients.

Each call automatically is routed to the same **PhotonView** (and GameObject) that was used on the originating client.

See: Remote Procedure Calls.

#### **Parameters**

methodName The name of a fitting method that was has the

RPC attribute.

target The group of targets and the way the RPC gets

sent.

**parameters** The parameters that the RPC method has (must

fit this call!).

Call a RPC method of this GameObject on remote clients of this room (or on all, inclunding this client).

Remote Procedure Calls are an essential tool in making multiplayer games with PUN. It enables you to make every client in a room call a specific method.

This method allows you to make an RPC calls on a specific player's client. Of course, calls are affected by this client's lag and that of remote clients.

Each call automatically is routed to the same **PhotonView** (and GameObject) that was used on the originating client.

See: Remote Procedure Calls.

#### **Parameters**

**methodName** The name of a fitting method that was has the RPC attribute.

targetPlayer The group of targets and the way the RPC gets

sent.

parameters

The parameters that the RPC method has (must fit this call!).

```
void PhotonView.RpcSecure ( string methodName, PhotonTargets target, bool encrypt, params object[] parameters )
```

Call a RPC method of this GameObject on remote clients of this room (or on all, inclunding this client).

Remote Procedure Calls are an essential tool in making multiplayer games with PUN. It enables you to make every client in a room call a specific method.

RPC calls can target "All" or the "Others". Usually, the target "All" gets executed locally immediately after sending the RPC. The "\*ViaServer" options send the RPC to the server and execute it on this client when it's sent back. Of course, calls are affected by this client's lag and that of remote clients.

Each call automatically is routed to the same **PhotonView** (and GameObject) that was used on the originating client.

See: Remote Procedure Calls.

param name="methodName">The name of a fitting method that was has the RPC attribute.

param name="target">The group of targets and the way the RPC gets sent.

param name="encrypt">

param name="parameters">The parameters that the RPC method has (must fit this call!).

```
void PhotonView.RpcSecure ( string methodName, PhotonPlayer targetPlayer, bool encrypt, params object[] parameters
```

Call a RPC method of this GameObject on remote clients of this room (or on all, inclunding this client).

Remote Procedure Calls are an essential tool in making multiplayer games with PUN. It enables you to make every client in a room call a specific method.

This method allows you to make an RPC calls on a specific player's client. Of course, calls are affected by this client's lag and that of remote clients.

Each call automatically is routed to the same **PhotonView** (and GameObject) that was used on the originating client.

See: Remote Procedure Calls.

param name="methodName">The name of a fitting method that was has the RPC attribute.

param name="targetPlayer">The group of targets and the way the RPC gets sent.

param name="encrypt">

param name="parameters">The parameters that the RPC method has (must fit this call!).

```
void PhotonView.SerializeView (PhotonStream stream, PhotonMessageInfo info
```

## override string PhotonView.ToString ()

# void PhotonView.TransferOwnership (PhotonPlayer newOwner)

Transfers the ownership of this **PhotonView** (and GameObject) to another player.

The owner/controller of a **PhotonView** is also the client which sends position updates of the GameObject.

## void PhotonView.TransferOwnership (int newOwnerId)

Transfers the ownership of this **PhotonView** (and GameObject) to another player.

The owner/controller of a **PhotonView** is also the client which sends position updates of the GameObject.

#### Member Data Documentation

#### int PhotonView.currentMasterID = -1

The current master ID so that we can compare when we receive **OnMasterClientSwitched()** callback It's public so that we can check it during ownerld assignments in networkPeer script TODO: Maybe we can have the networkPeer always aware of the previous MasterClient?

int PhotonView.group = 0

int PhotonView.instantiationId

**List<Component> PhotonView.ObservedComponents** 

OnSerializeRigidBody PhotonView.onSerializeRigidBodyOption = OnSerializeRigidBody.All

OnSerializeTransform PhotonView.onSerializeTransformOption = OnSerializeTransform.PositionAndRotation

#### int PhotonView.ownerId

OwnershipOption PhotonView.ownershipTransfer = OwnershipOption.Fixed

Defines if ownership of this **PhotonView** is fixed, can be requested or simply taken.

Note that you can't edit this value at runtime. The options are described in enum OwnershipOption. The current owner has to implement IPunCallbacks.OnOwnershipRequest to react to the ownership request.

#### bool PhotonView.OwnerShipWasTransfered

Flag to check if ownership of this photonView was set during the lifecycle. Used for checking when joining late if event with mismatched owner and sender needs addressing.

true if owner ship was transfered; otherwise, false.

### int PhotonView.prefixBackup = -1

ViewSynchronization PhotonView.synchronization

# **Property Documentation**

#### int PhotonView.CreatorActorNr

get

#### object [] PhotonView.instantiationData



This is the instantiationData that was passed when calling **PhotonNetwork.Instantiate\*** (if that was used to spawn this prefab)

#### bool PhotonView.isMine



True if the **PhotonView** is "mine" and can be controlled by this client.

PUN has an ownership concept that defines who can control and destroy each **PhotonView**. True in case the owner matches the local **PhotonPlayer**. True if this is a scene photonview on the Master client.

#### bool PhotonView.isOwnerActive



#### bool PhotonView.isSceneView



True if the **PhotonView** was loaded with the scene (game object) or instantiated with InstantiateSceneObject.

Scene objects are not owned by a particular player but belong to the scene. Thus they don't get destroyed when their creator leaves the game and the current Master Client can control them (whoever that is). The ownerld is 0 (player IDs are 1 and up).

#### PhotonPlayer PhotonView.owner

get

The owner of a **PhotonView** is the player who created the GameObject with that view. Objects in the scene don't have an owner.

The owner/controller of a **PhotonView** is also the client which sends position updates of the GameObject.

Ownership can be transferred to another player with PhotonView.TransferOwnership or any player can request ownership by calling the **PhotonView**'s RequestOwnership method. The current owner has to implement

IPunCallbacks.OnOwnershipRequest to react to the ownership request.

#### int PhotonView.OwnerActorNr



#### int PhotonView.prefix





#### int PhotonView.viewID





The ID of the **PhotonView**. Identifies it in a networked game (per room).

See: Network Instantiation

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page		Related Pages		Modules		Classes	Files	
Class List	Class	Index	Class Hi	Hierarchy Class Me		ss Members		
					Public Member Functions   Properties			
Room Class Reference Public API					<u>List of all members</u>			

This class resembles a room that PUN joins (or joined). The properties are settable as opposed to those of a **RoomInfo** and you can close or hide "your" room. More...

Inherits RoomInfo.

### **Public Member Functions**

# **SetCustomProperties (Hashtable** propertiesToSet, void **Hashtable** expectedValues=null, bool

webForward=false)

Updates the current room's Custom Properties with new/updated key-values. <u>More...</u>

# void SetPropertiesListedInLobby (string[] propsListedInLobby)

Enables you to define the properties available in the lobby if not all properties are needed to pick a room. More...

#### void ClearExpectedUsers ()

Attempts to remove all current expected users from the server's Slot Reservation list. More...

## override string ToString ()

Returns a summary of this **Room** instance as string. More...

### new string ToStringFull ()

Returns a summary of this **Room** instance as longer string, including Custom Properties. <u>More...</u>

#### ▶ Public Member Functions inherited from RoomInfo

override bool Equals (object other)

Makes RoomInfo comparable (by name). More...

## override int GetHashCode ()

Accompanies Equals, using the name's HashCode as return. More...

#### override string ToString ()

Simple printingin method. More...

string **ToStringFull ()**Simple printingin method. <u>More...</u>

# **Properties**

#### new string Name [get, set]

The name of a room. Unique identifier (per Loadbalancing group) for a room/match. More...

#### new bool IsOpen [get, set]

Defines if the room can be joined. This does not affect listing in a lobby but joining the room will fail if not open. If not open, the room is excluded from random matchmaking. Due to racing conditions, found matches might become closed before they are joined. Simply reconnect to master and find another. Use property "visible" to not list the room. More...

#### new bool IsVisible [get, set]

Defines if the room is listed in its lobby. Rooms can be created invisible, or changed to invisible. To change if a room can be joined, use property: open. More...

### string[ PropertiesListedInLobby [get, set]

A list of custom properties that should be forwarded to the lobby and listed there. <u>More...</u>

## bool AutoCleanUp [get]

Gets if this room uses autoCleanUp to remove all (buffered) RPCs and instantiated GameObjects when a player leaves. <u>More...</u>

## new int MaxPlayers [get, set]

Sets a limit of players to this room. This property is shown in lobby, too. If the room is full (players count == maxplayers), joining this room will fail. <u>More...</u>

#### new int PlayerCount [get]

Count of players in this room. More...

#### string[] **ExpectedUsers** [get]

List of users who are expected to join this room. In matchmaking, **Photon** blocks a slot for each of these UserIDs out of the MaxPlayers. More...

new string name [get, set]

new bool open [get, set]

new bool visible [get, set]

string[] propertiesListedInLobby [get, set]

bool autoCleanUp [get]

new int maxPlayers [get, set]

new int **playerCount** [get]

string[] expectedUsers [get]

### ▶ Properties inherited from RoomInfo

#### bool removedFromList [get, set]

Used internally in lobby, to mark rooms that are no longer listed. More...

#### Hashtable CustomProperties [get]

Read-only "cache" of custom properties of a room. Set via **Room.SetCustomProperties** (not available for **RoomInfo** class!). <u>More...</u>

#### string Name [get]

The name of a room. Unique identifier (per Loadbalancing group) for a room/match. More...

#### int PlayerCount [get, set]

Only used internally in lobby, to display number of players in room (while you're not in). More...

#### bool IsLocalClientInside [get, set]

State if the local client is already in the game or still going to join it on gameserver (in lobby always false). More...

### byte MaxPlayers [get]

Sets a limit of players to this room. This property is shown in lobby, too. If the room is full (players count == maxplayers), joining this room will fail. More...

#### bool IsOpen [get]

Defines if the room can be joined. This does not affect listing in a lobby but joining the room will fail if not open. If not open, the room is excluded from random matchmaking. Due to racing conditions, found matches might become closed before they are joined. Simply reconnect to master and find another. Use property "IsVisible" to not list the room. More...

#### bool IsVisible [get]

Defines if the room is listed in its lobby. Rooms can be created invisible, or changed to invisible. To change if a room can be joined, use property: open. More...

### Hashtable customProperties [get]

string name [get]

int playerCount [get, set]

bool isLocalClientInside [get, set]

byte maxPlayers [get]	
bool open [get]	
bool <b>visible</b> [get]	

## **Additional Inherited Members**

#### → Protected Attributes inherited from RoomInfo

### byte maxPlayersField = 0

Backing field for property. More...

#### string[] expectedUsersField

Backing field for property. More...

#### bool **openField** = true

Backing field for property. More...

#### bool visibleField = true

Backing field for property. More...

# bool autoCleanUpField =

## PhotonNetwork.autoCleanUpPlayerObjects

Backing field for property. False unless the GameProperty is set to true (else it's not sent). <u>More...</u>

## string nameField

Backing field for property. More...

# **Detailed Description**

This class resembles a room that PUN joins (or joined). The properties are settable as opposed to those of a **RoomInfo** and you can close or hide "your" room.

### Member Function Documentation

#### void Room.ClearExpectedUsers ( )

Attempts to remove all current expected users from the server's Slot Reservation list.

Note that this operation can conflict with new/other users joining. They might be adding users to the list of expected users before or after this client called ClearExpectedUsers.

This room's expectedUsers value will update, when the server sends a successful update.

Internals: This methods wraps up setting the ExpectedUsers property of a room.

Updates the current room's Custom Properties with new/updated key-values.

Custom Properties are a key-value set (Hashtable) which is available to all players in a room. They can relate to the room or individual players and are useful when only the current value of something is of interest. For example: The map of a room. All keys must be strings.

The **Room** and the **PhotonPlayer** class both have SetCustomProperties methods. Also, both classes offer access to

current key-values by: customProperties.

Always use SetCustomProperties to change values. To reduce network traffic, set only values that actually changed. New properties are added, existing values are updated. Other values will not be changed, so only provide values that changed or are new.

To delete a named (custom) property of this room, use null as value.

Locally, SetCustomProperties will update it's cache without delay. Other clients are updated through **Photon** (the server) with a fitting operation.

#### **Check and Swap**

SetCustomProperties have the option to do a server-side Check-And-Swap (CAS): Values only get updated if the expected values are correct. The expectedValues can be different key/values than the propertiesToSet. So you can check some key and set another key's value (if the check succeeds).

If the client's knowledge of properties is wrong or outdated, it can't set values with CAS. This can be useful to keep players from concurrently setting values. For example: If all players try to pickup some card or item, only one should get it. With CAS, only the first SetProperties gets executed server-side and any other (sent at the same time) fails.

The server will broadcast successfully changed values and the local "cache" of customProperties only gets updated after a roundtrip (if anything changed).

You can do a "webForward": **Photon** will send the changed properties to a WebHook defined for your application.

#### OfflineMode

While **PhotonNetwork.offlineMode** is true, the expectedValues and webForward parameters are ignored. In OfflineMode, the local customProperties values are immediately updated (without the roundtrip).

#### **Parameters**

**propertiesToSet** The new properties to be set.

**expectedValues** At least one property key/value set to check

server-side. Key and value must be correct.

Ignored in OfflineMode.

webForward Set to true, to forward the set properties to a

WebHook, defined for this app (in Dashboard). Ignored in OfflineMode.

#### void

Room.SetPropertiesListedInLobby (string[] propsListedInLobby)

Enables you to define the properties available in the lobby if not all properties are needed to pick a room.

It makes sense to limit the amount of properties sent to users in the lobby as this improves speed and stability.

#### **Parameters**

**propsListedInLobby** An array of custom room property names to forward to the lobby.

### override string Room.ToString ()

Returns a summary of this **Room** instance as string.

#### Returns

Summary of this Room instance.

#### new string Room.ToStringFull()

Returns a summary of this **Room** instance as longer string, including Custom Properties.

#### Returns

Summary of this **Room** instance.

# **Property Documentation**

#### bool Room.AutoCleanUp

get

Gets if this room uses autoCleanUp to remove all (buffered) RPCs and instantiated GameObjects when a player leaves.

#### bool Room.autoCleanUp



#### string [] Room.ExpectedUsers



List of users who are expected to join this room. In matchmaking, **Photon** blocks a slot for each of these UserIDs out of the MaxPlayers.

The corresponding feature in **Photon** is called "Slot Reservation" and can be found in the doc pages. Define expected players in the PhotonNetwork methods: CreateRoom, JoinRoom and JoinOrCreateRoom.

#### string [] Room.expectedUsers



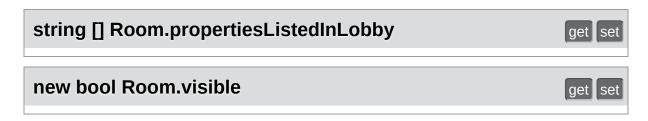
#### new bool Room.IsOpen



Defines if the room can be joined. This does not affect listing in a lobby but joining the room will fail if not open. If not open, the room is excluded from random matchmaking. Due to racing conditions, found matches might become closed before they are joined. Simply re-connect to master and find another. Use property "visible" to not list the room.

# new bool Room.IsVisible get set Defines if the room is listed in its lobby. Rooms can be created invisible, or changed to invisible. To change if a room can be joined, use property: open. new int Room.MaxPlayers get set Sets a limit of players to this room. This property is shown in lobby, too. If the room is full (players count == maxplayers), joining this room will fail. new int Room.maxPlayers get set new string Room.Name get set The name of a room. Unique identifier (per Loadbalancing group) for a room/match. new string Room.name get set new bool Room.open get set new int Room.PlayerCount get Count of players in this room. new int Room.playerCount get string [] Room.PropertiesListedInLobby get set

A list of custom properties that should be forwarded to the lobby and listed there.



Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# **Photon Unity Networking**

v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files	
Class List	Class Index	Class Hi	erarchy	Clas	ss Members		

# RoomInfo Class Reference

**Public API** 

Public Member Functions | Protected Attributes | Properties | List of all members

A simplified room with just the info required to list and join, used for the room listing in the lobby. The properties are not settable (open, MaxPlayers, etc). More...

Inherited by Room.

## **Public Member Functions**

override bool Equals (object other)

Makes RoomInfo comparable (by name). More...

override int GetHashCode ()

Accompanies Equals, using the name's HashCode

as return. More...

override string ToString ()

Simple printingin method. More...

string ToStringFull ()

Simple printingin method. More...

#### **Protected Attributes**

#### byte maxPlayersField = 0

Backing field for property. More...

#### string[] expectedUsersField

Backing field for property. More...

#### bool **openField** = true

Backing field for property. More...

#### bool visibleField = true

Backing field for property. More...

# bool autoCleanUpField =

PhotonNetwork.autoCleanUpPlayerObjects

Backing field for property. False unless the GameProperty is set to true (else it's not sent). More...

#### string nameField

Backing field for property. More...

## **Properties**

#### bool removedFromList [get, set]

Used internally in lobby, to mark rooms that are no longer listed. More...

#### Hashtable CustomProperties [get]

Read-only "cache" of custom properties of a room. Set via **Room.SetCustomProperties** (not available for **RoomInfo** class!). <u>More...</u>

#### string Name [get]

The name of a room. Unique identifier (per Loadbalancing group) for a room/match. More...

#### int PlayerCount [get, set]

Only used internally in lobby, to display number of players in room (while you're not in). More...

#### bool IsLocalClientInside [get, set]

State if the local client is already in the game or still going to join it on gameserver (in lobby always false). More...

#### byte MaxPlayers [get]

Sets a limit of players to this room. This property is shown in lobby, too. If the room is full (players count == maxplayers), joining this room will fail. <u>More...</u>

#### bool IsOpen [get]

Defines if the room can be joined. This does not affect listing in a lobby but joining the room will fail if not open. If not open, the room is excluded from random matchmaking. Due to racing conditions, found matches might become closed before they are joined. Simply reconnect to master and find another. Use property

"IsVisible" to not list the room. More...

#### bool IsVisible [get]

Defines if the room is listed in its lobby. Rooms can be created invisible, or changed to invisible. To change if a room can be joined, use property: open. More...

### Hashtable customProperties [get]

string name [get]

int playerCount [get, set]

bool isLocalClientInside [get, set]

byte maxPlayers [get]

bool open [get]

bool visible [get]

# **Detailed Description**

A simplified room with just the info required to list and join, used for the room listing in the lobby. The properties are not settable (open, MaxPlayers, etc).

This class resembles info about available rooms, as sent by the Master server's lobby. Consider all values as readonly. None are synced (only updated by events by server).

### **Member Function Documentation**

#### override bool RoomInfo.Equals (object other)

Makes RoomInfo comparable (by name).

#### override int RoomInfo.GetHashCode ( )

Accompanies Equals, using the name's HashCode as return.

#### **Returns**

#### override string RoomInfo.ToString ()

Simple printingin method.

#### Returns

Summary of this **RoomInfo** instance.

#### string RoomInfo.ToStringFull ( )

Simple printingin method.

#### Returns

Summary of this **RoomInfo** instance.

#### Member Data Documentation

# bool RoomInfo.autoCleanUpField = PhotonNetwork.autoCleanUpPlayerObjects

protected

Backing field for property. False unless the GameProperty is set to true (else it's not sent).

#### string [] RoomInfo.expectedUsersField

protected

Backing field for property.

#### byte RoomInfo.maxPlayersField = 0

protected

Backing field for property.

#### string RoomInfo.nameField

protected

Backing field for property.

#### bool RoomInfo.openField = true

protected

Backing field for property.

#### bool RoomInfo.visibleField = true

protected

Backing field for property.

# **Property Documentation**

#### Hashtable RoomInfo.CustomProperties

get

Read-only "cache" of custom properties of a room. Set via Room.SetCustomProperties (not available for RoomInfo class!).

All keys are string-typed and the values depend on the game/application.

#### Room.SetCustomProperties

#### Hashtable RoomInfo.customProperties

get

#### bool RoomInfo.IsLocalClientInside





State if the local client is already in the game or still going to join it on gameserver (in lobby always false).

#### bool RoomInfo.isLocalClientInside





#### bool RoomInfo.IsOpen

get

Defines if the room can be joined. This does not affect listing in a lobby but joining the room will fail if not open. If not open, the room is excluded from random matchmaking. Due to racing conditions. found matches might become closed before they are joined. Simply re-connect to master and find another. Use property "IsVisible" to not list the room.

As part of **RoomInfo** this can't be set. As part of a **Room** (which the player joined), the setter will update the server and all clients.

#### bool RoomInfo.IsVisible

get

Defines if the room is listed in its lobby. Rooms can be created invisible, or changed to invisible. To change if a room can be joined, use property: open.

As part of RoomInfo this can't be set. As part of a Room (which the player joined), the setter will update the server and all clients.

#### byte RoomInfo.MaxPlayers



Sets a limit of players to this room. This property is shown in lobby, too. If the room is full (players count == maxplayers), joining this room will fail.

As part of RoomInfo this can't be set. As part of a Room (which the player joined), the setter will update the server and all clients.

#### byte RoomInfo.maxPlayers



#### string RoomInfo.Name



The name of a room. Unique identifier (per Loadbalancing group) for a room/match.

#### string RoomInfo.name



#### bool RoomInfo.open



#### int RoomInfo.PlayerCount





Only used internally in lobby, to display number of players in room (while you're not in).







#### bool RoomInfo.removedFromList



Used internally in lobby, to mark rooms that are no longer listed.

#### bool RoomInfo.visible



Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main PageRelated PagesModulesClassesFilesOptional Gui Elements

Useful GUI elements for PUN. More...

# Classes

#### class PhotonLagSimulationGui

This MonoBehaviour is a basic GUI for the **Photon** client's network-simulation feature. It can modify lag (fixed delay), jitter (random lag) and packet loss. <u>More...</u>

#### class PhotonStatsGui

Basic GUI to show traffic and health statistics of the connection to **Photon**, toggled by shift+tab. <u>More...</u>

# **Detailed Description**

Useful GUI elements for PUN.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page Related P		d Pages	Modu	les	Classes	Files		
Class List	Class Index	Class H	ierarchy	Clas	ss Members			

<u>Public Member Functions</u> | <u>Public Attributes</u> | <u>Properties</u> | <u>List of all members</u>

# PhotonLagSimulationGui Class Reference

**Optional Gui Elements** 

This MonoBehaviour is a basic GUI for the **Photon** client's network-simulation feature. It can modify lag (fixed delay), jitter (random lag) and packet loss. More...

Inherits MonoBehaviour.

# **Public Member Functions**

void Start ()

void OnGUI ()

# **Public Attributes**

Rect **WindowRect** = new Rect(0, 100, 120, 100)

Positioning rect for window. More...

int Windowld = 101

Unity GUI Window ID (must be unique or will cause issues). More...

bool Visible = true

Shows or hides GUI (does not affect settings). More...

# **Properties**

PhotonPeer Peer [get, set]

The peer currently in use (to set the network simulation). More...

# **Detailed Description**

This MonoBehaviour is a basic GUI for the **Photon** client's network-simulation feature. It can modify lag (fixed delay), jitter (random lag) and packet loss.

# **Member Function Documentation**

void PhotonLagSimulationGui.OnGUI ( )

void PhotonLagSimulationGui.Start ( )

## Member Data Documentation

#### bool PhotonLagSimulationGui.Visible = true

Shows or hides GUI (does not affect settings).

#### int PhotonLagSimulationGui.Windowld = 101

Unity GUI Window ID (must be unique or will cause issues).

# Rect PhotonLagSimulationGui.WindowRect = new Rect(0, 100, 120, 100)

Positioning rect for window.

# **Property Documentation**

### PhotonPeer PhotonLagSimulationGui.Peer

get set

The peer currently in use (to set the network simulation).

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page	Main Page Related F		Modu	ıles	Classes	Files
Class List	Class Inc	lex Class	<b>Hierarchy</b>	Clas	ss Members	

<u>Public Member Functions</u> | <u>Public Attributes</u> | <u>List of all members</u>

# PhotonStatsGui Class Reference

**Optional Gui Elements** 

Basic GUI to show traffic and health statistics of the connection to **Photon**, toggled by shift+tab. <u>More...</u>

Inherits MonoBehaviour.

# **Public Member Functions**

## void Start ()

void Update ()
Checks for shift+tab input combination (to toggle statsOn). More...

### void OnGUI ()

void TrafficStatsWindow (int windowID)

#### **Public Attributes**

#### bool statsWindowOn = true

Shows or hides GUI (does not affect if stats are collected). More...

#### bool statsOn = true

Option to turn collecting stats on or off (used in **Update()**). More...

#### bool healthStatsVisible

Shows additional "health" values of connection. More...

#### bool trafficStatsOn

Shows additional "lower level" traffic stats. More...

#### bool buttonsOn

Show buttons to control stats and reset them. More...

#### Rect **statsRect** = new Rect(0, 100, 200, 50)

Positioning rect for window. More...

#### int Windowld = 100

Unity GUI Window ID (must be unique or will cause issues). More...

# **Detailed Description**

Basic GUI to show traffic and health statistics of the connection to **Photon**, toggled by shift+tab.

The shown health values can help identify problems with connection losses or performance. Example: If the time delta between two consecutive SendOutgoingCommands calls is a second or more, chances rise for a disconnect being caused by this (because acknowledgements to the server need to be sent in due time).

# **Member Function Documentation**

void PhotonStatsGui.OnGUI()

void PhotonStatsGui.Start ( )

void PhotonStatsGui.TrafficStatsWindow (int windowID)

void PhotonStatsGui.Update ( )

Checks for shift+tab input combination (to toggle statsOn).

#### Member Data Documentation

#### bool PhotonStatsGui.buttonsOn

Show buttons to control stats and reset them.

#### bool PhotonStatsGui.healthStatsVisible

Shows additional "health" values of connection.

#### bool PhotonStatsGui.statsOn = true

Option to turn collecting stats on or off (used in Update()).

#### **Rect PhotonStatsGui.statsRect = new Rect(0, 100, 200, 50)**

Positioning rect for window.

#### bool PhotonStatsGui.statsWindowOn = true

Shows or hides GUI (does not affect if stats are collected).

#### bool PhotonStatsGui.trafficStatsOn

Shows additional "lower level" traffic stats.

#### int PhotonStatsGui.Windowld = 100

Unity GUI Window ID (must be unique or will cause issues).

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

ı	Иai	n P	ag	е		Related Pages						Mc	du	les	5	С	las	F	iles	
F	Package Functions																			
P	AII	•	Туре	edef	s	Enumerations				Enumerator										
a	С	d	е	f	g	h	i	m	n	0	р	q	s	t	u	w	х			

Here is a list of all namespace members with links to the namespace documentation for each member:

#### - a -

- Authenticated : ExitGames::Client::Photon::Chat
- Authenticating : ExitGames::Client::Photon::Chat

#### - C -

- ChatDisconnectCause: ExitGames::Client::Photon::Chat
- ChatState : ExitGames::Client::Photon::Chat
- ConnectedToFrontEnd : ExitGames::Client::Photon::Chat
- ConnectedToNameServer : ExitGames::Client::Photon::Chat
- ConnectingToFrontEnd : ExitGames::Client::Photon::Chat
- ConnectingToNameServer : ExitGames::Client::Photon::Chat
- Cube : ExitGames::Client::GUI
- Custom: ExitGames::Client::Photon::Chat
- CustomAuthenticationFailed : ExitGames::Client::Photon::Chat
- CustomAuthenticationType : ExitGames::Client::Photon::Chat

#### - d -

- DisconnectByServer : ExitGames::Client::Photon::Chat
- DisconnectByServerUserLimit :

- ExitGames::Client::Photon::Chat
- Disconnected : ExitGames::Client::Photon::Chat
- Disconnecting : ExitGames::Client::Photon::Chat
- DisconnectingFromFrontEnd : ExitGames::Client::Photon::Chat
- DisconnectingFromNameServer : ExitGames::Client::Photon::Chat
- e -
  - Exception : ExitGames::Client::Photon::Chat
  - ExceptionOnConnect : ExitGames::Client::Photon::Chat
- f -
  - Facebook : ExitGames::Client::Photon::Chat
- g -
  - GizmoType: ExitGames::Client::GUI
- h -
  - Hashtable: Photon
- i -
  - InvalidAuthentication : ExitGames::Client::Photon::Chat
  - InvalidRegion : ExitGames::Client::Photon::Chat
- m -
  - MaxCcuReached : ExitGames::Client::Photon::Chat
- n -

- None: ExitGames::Client::Photon::Chat
- 0 -
  - Oculus : ExitGames::Client::Photon::Chat
  - OperationNotAllowedInCurrentState : ExitGames::Client::Photon::Chat
- p -
  - PlayStation : ExitGames::Client::Photon::Chat
- q -
  - QueuedComingFromFrontEnd : ExitGames::Client::Photon::Chat
- S -
  - Sphere : ExitGames::Client::GUI
  - Steam: ExitGames::Client::Photon::Chat
- t -
  - TimeoutDisconnect : ExitGames::Client::Photon::Chat
- u -
  - Uninitialized : ExitGames::Client::Photon::Chat
- W -
  - WireCube : ExitGames::Client::GUI
  - WireSphere : ExitGames::Client::GUI

• Xbox : ExitGames::Client::Photon::Chat

Online Documentation - Dashboard - Support Forum Exit Games GmbH



v1.81

Mair	n Page	R	elated Pages	Modules	Classes	Files
Package Functions						
All	Typedefs		Enumerations	Enumerator		

• Hashtable : Photon

Online Documentation - Dashboard - Support Forum



v1.81

Mair	n Page	R	elated Pages	Modules	Classes	Files
Package Functions						
All	Typedefs	,	Enumerations	Enumerator		

• ChatDisconnectCause : ExitGames::Client::Photon::Chat

• ChatState : ExitGames::Client::Photon::Chat

• CustomAuthenticationType : ExitGames::Client::Photon::Chat

• GizmoType : ExitGames::Client::GUI

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related				d P	ag	es		M	odu	ıles		Classe	S	Files				
Package Functions																		
P	AII		Туре	ede	fs	E	Enumerations			E	Enu	mera	ator					
a	С	d	е	f	i	m	n	0	р	q	s	t	u	w	х			

#### - a -

• Authenticated : ExitGames::Client::Photon::Chat

• Authenticating : ExitGames::Client::Photon::Chat

#### - C -

- ConnectedToFrontEnd : ExitGames::Client::Photon::Chat
- ConnectedToNameServer : ExitGames::Client::Photon::Chat
- ConnectingToFrontEnd : ExitGames::Client::Photon::Chat
- ConnectingToNameServer : ExitGames::Client::Photon::Chat
- Cube : ExitGames::Client::GUI
- Custom : ExitGames::Client::Photon::Chat
- CustomAuthenticationFailed : ExitGames::Client::Photon::Chat

#### - d -

- DisconnectByServer : ExitGames::Client::Photon::Chat
- DisconnectByServerUserLimit : ExitGames::Client::Photon::Chat
- Disconnected : ExitGames::Client::Photon::Chat
- Disconnecting : ExitGames::Client::Photon::Chat
- DisconnectingFromFrontEnd : ExitGames::Client::Photon::Chat

- DisconnectingFromNameServer : ExitGames::Client::Photon::Chat - e -• Exception : ExitGames::Client::Photon::Chat • ExceptionOnConnect : ExitGames::Client::Photon::Chat - f -Facebook : ExitGames::Client::Photon::Chat - i -• InvalidAuthentication : ExitGames::Client::Photon::Chat • InvalidRegion : ExitGames::Client::Photon::Chat - m -MaxCcuReached : ExitGames::Client::Photon::Chat - n -None: ExitGames::Client::Photon::Chat - 0 -
- p -
  - PlayStation : ExitGames::Client::Photon::Chat

Oculus : ExitGames::Client::Photon::Chat

 OperationNotAllowedInCurrentState : ExitGames::Client::Photon::Chat

- q -
  - QueuedComingFromFrontEnd : ExitGames::Client::Photon::Chat
- S -

  - Sphere : ExitGames::Client::GUI
     Steam : ExitGames::Client::Photon::Chat
- t -
  - TimeoutDisconnect: ExitGames::Client::Photon::Chat
- u -
  - Uninitialized : ExitGames::Client::Photon::Chat
- W -
  - WireCube : ExitGames::Client::GUI • WireSphere : ExitGames::Client::GUI
- X -
  - Xbox: ExitGames::Client::Photon::Chat



v1.81

Main Page	Related	Related Pages			Classes	Files
Class List	Class Index	Class Hierarchy		Clas	ss Members	

### **Class List**

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

[detail level 12345]
Container for user auther <b>Photon</b> . Set AuthValues connect - all else is hand
A channel of communicate Chat, updated by ChatC provided as READ ONLY
Central class of the <b>Phot</b> connect, handle channels
Wraps up internally used <b>Photon Chat</b> events. You use them directly usually.
Wraps up codes for operainternally in <b>Photon Cha</b> to use them directly usua

<b>○</b> ChatParameterCode	Wraps up codes for paral operations and events) up <b>Photon Chat</b> . You don't I directly usually.
<b>G</b> ChatPeer	Provides basic operations Chat server. This interna public ChatClient.
<b>©</b> ChatUserStatus	Contains commonly usec for SetOnlineStatus. You own.
<b>©</b> ErrorCode	<b>ErrorCode</b> defines the deassociated with <b>Photon</b> communication.
<b>□</b> IChatClientListener	Callback interface for Che Contains callback method app about updates. Must new ChatClient in constr
ParameterCode	
<b>▼ №</b> Photon	
<b>@</b> MonoBehaviour	This class adds the proper while logging a warning verstill uses the network/Viev
© PunBehaviour	This class provides a .ph callbacks/events that PU Override the events/meth use.
<b>▼ I</b> UnityEngine	
<b>▼ SceneManagement</b>	
<b>G</b> SceneManager	Minimal implementation of SceneManager for older v5.2.
♠ ActorProperties	Class for constants. Thes define "well known" proper Actor / Player. Pun uses internally.

AuthenticationValues	Container for user auther <b>Photon</b> . Set AuthValues connect - all else is hand
EncryptionDataParameters	
<b>©</b> ErrorCode	<b>ErrorCode</b> defines the deassociated with <b>Photon</b> communication.
<b>⊚</b> EventCode	Class for constants. These events defined by <b>Photo</b> Pun uses these constants
<b>©</b> Extensions	This static class defines sextension methods for se classes (e.g. Vector3, floaters)
FriendInfo	Used to store info about a state and in which room h
<b>G</b> GameObjectExtensions	Small number of extensic make it easier for PUN to Unity-versions.
<b>⊚</b> GamePropertyKey	Class for constants. Thes are for "well known" room properties used in <b>Photo</b> Loadbalancing. Pun uses internally.
• HelpURL	Empty implementation of <b>HelpURL</b> of Unity 5.1. The compatibility of attributes
<b> </b>	This interface is used as callback methods of PUN OnPhotonSerializeView. implement them individua
( IPunObservable	Defines the OnPhotonSe method to make it easy to correctly for observable s
	Defines all the methods to Pool must implement, so use it.

OperationCode	Class for constants. Cont codes. Pun uses these cointernally.
© ParameterCode	Class for constants. Code parameters of Operations Pun uses these constants
<b>▼                                    </b>	This class helps you to sy Mecanim animations Sim component to your Game make sure that the <b>Photo</b> is added to the list of obs components
SynchronizedLayer	
SynchronizedParameter	
PhotonLagSimulationGui	This MonoBehaviour is a the <b>Photon</b> client's network feature. It can modify lag jitter (random lag) and pa
PhotonMessageInfo	Container class for info a message, RPC or update
PhotonNetwork	The main class to use the <b>PhotonNetwork</b> plugin. static.
PhotonPingManager	
PhotonPlayer	Summarizes a "player" w identified (in that room) b
♠ PhotonRigidbody2DView	This class helps you to sy velocities of a 2d physics that only the velocities are and because Unitys physics deterministic (ie. the resulthe same on all computer positions of the objects many sync. If you want to have this object the same on a

	should also add a  PhotonTransformView t  the position. Simply add t  your GameObject and ma  PhotonRigidbody2DVie  the list of observed comp
	This class helps you to sy velocities of a physics Rightat only the velocities are and because Unitys physic deterministic (ie. the resulthe same on all computer positions of the objects mayne. If you want to have this object the same on a should also add a <b>PhotonTransformView</b> to the position. Simply add the your GameObject and may <b>PhotonRigidbodyView</b> is list of observed compone
PhotonStatsGui	Basic GUI to show traffic statistics of the connectio toggled by shift+tab.
PhotonStream	This container is used in <b>OnPhotonSerializeView</b> provide incoming data of or for you to provide it.
PhotonStreamQueue	The PhotonStreamQuet object states at higher fre what PhotonNetwork.se and then sends all those when Serialize() is called receiving end you can called and then the stream will received object states in and timeStep they were received.

♠ PhotonTransformView	This class helps you to sy position, rotation and sca GameObject. It also gives different options to make synchronized values appeven when the data is on of times per second. Sim component to your Game make sure that the <b>PhotonTransformView</b> i list of observed compone
PhotonTransformViewPositionControl	
PhotonTransformViewPositionModel	
PhotonTransformViewRotationControl	
PhotonTransformViewRotationModel	
PhotonTransformViewScaleControl	
PhotonTransformViewScaleModel	
PhotonView	PUN's NetworkView replator networking. Use it like
PingMonoEditor	Uses C# Socket class fro System.Net.Sockets (as does).
© PunRPC	Replacement for RPC att different name. Used to fl remote-callable.
RaiseEventOptions	Aggregates several less- options for operation Rais field descriptions for usaç
Region	
<b>©</b> Room	This class resembles a rejoins (or joined). The propertable as opposed to the RoomInfo and you can compour room.

	A simplified room with just required to list and join, unlisting in the lobby. The pasettable (open, MaxPlaye
<b>■</b> RoomOptions	Wraps up common room needed when you create individual entries for more
SceneManagerHelper	
<b>©</b> ServerSettings	Collection of connection-used internally by <b>PhotonNetwork.Connection</b>
■ TypedLobby	Refers to a specific lobby the server.
■ TypedLobbyInfo	
<b>©</b> WebRpcResponse	Reads an operation responsible webRpc and provides conto most common values.

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related Pages		Modules	Classes	Files
Package Functio	ns				
Package I	Exi	tGames			Namespaces

## Namespaces

package Client

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related Pages	Modules	Classes	Files
Package Function	ns			
ExitGames Clie	nt 〉			
Package ExitGame	s.Client			Namespaces

### Namespaces

package **GUI** 

package Photon

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related Pages	Modules	Classes	Files
Package Functio	ns			
ExitGames Clie	nt GUI			
Package ExitGame	s.Client.GUI		<u>Classes</u>	Enumerations

### Classes

class **GizmoTypeDrawer** 

### Enumerations

enum

GizmoType { GizmoType.WireSphere, GizmoType.Sphere, GizmoType.WireCube, GizmoType.Cube }

## **Enumeration Type Documentation**

enum ExitGames.Client.GUI.GizmoType				
Enumerator				
WireSphere				
Sphere				
WireCube				
Cube				

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	e Related Pages		Modules		Classes	Files
Class List	C	Class Index	Class Hierarchy Clas		ss Members	
ExitGames	ExitGames Client GUI GizmoTypeDrawer					
Static Public Member Functions					er Functions	
List of all members					of all members	

# **ExitGames.Client.GUI.GizmoTypeDrawer Class Reference**

### Static Public Member Functions

static void Draw (Vector3 center, GizmoType type, Color color, float size)

### **Member Function Documentation**

```
static void
ExitGames.Client.GUI.GizmoTypeDrawer.Draw ( Vector3 center
GizmoType type,
Color color,
float size
)
```

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related Pages	Modules	Classes	Files
Package Functio	ns			
ExitGames Clie	nt Photon			
Package ExitGame	s.Client.Phot	on		Namespaces

## Namespaces

package Chat

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related Pages	Modules	Classes	Files	
Package Functio	ns				
ExitGames Clie	ExitGames Client Photon Chat				
Package ExitGames.Client.Photor		on.Chat	<u>Classes</u>	Enumerations	

#### Classes

#### class AuthenticationValues

Container for user authentication in **Photon**. Set AuthValues before you connect - all else is handled. More...

#### class ChatChannel

A channel of communication in **Photon Chat**, updated by **ChatClient** and provided as READ ONLY. <u>More...</u>

#### class ChatClient

Central class of the **Photon Chat** API to connect, handle channels and messages. <u>More...</u>

#### class ChatEventCode

Wraps up internally used constants in **Photon Chat** events. You don't have to use them directly usually. More...

#### class ChatOperationCode

Wraps up codes for operations used internally in **Photon Chat**. You don't have to use them directly usually. <u>More...</u>

#### class ChatParameterCode

Wraps up codes for parameters (in operations and events) used internally in **Photon Chat**. You don't have to use them directly usually. More...

#### class ChatPeer

Provides basic operations of the **Photon Chat** server. This internal class is used by public **ChatClient**. <u>More...</u>

#### class ChatUserStatus

Contains commonly used status values for SetOnlineStatus. You can define your own. More...

class **ErrorCode** 

ErrorCode defines the default codes associated with

**Photon** client/server communication. More...

interface IChatClientListener

Callback interface for **Chat** client side. Contains callback methods to notify your app about updates. Must be

provided to new **ChatClient** in constructor <u>More...</u>

class ParameterCode

#### **Enumerations**

```
ChatDisconnectCause.None,
ChatDisconnectCause.DisconnectByServerUserLimit,
ChatDisconnectCause.ExceptionOnConnect,
ChatDisconnectCause.DisconnectByServer,
ChatDisconnectCause.TimeoutDisconnect,

enum
ChatDisconnectCause.Exception,
ChatDisconnectCause.InvalidAuthentication,
ChatDisconnectCause.MaxCcuReached,
ChatDisconnectCause.InvalidRegion,
ChatDisconnectCause.OperationNotAllowedInCurrentState
ChatDisconnectCause.CustomAuthenticationFailed
}
Enumaration of causes for Disconnects (used in
LoadBalancingClient.DisconnectedCause). More...
```

```
CustomAuthenticationType.Custom = 0,
CustomAuthenticationType.Steam = 1,
CustomAuthenticationType.Facebook = 2,
enum
CustomAuthenticationType.Oculus = 3,
CustomAuthenticationType.PlayStation = 4,
CustomAuthenticationType.Xbox = 5,
CustomAuthenticationType.None = byte.MaxValue
}
Options for optional "Custom Authentication" services used with Photon. Used by OpAuthenticate after connecting to Photon. More...
```

```
ChatState {
    ChatState.Uninitialized,
    ChatState.ConnectingToNameServer,
    ChatState.ConnectedToNameServer,
    ChatState.Authenticating,
    ChatState.Authenticated,
enum ChatState.DisconnectingFromNameServer,
```

```
ChatState.ConnectingToFrontEnd,
ChatState.ConnectedToFrontEnd,
ChatState.DisconnectingFromFrontEnd,
ChatState.QueuedComingFromFrontEnd,
ChatState.Disconnecting, ChatState.Disconnected
}
Possible states for a LoadBalancingClient. More...
```

### **Enumeration Type Documentation**

#### enum ExitGames.Client.Photon.Chat.ChatDisconnectCause

Enumaration of causes for Disconnects (used in LoadBalancingClient.DisconnectedCause).

Read the individual descriptions to find out what to do about this type of disconnect.

Enumerator	
None	No error was tracked.
DisconnectByServerUserLimit	OnStatusChanged: The CCUs count of your <b>Photon</b> Server License is exausted (temporarily).
ExceptionOnConnect	OnStatusChanged: The server is not available or the address is wrong. Make sure the port is provided and the server is up.
DisconnectByServer	OnStatusChanged: The server disconnected this client. Most likely the server's send buffer is full (receiving too much from other clients).
TimeoutDisconnect	OnStatusChanged: This client detected that the server's responses are not received in due time. Maybe you send / receive too much?
Exception	OnStatusChanged: Some

	internal exception caused the socket code to fail. Contact Exit Games.
InvalidAuthentication	OnOperationResponse: Authenticate in the <b>Photon</b> Cloud with invalid Appld. Update your subscription or contact Exit Games.
MaxCcuReached	OnOperationResponse: Authenticate (temporarily) failed when using a <b>Photon</b> Cloud subscription without CCU Burst. Update your subscription.
InvalidRegion	OnOperationResponse: Authenticate when the app's Photon Cloud subscription is locked to some (other) region(s). Update your subscription or master server address.
OperationNotAllowedInCurrentState	OnOperationResponse: Operation that's (currently) not available for this client (not authorized usually). Only tracked for op Authenticate.
CustomAuthenticationFailed	OnOperationResponse: Authenticate in the <b>Photon</b> Cloud with invalid client values or custom authentication setup in Cloud Dashboard.

### enum ExitGames.Client.Photon.Chat.ChatState

Possible states for a LoadBalancingClient.

Enumerator	
Uninitialized	Peer is created but not used yet.
ConnectingToNameServer	Connecting to master (includes connect, authenticate and joining the lobby)
ConnectedToNameServer	Connected to master server.
Authenticating	Usually when Authenticated, the client will join a game or the lobby (if AutoJoinLobby is true).
Authenticated	Usually when Authenticated, the client will join a game or the lobby (if AutoJoinLobby is true).
DisconnectingFromNameServer	Transition from master to game server.
ConnectingToFrontEnd	Transition to gameserver (client will authenticate and join/create game).
ConnectedToFrontEnd	Connected to gameserver (going to auth and join game).
DisconnectingFromFrontEnd	Transition from gameserver to master (after leaving a room/game).
QueuedComingFromFrontEnd	Currently not used.
Disconnecting	The client disconnects (from any server).
Disconnected	The client is no longer connected (to any server). Connect to master to go on.

## enum ExitGames.Client.Photon.Chat.CustomAuthenticationType : byte

Options for optional "Custom Authentication" services used with

**Photon**. Used by OpAuthenticate after connecting to **Photon**.

Enumerator	
Custom	Use a custom authentification service. Currently the only implemented option.
Steam	Authenticates users by their Steam Account. Set auth values accordingly!
Facebook	Authenticates users by their Facebook Account. Set auth values accordingly!
Oculus	Authenticates users by their Oculus Account and token.
PlayStation	Authenticates users by their PSN Account and token.
Xbox	Authenticates users by their Xbox Account and XSTS token.
None	Disables custom authentification. Same as not providing any <b>AuthenticationValues</b> for connect (more precisely for: OpAuthenticate).

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	e Relate	ed Pages	Modules		Classes	Files
Class List	Class Index	Class Hi	ierarchy	Class Members		
ExitGames	ExitGames Client Photon Chat Authentication Values					
	Public Member Functions   Properties					
<u>List of all members</u>						
ExitGames Client Photon Chat Authentication Va						

# ExitGames.Client.Photon.Chat.AuthenticationVa Class Reference

Container for user authentication in **Photon**. Set AuthValues before you connect - all else is handled. <u>More...</u>

#### **Public Member Functions**

#### **AuthenticationValues ()**

Creates empty auth values without any info. More...

#### **AuthenticationValues** (string userId)

Creates minimal info about the user. If this is authenticated or not, depends on the set AuthType. More...

#### virtual void **SetAuthPostData** (string stringData)

Sets the data to be passed-on to the auth service via POST. More...

#### virtual void SetAuthPostData (byte[] byteData)

Sets the data to be passed-on to the auth service via POST. More...

#### virtual void AddAuthParameter (string key, string value)

Adds a key-value pair to the get-parameters used for Custom Auth. More...

#### override string ToString ()

# **Properties**

# CustomAuthenticationType AuthType [get, set]

The type of custom authentication provider that should be used. Currently only "Custom" or "None" (turns this off). More...

### string AuthGetParameters [get, set]

This string must contain any (http get) parameters expected by the used authentication service. By default, username and token. More...

### object AuthPostData [get, set]

Data to be passed-on to the auth service via POST. Default: null (not sent). Either string or byte[] (see setters). More...

# string Token [get, set]

After initial authentication, **Photon** provides a token for this client / user, which is subsequently used as (cached) validation. More...

# string UserId [get, set]

The UserId should be a unique identifier per user. This is for finding friends, etc.. <u>More...</u>

# **Detailed Description**

Container for user authentication in **Photon**. Set AuthValues before you connect - all else is handled.

On **Photon**, user authentication is optional but can be useful in many cases. If you want to FindFriends, a unique ID per user is very practical.

There are basically three options for user authentification: None at all, the client sets some Userld or you can use some account web-service to authenticate a user (and set the Userld server-side).

Custom Authentication lets you verify end-users by some kind of login or token. It sends those values to **Photon** which will verify them before granting access or disconnecting the client.

The **Photon** Cloud Dashboard will let you enable this feature and set important server values for it.

https://www.photonengine.com/dashboard

# Constructor & Destructor Documentation

### ExitGames.Client.Photon.Chat.AuthenticationValues.Authenticatio

Creates empty auth values without any info.

# ExitGames.Client.Photon.Chat.AuthenticationValues.Authenticatio

Creates minimal info about the user. If this is authenticated or not, depe

### **Parameters**

userId Some UserId to set in Photon.

# **Member Function Documentation**

### virtual void

### ExitGames.Client.Photon.Chat.AuthenticationValues.AddAuthPara

Adds a key-value pair to the get-parameters used for Custom Auth.

This method does uri-encoding for you.

### **Parameters**

**key** Key for the value to set.

**value** Some value relevant for Custom Authentication.

### virtual void

### ExitGames.Client.Photon.Chat.AuthenticationValues.SetAuthPostI

Sets the data to be passed-on to the auth service via POST.

### **Parameters**

**stringData** String data to be used in the body of the POST reques AuthPostData to null.

### virtual void

### ExitGames.Client.Photon.Chat.AuthenticationValues.SetAuthPostI

Sets the data to be passed-on to the auth service via POST.

### **Parameters**

byteData Binary token / auth-data to pass on.

override string ExitGames.Client.Photon.Chat.AuthenticationValues.ToString ( )

# **Property Documentation**

### string

### ExitGames.Client.Photon.Chat.AuthenticationValues.AuthGetParai

This string must contain any (http get) parameters expected by the use authentication service. By default, username and token.

Standard http get parameters are used here and passed on to the servi defined in the server (Photon Cloud Dashboard).

# object

### ExitGames.Client.Photon.Chat.AuthenticationValues.AuthPostData

Data to be passed-on to the auth service via POST. Default: null (not se Either string or byte[] (see setters).

# CustomAuthenticationType

# ExitGames.Client.Photon.Chat.AuthenticationValues.AuthType



The type of custom authentication provider that should be used. Curren only "Custom" or "None" (turns this off).

# string

# ExitGames.Client.Photon.Chat.AuthenticationValues.Token get set





After initial authentication, **Photon** provides a token for this client / user, which is subsequently used as (cached) validation.

# string

# ExitGames.Client.Photon.Chat.AuthenticationValues.UserId get se



The UserId should be a unique identifier per user. This is for finding friends, etc..

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page	Related	Pages	Modu	ıles	Classes	Files			
Class List	Class Index	Class Hi	erarchy	Class Members					
ExitGames Client Photon Chat ChatChannel									
Public Member Functions   Public Attributes									
Properties   List of all members									
ExitGames.Client.Photon.Chat.ChatChannel Class Reference									

A channel of communication in **Photon Chat**, updated by **ChatClient** and provided as READ ONLY. <u>More...</u>

# **Public Member Functions**

# **ChatChannel** (string name)

Used internally to create new channels. This does NOT create a channel on the server! Use **ChatClient.Subscribe**. More...

# void **Add** (string sender, object message)

Used internally to add messages to this channel. More...

# void Add (string[] senders, object[] messages)

Used internally to add messages to this channel. More...

### void TruncateMessages ()

Reduces the number of locally cached messages in this channel to the MessageLimit (if set). More...

# void ClearMessages ()

Clear the local cache of messages currently stored. This frees memory but doesn't affect the server. More...

# string ToStringMessages ()

Provides a string-representation of all messages in this channel. <u>More...</u>

# **Public Attributes**

# readonly string Name

Name of the channel (used to subscribe and unsubscribe). More...

# readonly List< string > Senders = new List<string>()

Senders of messages in chronoligical order. Senders and Messages refer to each other by index. Senders[x] is the sender of Messages[x]. <u>More...</u>

# readonly List< object > Messages = new List<object>()

Messages in chronoligical order. Senders and Messages refer to each other by index. Senders[x] is the sender of Messages[x]. More...

### int MessageLimit

If greater than 0, this channel will limit the number of messages, that it caches locally. More...

# **Properties**

# bool IsPrivate [get, set]

Is this a private 1:1 channel? More...

# int MessageCount [get]

Count of messages this client still buffers/knows for this channel. <u>More...</u>

# **Detailed Description**

A channel of communication in **Photon Chat**, updated by **ChatClient** and provided as READ ONLY.

Contains messages and senders to use (read!) and display by your GUI. Access these by: ChatClient.PublicChannels
ChatClient.PrivateChannels

# Constructor & Destructor Documentation

# ExitGames.Client.Photon.Chat.ChatChannel.ChatChannel (string

Used internally to create new channels. This does NOT create a channel the server! Use **ChatClient.Subscribe**.

# **Member Function Documentation**

# void ExitGames.Client.Photon.Chat.ChatChannel.Add (string sender, object message)

Used internally to add messages to this channel.

# void ExitGames.Client.Photon.Chat.ChatChannel.Add ( string[] senders object[] messaç )

Used internally to add messages to this channel.

# void ExitGames.Client.Photon.Chat.ChatChannel.ClearMessages ( )

Clear the local cache of messages currently stored. This frees memory but doesn't affect the server.

# string

ExitGames.Client.Photon.Chat.ChatChannel.ToStringMessages ( )

Provides a string-representation of all messages in this channel.

### **Returns**

All known messages in format "Sender: Message", line by line.

# void ExitGames.Client.Photon.Chat.ChatChannel.TruncateMessages ( )

Reduces the number of locally cached messages in this channel to the MessageLimit (if set).

# Member Data Documentation

### int ExitGames.Client.Photon.Chat.ChatChannel.MessageLimit

If greater than 0, this channel will limit the number of messages, that it caches locally.

# readonly List<object> ExitGames.Client.Photon.Chat.ChatChannel.Messages = new List<object>()

Messages in chronoligical order. Senders and Messages refer to each other by index. Senders[x] is the sender of Messages[x].

### readonly string ExitGames.Client.Photon.Chat.ChatChannel.Name

Name of the channel (used to subscribe and unsubscribe).

# readonly List<string> ExitGames.Client.Photon.Chat.ChatChannel.Senders = new List<string>()

Senders of messages in chronoligical order. Senders and Messages refer to each other by index. Senders[x] is the sender of Messages[x].

# **Property Documentation**

### bool

ExitGames.Client.Photon.Chat.ChatChannel.IsPrivate





Is this a private 1:1 channel?

### int

ExitGames.Client.Photon.Chat.ChatChannel.MessageCount



Count of messages this client still buffers/knows for this channel.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page	e Related	Pages	Modu	lles	Classes	Files			
Class List	Class Index	Class Hi	erarchy	Class Members					
ExitGames Client Photon Chat ChatClient									
Public Member Functions   Public Attributes									
Properties   List of all members									
ExitGames.Client.Photon.Chat.ChatClient Class Reference									

Central class of the **Photon Chat** API to connect, handle channels and messages. <u>More...</u>

Inherits IPhotonPeerListener.

# **Public Member Functions**

# bool CanChatInChannel (string channelName)

**ChatClient (IChatClientListener** listener, ConnectionProtocol protocol=ConnectionProtocol.Udp)

# bool Connect (string appld, string appVersion, AuthenticationValues authValues)

Connects this client to the **Photon Chat** Cloud service, which will also authenticate the user (and set a UserId). More...

# void Service ()

Must be called regularly to keep connection between client and server alive and to process incoming messages. More...

### void Disconnect ()

# void StopThread ()

# bool Subscribe (string[] channels)

Sends operation to subscribe to a list of channels by name. More...

# bool **Subscribe** (string[] channels, int messagesFromHistory) Sends operation to subscribe client to channels, optionally fetching a number of messages from the cache. More...

# bool Unsubscribe (string[] channels)

Unsubscribes from a list of channels, which stops getting messages from those. More...

# bool **PublishMessage** (string channelName, object message) Sends a message to a public channel which this client subscribed to. More...

# bool **SendPrivateMessage** (string target, object message)

Sends a private message to a single target user. Calls OnPrivateMessage on the receiving client. <u>More...</u>

# bool SendPrivateMessage (string target, object message, bool encrypt)

Sends a private message to a single target user. Calls OnPrivateMessage on the receiving client. <u>More...</u>

# bool **SetOnlineStatus** (int status)

Sets the user's status without changing your status-message. More...

# bool **SetOnlineStatus** (int status, object message)

Sets the user's status without changing your status-message. More...

# bool AddFriends (string[] friends)

Adds friends to a list on the **Chat** Server which will send you status updates for those. <u>More...</u>

# bool RemoveFriends (string[] friends)

Removes the provided entries from the list on the **Chat** Server and stops their status updates. <u>More...</u>

# string **GetPrivateChannelNameByUser** (string userName)

Get you the (locally used) channel name for the chat between this client and another user. More...

# bool **TryGetChannel** (string channelName, bool isPrivate, out **ChatChannel** channel)

Simplified access to either private or public channels by name. More...

# bool **TryGetChannel** (string channelName, out **ChatChannel** channel)

Simplified access to all channels by name. Checks public

channels first, then private ones. More...

# void SendAcksOnly ()

# **Public Attributes**

# int MessageLimit

If greater than 0, new channels will limit the number of messages they cache locally. More...

readonly Dictionary< string, **ChatChannel** >

**PublicChannels** 

readonly Dictionary< string, **ChatChannel** >

**PrivateChannels** 

ChatPeer chatPeer = null

# **Properties**

# string NameServerAddress [get, set]

The address of last connected Name Server. More...

# string FrontendAddress [get, set]

The address of the actual chat server assigned from NameServer. Public for read only. More...

# string ChatRegion [get, set]

Settable only before you connect! Defaults to "EU". More...

# ChatState State [get, set]

Current state of the **ChatClient**. Also use CanChat. More...

# ChatDisconnectCause DisconnectedCause [get, set]

bool CanChat [get]

# string AppVersion [get, set]

The version of your client. A new version also creates a new "virtual app" to separate players from older client versions. More...

# string Appld [get, set]

The AppID as assigned from the **Photon** Cloud. If you host yourself, this is the "regular" **Photon** Server Application Name (most likely: "LoadBalancing"). More...

# AuthenticationValues AuthValues [get, set]

Settable only before you connect! More...

# string UserId [get, set]

The unique ID of a user/person, stored in AuthValues.UserId. Set it before you connect. More...

# DebugLevel DebugOut [get, set]

Sets the level (and amount) of debug output provided by the library. More...

# **Detailed Description**

Central class of the **Photon Chat** API to connect, handle channels and messages.

This class must be instantiated with a **IChatClientListener** instance to get the callbacks. Integrate it into your game loop by calling Service regularly. Call Connect with an Appld that is setup as **Photon Chat** application. Note: Connect covers multiple messages between this client and the servers. A short workflow will connect you to a chat server.

Each **ChatClient** resembles a user in chat (set in Connect). Each user automatically subscribes a channel for incoming private messages and can message any other user privately. Before you publish messages in any non-private channel, that channel must be subscribed.

PublicChannels is a list of subscribed channels, containing messages and senders. PrivateChannels contains all incoming and sent private messages.

# Constructor & Destructor Documentation

ExitGames.Client.Photon.Chat.ChatClient.ChatClient (IChatClientL Connection)

# Member Function Documentation

### bool

ExitGames.Client.Photon.Chat.ChatClient.AddFriends (string[] friends

Adds friends to a list on the **Chat** Server which will send you status upd for those.

AddFriends and RemoveFriends enable clients to handle their friend lis the **Photon Chat** server. Having users on your friends list gives you act to their current online status (and whatever info your client sets in it).

Each user can set an online status consisting of an integer and an arbit (serializable) object. The object can be null, Hashtable, object[] or anythele Photon can serialize.

The status is published automatically to friends (anyone who set your u ID with AddFriends).

**Photon** flushes friends-list when a chat client disconnects, so it has to I each time. If your community API gives you access to online status alre you could filter and set online friends in AddFriends.

Actual friend relations are not persistent and have to be stored outside **Photon**.

#### **Parameters**

friends Array of friend userIds.

#### Returns

If the operation could be sent.

#### bool

ExitGames.Client.Photon.Chat.ChatClient.CanChatInChannel (stri

# bool ExitGames.Client.Photon.Chat.ChatClient.Connect ( string string Authentication )

Connects this client to the **Photon Chat** Cloud service, which will also a (and set a UserId).

### **Parameters**

appld Get your Photon Chat Appld from the <u>Dashboard</u>.

**appVersion** Any version string you make up. Used to separate use your clients, which might be incompatible.

**authValues** Values for authentication. You can leave this null, if yo before. If you set authValues, they will override any U

### Returns

### void ExitGames.Client.Photon.Chat.ChatClient.Disconnect()

### string

ExitGames.Client.Photon.Chat.ChatClient.GetPrivateChannelName

Get you the (locally used) channel name for the chat between this clien

### **Parameters**

userName Remote user's name or Userld.

#### Returns

The (locally used) channel name for a private channel.

### bool

ExitGames.Client.Photon.Chat.ChatClient.PublishMessage ( string object

Sends a message to a public channel which this client subscribed to.

Before you publish to a channel, you have to subscribe it. Everyone in t get the message.

### **Parameters**

channelName Name of the channel to publish to.message Your message (string or any serializable data).

### Returns

False if the client is not yet ready to send messages.

### bool

### ExitGames.Client.Photon.Chat.ChatClient.RemoveFriends ( string[

Removes the provided entries from the list on the **Chat** Server and stor status updates.

**Photon** flushes friends-list when a chat client disconnects. Unless you remove individual entries, you don't have to RemoveFriends.

AddFriends and RemoveFriends enable clients to handle their friend lis **Photon Chat** server. Having users on your friends list gives you access current online status (and whatever info your client sets in it).

Each user can set an online status consisting of an integer and an arbit (serializable) object. The object can be null, Hashtable, object[] or anythe Photon can serialize.

The status is published automatically to friends (anyone who set your u with AddFriends).

**Photon** flushes friends-list when a chat client disconnects, so it has to I each time. If your community API gives you access to online status alre could filter and set online friends in AddFriends.

Actual friend relations are not persistent and have to be stored outside **Photon**.

AddFriends and RemoveFriends enable clients to handle their friend lis **Photon Chat** server. Having users on your friends list gives you access current online status (and whatever info your client sets in it).

Each user can set an online status consisting of an integer and an arbit (serializable) object. The object can be null, Hashtable, object[] or anythe Photon can serialize.

The status is published automatically to friends (anyone who set your u with AddFriends).

Actual friend relations are not persistent and have to be stored outside **Photon**.

#### **Parameters**

friends Array of friend userIds.

### **Returns**

If the operation could be sent.

### void

ExitGames.Client.Photon.Chat.ChatClient.SendAcksOnly

### bool

ExitGames.Client.Photon.Chat.ChatClient.SendPrivateMessage ( s

0

١

()

Sends a private message to a single target user. Calls OnPrivateMessareceiving client.

### **Parameters**

**target** Username to send this message to.

**message** The message you want to send. Can be a simple string serializable.

### **Returns**

True if this clients can send the message to the server.

### bool

ExitGames.Client.Photon.Chat.ChatClient.SendPrivateMessage ( st

0

b

)

Sends a private message to a single target user. Calls OnPrivateMessa receiving client.

### **Parameters**

**target** Username to send this message to.

message The message you want to send. Can be a simple string

serializable.

**encrypt** Optionally, private messages can be encrypted. Encrypt

to-end as the server decrypts the message.

### Returns

True if this clients can send the message to the server.

# void ExitGames.Client.Photon.Chat.ChatClient.Service ( )

Must be called regularly to keep connection between client and server alive and to process incoming messages.

This method limits the effort it does automatically using the private variable msDeltaForServiceCalls. That value is lower for connect and multiplied by 4 when chat-server connection is ready.

#### bool

ExitGames.Client.Photon.Chat.ChatClient.SetOnlineStatus (int sta

Sets the user's status without changing your status-message.

The predefined status values can be found in class ChatUserStatus. S

**ChatUserStatus.Invisible** will make you offline for everyone and send message.

You can set custom values in the status integer. Aside from the preconfigured ones, all states will be considered visible and online. Else, n one would see the custom state.

This overload does not change the set message.

### **Parameters**

**status** Predefined states are in class **ChatUserStatus**. Other valu can be used at will.

### **Returns**

True if the operation gets called on the server.

### bool

ExitGames.Client.Photon.Chat.ChatClient.SetOnlineStatus ( int object )

Sets the user's status without changing your status-message.

The predefined status values can be found in class **ChatUserStatus**. S **ChatUserStatus.Invisible** will make you offline for everyone and send message.

You can set custom values in the status integer. Aside from the pre-con ones, all states will be considered visible and online. Else, no one would custom state.

The message object can be anything that **Photon** can serialize, including limited to) Hashtable, object[] and string. This value is defined by your conventions.

#### **Parameters**

**status** Predefined states are in class **ChatUserStatus**. Other v be used at will.

message Also sets a status-message which your friends can get.

### **Returns**

True if the operation gets called on the server.

# void ExitGames.Client.Photon.Chat.ChatClient.StopThread ( )

### bool

ExitGames.Client.Photon.Chat.ChatClient.Subscribe (string[] cha

Sends operation to subscribe to a list of channels by name.

### **Parameters**

**channels** List of channels to subscribe to. Avoid null or empty value

### Returns

If the operation could be sent at all (Example: Fails if not connected **Chat** Server).

#### bool

ExitGames.Client.Photon.Chat.ChatClient.Subscribe (string[] chaint int mes

Sends operation to subscribe client to channels, optionally fetching a nu from the cache.

Subscribes channels will forward new messages to this user. Use Publi so. The messages cache is limited but can be useful to get into ongoing that's needed.

### **Parameters**

**channels**List of channels to subscribe to. Avoid nu **messagesFromHistory** 0: no history. 1 and higher: number of me

-1: all available history.

#### Returns

If the operation could be sent at all (Example: Fails if not connected

### bool

ExitGames.Client.Photon.Chat.ChatClient.TryGetChannel (string bool out Cha )

Simplified access to either private or public channels by name.

### **Parameters**

channelName Name of the channel to get. For private channels, t

composed of both user's names.

Define if you expect a private or public channel. isPrivate Out parameter gives you the found channel, if any. channel

### Returns

True if the channel was found.

### bool

ExitGames.Client.Photon.Chat.ChatClient.TryGetChannel (string out Cha

Simplified access to all channels by name. Checks public channels first

#### **Parameters**

channelName Name of the channel to get.

Out parameter gives you the found channel, if any. channel

#### Returns

True if the channel was found.

### bool

ExitGames.Client.Photon.Chat.ChatClient.Unsubscribe (string[] c

Unsubscribes from a list of channels, which stops getting messages fro

The client will remove these channels from the PublicChannels dictiona the server sent a response to this request.

The request will be sent to the server and **IChatClientListener.OnUnsubscribed** gets called when the server act removed the channel subscriptions.

Unsubscribe will fail if you include null or empty channel names.

### **Parameters**

**channels** Names of channels to unsubscribe.

### Returns

False, if not connected to a chat server.

# Member Data Documentation

# ChatPeer ExitGames.Client.Photon.Chat.ChatClient.chatPeer = null

# int ExitGames.Client.Photon.Chat.ChatClient.MessageLimit

If greater than 0, new channels will limit the number of messages they cache locally.

This can be useful to limit the amount of memory used by chats. You can set a MessageLimit per channel but this value gets applied to new ones.

Note: Changing this value, does not affect ChatChannels that are already in use!

readonly Dictionary<string, ChatChannel>
ExitGames.Client.Photon.Chat.ChatClient.PrivateChannels

readonly Dictionary<string, ChatChannel>
ExitGames.Client.Photon.Chat.ChatClient.PublicChannels

#### **Property Documentation**

#### string ExitGames.Client.Photon.Chat.ChatClient.Appld





The AppID as assigned from the **Photon** Cloud. If you host yourself, this is the "regular" **Photon** Server Application Name (most likely: "LoadBalancing").

#### string

#### ExitGames.Client.Photon.Chat.ChatClient.AppVersion





The version of your client. A new version also creates a new "virtual app" to separate players from older client versions.

#### **AuthenticationValues**

#### ExitGames.Client.Photon.Chat.ChatClient.AuthValues





Settable only before you connect!

#### bool ExitGames.Client.Photon.Chat.ChatClient.CanChat



#### string

#### ExitGames.Client.Photon.Chat.ChatClient.ChatRegion





Settable only before you connect! Defaults to "EU".

#### DebugLevel

ExitGames.Client.Photon.Chat.ChatClient.DebugOut





Sets the level (and amount) of debug output provided by the library.

This affects the callbacks to IChatClientListener.DebugReturn. Default Level: Error.

#### ChatDisconnectCause

ExitGames.Client.Photon.Chat.ChatClient.DisconnectedCause



#### string

ExitGames.Client.Photon.Chat.ChatClient.FrontendAddress get se



The address of the actual chat server assigned from NameServer. Public for read only.

#### string

ExitGames.Client.Photon.Chat.ChatClient.NameServerAddress



The address of last connected Name Server.

#### ChatState

ExitGames.Client.Photon.Chat.ChatClient.State





Current state of the **ChatClient**. Also use CanChat.

#### string ExitGames.Client.Photon.Chat.ChatClient.UserId



The unique ID of a user/person, stored in AuthValues.UserId. Set it before you connect.

This value wraps AuthValues. UserId. It's not a nickname and we assume users with the same userID are the same person.



# Photon Unity Networking

v1.81

Main Page Related		Pages Module		lles	Classes	Files	
Class List	t Class Index		Class Hierarchy		Class Members		
ExitGames Client Photon Chat ChatEventCode							
Public Attributes   List of all members  ExitGames.Client.Photon.Chat.ChatEventCode  Class Reference							

Wraps up internally used constants in **Photon Chat** events. You don't have to use them directly usually. <u>More...</u>

#### **Public Attributes**

const byte **ChatMessages** = 0

const byte Users = 1

const byte **PrivateMessage** = 2

const byte **FriendsList** = 3

const byte **StatusUpdate** = 4

const byte **Subscribe** = 5

const byte **Unsubscribe** = 6

## **Detailed Description**

Wraps up internally used constants in **Photon Chat** events. You don't have to use them directly usually.

#### Member Data Documentation

const byte

ExitGames.Client.Photon.Chat.ChatEventCode.ChatMessages = 0

const byte

ExitGames.Client.Photon.Chat.ChatEventCode.FriendsList = 3

const byte

ExitGames.Client.Photon.Chat.ChatEventCode.PrivateMessage = 2

const byte

ExitGames.Client.Photon.Chat.ChatEventCode.StatusUpdate = 4

const byte

ExitGames.Client.Photon.Chat.ChatEventCode.Subscribe = 5

const byte

ExitGames.Client.Photon.Chat.ChatEventCode.Unsubscribe = 6

const byte ExitGames.Client.Photon.Chat.ChatEventCode.Users = 1



## Photon Unity Networking

v1.81

Main Page	Related	Related Pages		les	Classes	Files		
Class List	ss List Class Index		Class Hierarchy		ss Members			
ExitGames Client Photon Chat ChatOperationCode								
Public Attributes   List of all memb								
ExitGames.Client.Photon.Chat.ChatOperationCc								

Wraps up codes for operations used internally in **Photon Chat**. You don't have to use them directly usually. <u>More...</u>

#### **Public Attributes**

const byte Authenticate = 230

(230) Operation Authenticate. More...

const byte Subscribe = 0

(0) Operation to subscribe to chat channels. More...

const byte **Unsubscribe** = 1

(1) Operation to unsubscribe from chat channels. More...

const byte **Publish** = 2

(2) Operation to publish a message in a chat channel. More...

const byte **SendPrivate** = 3

(3) Operation to send a private message to some other user. More...

const byte **ChannelHistory** = 4

(4) Not used yet. More...

const byte **UpdateStatus** = 5

(5) Set your (client's) status. More...

const byte AddFriends = 6

(6) Add friends the list of friends that should update you of their status. More...

const byte **RemoveFriends** = 7

(7) Remove friends from list of friends that should update you of their status. <u>More...</u>

## **Detailed Description**

Wraps up codes for operations used internally in **Photon Chat**. You don't have to use them directly usually.

#### Member Data Documentation

#### const byte

ExitGames.Client.Photon.Chat.ChatOperationCode.AddFriends = 6

(6) Add friends the list of friends that should update you of their status.

#### const byte

ExitGames.Client.Photon.Chat.ChatOperationCode.Authenticate = 230

(230) Operation Authenticate.

#### const byte

ExitGames.Client.Photon.Chat.ChatOperationCode.ChannelHistory = 4

(4) Not used yet.

#### const byte

ExitGames.Client.Photon.Chat.ChatOperationCode.Publish = 2

(2) Operation to publish a message in a chat channel.

#### const byte

ExitGames.Client.Photon.Chat.ChatOperationCode.RemoveFriend = 7

(7) Remove friends from list of friends that should update you of their status.

#### const byte

ExitGames.Client.Photon.Chat.ChatOperationCode.SendPrivate = 3

(3) Operation to send a private message to some other user.

#### const byte

ExitGames.Client.Photon.Chat.ChatOperationCode.Subscribe = 0

(0) Operation to subscribe to chat channels.

#### const byte

ExitGames.Client.Photon.Chat.ChatOperationCode.Unsubscribe = 1

(1) Operation to unsubscribe from chat channels.

#### const byte

ExitGames.Client.Photon.Chat.ChatOperationCode.UpdateStatus = 5

(5) Set your (client's) status.



## Photon Unity Networking

v1.81

Main Page		Related Pages		Modules		Classes	Files	
Class List	Class Index Class Hie		ierarchy	Class Members				
ExitGames	Clier	nt Photon	Chat	ChatParameterCode >				
					Public	Attributes   List	of all members	

# ExitGames.Client.Photon.Chat.ChatParameterCo

Wraps up codes for parameters (in operations and events) used internally in **Photon Chat**. You don't have to use them directly usually. More...

#### **Public Attributes**

const byte Channels = 0

(0) Array of chat channels. More...

const byte **Channel** = 1

(1) Name of a single chat channel. More...

const byte Messages = 2

(2) Array of chat messages. More...

const byte Message = 3

(3) A single chat message. More...

const byte **Senders** = 4

(4) Array of names of the users who sent the array of chat mesages. More...

const byte **Sender** = 5

(5) Name of a the user who sent a chat message. More...

const byte **ChannelUserCount** = 6

(6) Not used. More...

const byte **UserId** = 225

(225) Name of user to send a (private) message to. More...

const byte Msgld = 8

(8) Id of a message. More...

const byte Msglds = 9

(9) Not used. More...

const byte **Secret** = 221

(221) Secret token to identify an authorized user. More...

#### const byte **SubscribeResults** = 15

(15) Subscribe operation result parameter. A bool[] with result per channel. <u>More...</u>

#### const byte **Status** = 10

(10) Status More...

#### const byte Friends = 11

(11) Friends More...

#### const byte **SkipMessage** = 12

(12) SkipMessage is used in SetOnlineStatus and if true, the message is not being broadcast. More...

#### const byte **HistoryLength** = 14

(14) Number of message to fetch from history. 0: no history. 1 and higher: number of messages in history. -1: all history. More...

## **Detailed Description**

Wraps up codes for parameters (in operations and events) used internally in **Photon Chat**. You don't have to use them directly usually.

#### Member Data Documentation

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.Channel = 1

(1) Name of a single chat channel.

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.Channels = 0

(0) Array of chat channels.

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.ChannelUserC = 6

(6) Not used.

#### const byte

**ExitGames.Client.Photon.Chat.ChatParameterCode.Friends = 11** 

(11) Friends

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.HistoryLength = 14

(14) Number of message to fetch from history. 0: no history. 1 and higher: number of messages in history. -1: all history.

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.Message = 3

(3) A single chat message.

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.Messages = 2

(2) Array of chat messages.

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.MsgId = 8

(8) Id of a message.

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.Msglds = 9

(9) Not used.

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.Secret = 221

(221) Secret token to identify an authorized user.

The code is used in LoadBalancing and copied over here.

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.Sender = 5

(5) Name of a the user who sent a chat message.

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.Senders = 4

(4) Array of names of the users who sent the array of chat mesages.

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.SkipMessage = 12

(12) SkipMessage is used in SetOnlineStatus and if true, the message is not being broadcast.

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.Status = 10

(10) Status

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.SubscribeResi = 15

(15) Subscribe operation result parameter. A bool[] with result per channel.

#### const byte

ExitGames.Client.Photon.Chat.ChatParameterCode.UserId = 225

(225) Name of user to send a (private) message to.

The code is used in LoadBalancing and copied over here.



## Photon Unity Networking

v1.81

Main Page		Related Pages		Modules		Classes	Files	
Class List	С	lass Index	Class Hi	erarchy	Clas	ss Members		
ExitGames Client Photon Chat ChatPeer								
				Public I	<u>Membe</u> i	Functions   Pub	olic Attributes	
Properties   List of all members						of all members		
ExitGames.Client.Photon.Chat.ChatPeer Class								
Referer	nce	<b>)</b>						

Provides basic operations of the **Photon Chat** server. This internal class is used by public **ChatClient**. <u>More...</u>

Inherits PhotonPeer.

#### **Public Member Functions**

**ChatPeer** (IPhotonPeerListener listener, ConnectionProtocol protocol)

bool Connect ()

bool AuthenticateOnNameServer (string appld, string appVersion, string region, AuthenticationValues authValues)

#### **Public Attributes**

const string NameServerHost = "ns.exitgames.com"

Name Server Host Name for **Photon** Cloud. Without

port and without any prefix. More...

const string NameServerHttp =

"http://ns.exitgamescloud.com:80/photon/n"

Name Server for HTTP connections to the **Photon** 

Cloud. Includes prefix and port. More...

## **Properties**

#### string NameServerAddress [get]

Name Server Address for **Photon** Cloud (based on current protocol). You can use the default values and usually won't have to set this value. <u>More...</u>

## **Detailed Description**

Provides basic operations of the **Photon Chat** server. This internal class is used by public **ChatClient**.

### Constructor & Destructor Documentation

ExitGames.Client.Photon.Chat.ChatPeer.ChatPeer ( IPhotonPeerLie ConnectionPro )

### **Member Function Documentation**

#### bool

ExitGames.Client.Photon.Chat.ChatPeer.AuthenticateOnNameServ

bool ExitGames.Client.Photon.Chat.ChatPeer.Connect ( )

#### Member Data Documentation

#### const string

ExitGames.Client.Photon.Chat.ChatPeer.NameServerHost = "ns.exitgames.com"

Name Server Host Name for **Photon** Cloud. Without port and without any prefix.

#### const string

ExitGames.Client.Photon.Chat.ChatPeer.NameServerHttp = "http://ns.exitgamescloud.com:80/photon/n"

Name Server for HTTP connections to the **Photon** Cloud. Includes prefix and port.

## **Property Documentation**

#### string

ExitGames.Client.Photon.Chat.ChatPeer.NameServerAddress get



Name Server Address for **Photon** Cloud (based on current protocol). You can use the default values and usually won't have to set this value.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page Related		Pages Module		les	Classes	Files	
Class List	ist Class Index		Class Hierarchy		Class Members		
ExitGames Client Photon Chat ChatUserStatus							
Public Attributes   List of all members  ExitGames.Client.Photon.Chat.ChatUserStatus  Class Reference							

Contains commonly used status values for SetOnlineStatus. You can define your own.  $\underline{\mathsf{More}...}$ 

#### **Public Attributes**

const int Offline = 0

(0) Offline. More...

const int Invisible = 1

(1) Be invisible to everyone. Sends no message. More...

const int Online = 2

(2) Online and available. More...

const int Away = 3

(3) Online but not available. More...

const int DND = 4

(4) Do not disturb. More...

const int LFG = 5

(5) Looking For Game/Group. Could be used when you want to be invited or do matchmaking. More...

const int **Playing** = 6

(6) Could be used when in a room, playing. More...

### **Detailed Description**

Contains commonly used status values for SetOnlineStatus. You can define your own.

While "online" (value 2 and up), the status message will be sent to anyone who has you on his friend list.

Define custom online status values as you like with these rules: 0: Means "offline". It will be used when you are not connected. In this status, there is no status message. 1: Means "invisible" and is sent to friends as "offline". They see status 0, no message but you can chat. 2: And any higher value will be treated as "online". Status can be set.

#### Member Data Documentation

## const int ExitGames.Client.Photon.Chat.ChatUserStatus.Away = 3

(3) Online but not available.

#### const int ExitGames.Client.Photon.Chat.ChatUserStatus.DND = 4

(4) Do not disturb.

#### const int

#### ExitGames.Client.Photon.Chat.ChatUserStatus.Invisible = 1

(1) Be invisible to everyone. Sends no message.

#### const int ExitGames.Client.Photon.Chat.ChatUserStatus.LFG = 5

(5) Looking For Game/Group. Could be used when you want to be invited or do matchmaking.

## const int ExitGames.Client.Photon.Chat.ChatUserStatus.Offline = 0

(0) Offline.

## const int ExitGames.Client.Photon.Chat.ChatUserStatus.Online = 2

(2) Online and available.

const int ExitGames.Client.Photon.Chat.ChatUserStatus.Playing = 6

(6) Could be used when in a room, playing.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



## Photon Unity Networking

v1.81

Main Page Related		l Pages	Pages Module		Classes	Files		
Class List	ass List Class Index		Class Hierarchy		ss Members			
ExitGames \rangle Client \rangle Photon \rangle Chat \rangle ErrorCode \rangle								
	Public Attributes   List of all members							
ExitGames.Client.Photon.Chat.ErrorCode Class Reference								

**ErrorCode** defines the default codes associated with **Photon** client/server communication. <u>More...</u>

#### **Public Attributes**

#### const int Ok = 0

(0) is always "OK", anything else an error or specific situation. More...

#### const int OperationNotAllowedInCurrentState = -3

(-3) Operation can't be executed yet (e.g. OpJoin can't be called before being authenticated, RaiseEvent cant be used before getting into a room). More...

#### const int InvalidOperationCode = -2

(-2) The operation you called is not implemented on the server (application) you connect to. Make sure you run the fitting applications. <u>More...</u>

#### const int InternalServerError = -1

(-1) Something went wrong in the server. Try to reproduce and contact Exit Games. More...

#### const int **InvalidAuthentication** = 0x7FFF

(32767) Authentication failed. Possible cause: Appld is unknown to **Photon** (in cloud service). <u>More...</u>

#### const int GameldAlreadyExists = 0x7FFF - 1

(32766) Gameld (name) already in use (can't create another). Change name. More...

#### const int GameFull = 0x7FFF - 2

(32765) Game is full. This rarely happens when some player joined the room before your join completed. More...

#### const int **GameClosed** = 0x7FFF - 3

(32764) Game is closed and can't be joined. Join another game. More...

#### const int **ServerFull** = 0x7FFF - 5

(32762) Not in use currently. More...

#### const int **UserBlocked** = 0x7FFF - 6

(32761) Not in use currently. More...

#### const int **NoRandomMatchFound** = 0x7FFF - 7

(32760) Random matchmaking only succeeds if a room exists thats neither closed nor full. Repeat in a few seconds or create a new room. More...

#### const int GameDoesNotExist = 0x7FFF - 9

(32758) Join can fail if the room (name) is not existing (anymore). This can happen when players leave while you join. More...

#### const int MaxCcuReached = 0x7FFF - 10

(32757) Authorization on the **Photon** Cloud failed becaus the concurrent users (CCU) limit of the app's subscription is reached. More...

### const int **InvalidRegion** = 0x7FFF - 11

(32756) Authorization on the **Photon** Cloud failed because the app's subscription does not allow to use a particular region's server. <u>More...</u>

#### const int CustomAuthenticationFailed = 0x7FFF - 12

(32755) Custom Authentication of the user failed due to setup reasons (see Cloud Dashboard) or the provided user data (like username or token). Check error message for details. More...

# **Detailed Description**

**ErrorCode** defines the default codes associated with **Photon** client/server communication.

### Member Data Documentation

#### const int

ExitGames.Client.Photon.Chat.ErrorCode.CustomAuthenticationFi = 0x7FFF - 12

(32755) Custom Authentication of the user failed due to setup reasons (see Cloud Dashboard) or the provided user data (like username or tok Check error message for details.

const int ExitGames.Client.Photon.Chat.ErrorCode.GameClosed = 0x7FFF - 3

(32764) Game is closed and can't be joined. Join another game.

#### const int

ExitGames.Client.Photon.Chat.ErrorCode.GameDoesNotExist = 0x7FFF - 9

(32758) Join can fail if the room (name) is not existing (anymore). This can happen when players leave while you join.

const int ExitGames.Client.Photon.Chat.ErrorCode.GameFull = 0x7FFF - 2

(32765) Game is full. This rarely happens when some player joined the room before your join completed.

#### const int

ExitGames.Client.Photon.Chat.ErrorCode.GameIdAlreadyExists

#### = 0x7FFF - 1

(32766) Gameld (name) already in use (can't create another). Change name.

#### const int

ExitGames.Client.Photon.Chat.ErrorCode.InternalServerError = -1

(-1) Something went wrong in the server. Try to reproduce and contact Exit Games.

#### const int

ExitGames.Client.Photon.Chat.ErrorCode.InvalidAuthentication = 0x7FFF

(32767) Authentication failed. Possible cause: Appld is unknown to **Photon** (in cloud service).

#### const int

ExitGames.Client.Photon.Chat.ErrorCode.InvalidOperationCode = -2

(-2) The operation you called is not implemented on the server (application) you connect to. Make sure you run the fitting applications.

#### const int

ExitGames.Client.Photon.Chat.ErrorCode.InvalidRegion = 0x7FFF - 11

(32756) Authorization on the **Photon** Cloud failed because the app's subscription does not allow to use a particular region's server.

Some subscription plans for the **Photon** Cloud are region-bound. Servers of other regions can't be used then. Check your master server address and compare it with your **Photon** Cloud Dashboard's info. <a href="https://cloud.photonengine.com/dashboard">https://cloud.photonengine.com/dashboard</a>

OpAuthorize is part of connection workflow but only on the **Photon** Cloud, this error can happen. Self-hosted **Photon** servers with a CCU limited license won't let a client connect at all.

# const int ExitGames.Client.Photon.Chat.ErrorCode.MaxCcuReached = 0x7FFF - 10

(32757) Authorization on the **Photon** Cloud failed becaus the concurrent users (CCU) limit of the app's subscription is reached.

Unless you have a plan with "CCU Burst", clients might fail the authentication step during connect. Affected client are unable to call operations. Please note that players who end a game and return to the master server will disconnect and re-connect, which means that they just played and are rejected in the next minute / re-connect. This is a temporary measure. Once the CCU is below the limit, players will be able to connect an play again.

OpAuthorize is part of connection workflow but only on the **Photon** Cloud, this error can happen. Self-hosted **Photon** servers with a CCU limited license won't let a client connect at all.

#### const int

ExitGames.Client.Photon.Chat.ErrorCode.NoRandomMatchFound = 0x7FFF - 7

(32760) Random matchmaking only succeeds if a room exists thats neither closed nor full. Repeat in a few seconds or create a new room.

#### const int ExitGames.Client.Photon.Chat.ErrorCode.Ok = 0

(0) is always "OK", anything else an error or specific situation.

#### const int

ExitGames.Client.Photon.Chat.ErrorCode.OperationNotAllowedInC = -3

(-3) Operation can't be executed yet (e.g. OpJoin can't be called before authenticated, RaiseEvent cant be used before getting into a room).

Before you call any operations on the Cloud servers, the automated cliemust complete its authorization. In PUN, wait until State is: JoinedLobb AutoJoinLobby = true) or ConnectedToMaster (AutoJoinLobby = false)

const int ExitGames.Client.Photon.Chat.ErrorCode.ServerFull = 0x7FFF - 5

(32762) Not in use currently.

const int ExitGames.Client.Photon.Chat.ErrorCode.UserBlocked = 0x7FFF - 6

(32761) Not in use currently.

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	е	Related Pages Modules		les	Classes	Files	
Class List	Cla	ss Index	Class Hierarchy Cla		Clas	ss Members	
ExitGames	Client	Photon	Chat	<b>IChatClien</b>	tListen	er	
				Public N	Jemher	Functions I List (	of all members

# ExitGames.Client.Photon.Chat.IChatClientListen Interface Reference

Callback interface for **Chat** client side. Contains callback methods to notify your app about updates. Must be provided to new **ChatClient** in constructor More...

### **Public Member Functions**

### void **DebugReturn** (DebugLevel level, string message)

All debug output of the library will be reported through this method. Print it or put it in a buffer to use it on-screen. More...

#### void OnDisconnected ()

Disconnection happened. More...

### void OnConnected ()

Client is connected now. More...

### void OnChatStateChange (ChatState state)

The **ChatClient**'s state changed. Usually, OnConnected and OnDisconnected are the callbacks to react to. More...

# void OnGetMessages (string channelName, string[] senders, object[] messages)

Notifies app that client got new messages from server Number of senders is equal to number of messages in 'messages'. Sender with number '0' corresponds to message with number '0', sender with number '1' corresponds to message with number '1' and so on More...

# void OnPrivateMessage (string sender, object message, string channelName)

Notifies client about private message More...

### void OnSubscribed (string[] channels, bool[] results)

Result of Subscribe operation. Returns subscription result for every requested channel name. <u>More...</u>

### void OnUnsubscribed (string[] channels)

Result of Unsubscribe operation. Returns for channel name if the channel is now unsubscribed. More...

# void **OnStatusUpdate** (string user, int status, bool gotMessage, object message)

New status of another user (you get updates for users set in your friends list). More...

# **Detailed Description**

Callback interface for **Chat** client side. Contains callback methods to notify your app about updates. Must be provided to new **ChatClient** in constructor

## **Member Function Documentation**

#### void

ExitGames.Client.Photon.Chat.IChatClientListener.DebugReturn (

)

All debug output of the library will be reported through this method. Prin to use it on-screen.

#### **Parameters**

**level** DebugLevel (severity) of the message. **message** Debug text. Print to System.Console or screen.

#### void

ExitGames.Client.Photon.Chat.IChatClientListener.OnChatStateCh

The **ChatClient**'s state changed. Usually, OnConnected and OnDiscon to react to.

#### **Parameters**

state The new state.

#### void

ExitGames.Client.Photon.Chat.IChatClientListener.OnConnected (

**Client** is connected now.

Clients have to be connected before they can send their state, subscribe to channels and send any messages.

#### void

#### ExitGames.Client.Photon.Chat.IChatClientListener.OnDisconnecte

Disconnection happened.

#### void

ExitGames.Client.Photon.Chat.IChatClientListener.OnGetMessage

Notifies app that client got new messages from server Number of sender messages in 'messages'. Sender with number '0' corresponds to messages' sender with number '1' corresponds to message with number '1' and so

#### **Parameters**

channelName channel from where messages came

**senders** list of users who sent messages

messages list of messages it self

#### void

ExitGames.Client.Photon.Chat.IChatClientListener.OnPrivateMess

Notifies client about private message

#### **Parameters**

**sender** user who sent this message

message message it self

channelName channelName for private messages (messages you

a channel per target username)

#### void

### ExitGames.Client.Photon.Chat.IChatClientListener.OnStatusUpdat

New status of another user (you get updates for users set in your friend

#### **Parameters**

**user** Name of the user.

**status** New status of that user.

gotMessage True if the status contains a message you should car

status update does not include a message (keep any

**message** Message that user set.

#### void

ExitGames.Client.Photon.Chat.IChatClientListener.OnSubscribed (

Result of Subscribe operation. Returns subscription result for every req name.

If multiple channels sent in Subscribe operation, OnSubscribed may be times, each call with part of sent array or with single channel in "channe Calls order and order of channels in "channels" parameter may differ from the channels in "channels" parameter of Subscribe operation.

#### **Parameters**

**channels** Array of channel names.

results Per channel result if subscribed.

#### void

ExitGames.Client.Photon.Chat.IChatClientListener.OnUnsubscribe

Result of Unsubscribe operation. Returns for channel name if the channunsubscribed.

If multiple channels sent in Unsubscribe operation, OnUnsubscribed matimes, each call with part of sent array or with single channel in "channel order and order of channels in "channels" parameter may differ from or "channels" parameter of Unsubscribe operation.

#### **Parameters**

**channels** Array of channel names that are no longer subscribed.

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	Main Page Related		Pages Modules		Classes	Files	
Class List	Class Index	Class Hi	ierarchy Class M		ss Members		
ExitGames	ExitGames Client Photon Chat ParameterCode						
_	-			Public	: Attributes   List	of all members	
ExitGames.Client.Photon.Chat.ParameterCode							

# ExitGames.Client.Photon.Chat.ParameterCode Class Reference

### **Public Attributes**

const byte **ApplicationId** = 224

const byte **Secret** = 221

(221) Internally used to establish encryption More...

const byte **AppVersion** = 220

const byte **ClientAuthenticationType** = 217

(217) This key's (byte) value defines the target custom authentication type/service the client connects with. Used in OpAuthenticate More...

const byte **ClientAuthenticationParams** = 216

(216) This key's (string) value provides parameters sent to the custom authentication type/service the client connects with. Used in OpAuthenticate More...

const byte ClientAuthenticationData = 214

(214) This key's (string or byte[]) value provides parameters sent to the custom authentication service setup in **Photon** Dashboard. Used in OpAuthenticate More...

const byte Region = 210

(210) Used for region values in OpAuth and OpGetRegions. More...

const byte Address = 230

(230) Address of a (game) server to use. More...

const byte UserId = 225

(225) User's ID More ...

## Member Data Documentation

#### const byte

ExitGames.Client.Photon.Chat.ParameterCode.Address = 230

(230) Address of a (game) server to use.

#### const byte

ExitGames.Client.Photon.Chat.ParameterCode.ApplicationId = 224

#### const byte

ExitGames.Client.Photon.Chat.ParameterCode.AppVersion = 220

#### const byte

ExitGames.Client.Photon.Chat.ParameterCode.ClientAuthenticatio = 214

(214) This key's (string or byte[]) value provides parameters sent to the custom authentication service setup in **Photon** Dashboard. Used in OpAuthenticate

## const byte

ExitGames.Client.Photon.Chat.ParameterCode.ClientAuthenticatio = 216

(216) This key's (string) value provides parameters sent to the custom authentication type/service the client connects with. Used in OpAuthent

### const byte

# ExitGames.Client.Photon.Chat.ParameterCode.ClientAuthenticatio = 217

(217) This key's (byte) value defines the target custom authentication type/service the client connects with. Used in OpAuthenticate

### const byte

ExitGames.Client.Photon.Chat.ParameterCode.Region = 210

(210) Used for region values in OpAuth and OpGetRegions.

const byte ExitGames.Client.Photon.Chat.ParameterCode.Secret = 221

(221) Internally used to establish encryption

const byte ExitGames.Client.Photon.Chat.ParameterCode.UserId = 225

(225) User's ID

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related Pages		Modules	Classes	Files
Package Functions					
Package I	Pho	oton		Class	es   <u>Typedefs</u>

## Classes

#### class MonoBehaviour

This class adds the property photonView, while logging a warning when your game still uses the networkView. More...

#### class PunBehaviour

This class provides a .photonView and all callbacks/events that PUN can call. Override the events/methods you want to use. More...

# Typedefs

using **Hashtable** = ExitGames.Client.Photon.Hashtable

# **Typedef Documentation**

using Photon.Hashtable = typedef ExitGames.Client.Photon.Hashtable

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Related	Related Pages		les	Classes	Files	
Class List	Class Index	Class Hi	erarchy	Class Members			
Photon MonoBehaviour							
Properties   List of all members							
Photon.MonoBehaviour Class Reference							

This class adds the property photonView, while logging a warning when your game still uses the networkView. <u>More...</u>

Inherits MonoBehaviour.

Inherited by Photon.PunBehaviour, and PhotonView.

# **Properties**

## PhotonView photonView [get]

A cached reference to a **PhotonView** on this GameObject. <u>More...</u>

## new PhotonView networkView [get]

This property is only here to notify developers when they use the outdated value. More...

# **Detailed Description**

This class adds the property photonView, while logging a warning when your game still uses the networkView.

## **Property Documentation**

#### new PhotonView Photon.MonoBehaviour.networkView

get

This property is only here to notify developers when they use the outdated value.

If Unity 5.x logs a compiler warning "Use the new keyword if hiding was intended" or "The new keyword is not required", you may suffer from an Editor issue. Try to modify networkView with a if-def condition:

#if UNITY\_EDITOR new #endif public PhotonView networkView

### PhotonView Photon.MonoBehaviour.photonView

get

A cached reference to a **PhotonView** on this GameObject.

If you intend to work with a **PhotonView** in a script, it's usually easier to write this.photonView.

If you intend to remove the **PhotonView** component from the GameObject but keep this **Photon.MonoBehaviour**, avoid this reference or modify this code to use PhotonView.Get(obj) instead.

Online Documentation - Dashboard - Support Forum



Main Page	Related Pages		Modules	Classes	Files
Package Functions					
Package I	Jni	tyEngine			Namespaces

# Namespaces

## package SceneManagement

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	R	elated Pages	Modules	Classes	Files		
Package Functio	ns						
UnityEngine Sc	UnityEngine SceneManagement						
Package UnityEngi	ne	.SceneMar	nagement		Classes		

## Classes

class SceneManager

Minimal implementation of the **SceneManager** for older Unity, up to v5.2. More...

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Main Page Related		ed Pages Module		les	Classes	Files
Class List	Class Index		Class Hierarchy		Clas	ss Members	
UnityEngine	Sc	eneManagem	ent Scer	neManager	$\rightarrow$		
			·		Sta	atic Public Memb	oer Functions
						<u>List</u>	of all members
		_			_		_

# UnityEngine.SceneManagement.SceneManager Class Reference

Minimal implementation of the **SceneManager** for older Unity, up to v5.2. More...

# Static Public Member Functions

static void LoadScene (string name)

static void LoadScene (int buildIndex)

# **Detailed Description**

Minimal implementation of the **SceneManager** for older Unity, up to v5.2.

## **Member Function Documentation**

static void

UnityEngine.SceneManagement.SceneManager.LoadScene (string

static void

UnityEngine.SceneManagement.SceneManager.LoadScene (int bu

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Related Pages Modules		les	Classes	Files	
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	
				Public	: Attributes   List	of all members

# **ActorProperties Class Reference**

Class for constants. These (byte) values define "well known" properties for an Actor / Player. Pun uses these constants internally. More...

## **Public Attributes**

const byte **PlayerName** = 255

(255) Name of a player/actor. More...

const byte **Islnactive** = 254

(254) Tells you if the player is currently in this game

(getting events live). More...

const byte **UserId** = 253

(253) UserId of the player. Sent when room gets created

with RoomOptions.PublishUserId = true. More...

## **Detailed Description**

Class for constants. These (byte) values define "well known" properties for an Actor / Player. Pun uses these constants internally.

"Custom properties" have to use a string-type as key. They can be assigned at will.

## Member Data Documentation

#### const byte ActorProperties.IsInactive = 254

(254) Tells you if the player is currently in this game (getting events live).

A server-set value for async games, where players can leave the game and return later.

### **const byte ActorProperties.PlayerName = 255**

(255) Name of a player/actor.

## const byte ActorProperties.UserId = 253

(253) UserId of the player. Sent when room gets created with **RoomOptions.PublishUserId** = true.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



## Photon Unity Networking

v1.81

Main Page	Related Pages		Modules		Classes	Files
Class List C	Class Index	Class Hi	erarchy	Clas	ss Members	

## AuthenticationValues Class Reference

<u>Public Member Functions</u> | <u>Properties</u> | <u>List of all members</u>

Container for user authentication in **Photon**. Set AuthValues before you connect - all else is handled. <u>More...</u>

## **Public Member Functions**

## **AuthenticationValues ()**

Creates empty auth values without any info. More...

#### **AuthenticationValues** (string userId)

Creates minimal info about the user. If this is authenticated or not, depends on the set AuthType. More...

#### virtual void **SetAuthPostData** (string stringData)

Sets the data to be passed-on to the auth service via POST. More...

#### virtual void SetAuthPostData (byte[] byteData)

Sets the data to be passed-on to the auth service via POST. More...

## virtual void AddAuthParameter (string key, string value)

Adds a key-value pair to the get-parameters used for Custom Auth. More...

### override string ToString ()

## **Properties**

## CustomAuthenticationType AuthType [get, set]

The type of custom authentication provider that should be used. Currently only "Custom" or "None" (turns this off). More...

#### string AuthGetParameters [get, set]

This string must contain any (http get) parameters expected by the used authentication service. By default, username and token. More...

#### object AuthPostData [get, set]

Data to be passed-on to the auth service via POST. Default: null (not sent). Either string or byte[] (see setters). More...

### string Token [get, set]

After initial authentication, **Photon** provides a token for this client / user, which is subsequently used as (cached) validation. More...

## string UserId [get, set]

The UserId should be a unique identifier per user. This is for finding friends, etc.. <u>More...</u>

## **Detailed Description**

Container for user authentication in **Photon**. Set AuthValues before you connect - all else is handled.

On **Photon**, user authentication is optional but can be useful in many cases. If you want to FindFriends, a unique ID per user is very practical.

There are basically three options for user authentification: None at all, the client sets some Userld or you can use some account web-service to authenticate a user (and set the Userld server-side).

Custom Authentication lets you verify end-users by some kind of login or token. It sends those values to **Photon** which will verify them before granting access or disconnecting the client.

The AuthValues are sent in OpAuthenticate when you connect, so they must be set before you connect. Should you not set any AuthValues, PUN will create them and set the playerName as userId in them. If the AuthValues.userId is null or empty when it's sent to the server, then the **Photon** Server assigns a userId!

The **Photon** Cloud Dashboard will let you enable this feature and set important server values for it.

https://www.photonengine.com/dashboard

## Constructor & Destructor Documentation

## AuthenticationValues. AuthenticationValues ()

Creates empty auth values without any info.

## AuthenticationValues.AuthenticationValues (string userId)

Creates minimal info about the user. If this is authenticated or not, depends on the set AuthType.

#### **Parameters**

userId Some UserId to set in Photon.

### Member Function Documentation

#### virtual void AuthenticationValues.AddAuthParameter

(string key, string value

virtual

Adds a key-value pair to the get-parameters used for Custom Auth.

This method does uri-encoding for you.

#### **Parameters**

**key** Key for the value to set.

value Some value relevant for Custom Authentication.

#### virtual void

AuthenticationValues.SetAuthPostData (string stringData)



Sets the data to be passed-on to the auth service via POST.

#### **Parameters**

**stringData** String data to be used in the body of the POST request. Null or empty string will set AuthPostData to null.

#### virtual void

AuthenticationValues.SetAuthPostData (byte[] byteData)



Sets the data to be passed-on to the auth service via POST.

#### **Parameters**

byteData Binary token / auth-data to pass on.

override string AuthenticationValues.ToString ( )

## **Property Documentation**

#### string AuthenticationValues.AuthGetParameters

get set

This string must contain any (http get) parameters expected by the used authentication service. By default, username and token.

Standard http get parameters are used here and passed on to the service that's defined in the server (**Photon** Cloud Dashboard).

#### object AuthenticationValues.AuthPostData





Data to be passed-on to the auth service via POST. Default: null (not sent). Either string or byte[] (see setters).

## CustomAuthenticationType AuthenticationValues.AuthType





The type of custom authentication provider that should be used. Currently only "Custom" or "None" (turns this off).

## string AuthenticationValues.Token





After initial authentication, **Photon** provides a token for this client / user, which is subsequently used as (cached) validation.

## string AuthenticationValues.UserId





The UserId should be a unique identifier per user. This is for finding friends, etc..

See remarks of AuthValues for info about how this is set and used.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Attributes | List of all members

## **EncryptionDataParameters Class Reference**

## **Public Attributes**

const byte Mode = 0

Key for encryption mode More...

const byte **Secret1** = 1

Key for first secret More...

const byte **Secret2** = 2

Key for second secret More...

## Member Data Documentation

const byte EncryptionDataParameters.Mode = 0

Key for encryption mode

const byte EncryptionDataParameters.Secret1 = 1

Key for first secret

const byte EncryptionDataParameters.Secret2 = 2

Key for second secret

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



## Photon Unity Networking

v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Attributes | List of all members

# **ErrorCode Class Reference**

**ErrorCode** defines the default codes associated with **Photon** client/server communication. More...

## **Public Attributes**

#### const int Ok = 0

(0) is always "OK", anything else an error or specific situation. More...

#### const int OperationNotAllowedInCurrentState = -3

(-3) Operation can't be executed yet (e.g. OpJoin can't be called before being authenticated, RaiseEvent cant be used before getting into a room). More...

#### const int InvalidOperationCode = -2

(-2) The operation you called is not implemented on the server (application) you connect to. Make sure you run the fitting applications. <u>More...</u>

#### const int **InvalidOperation** = -2

(-2) The operation you called could not be executed on the server. More...

#### const int **InternalServerError** = -1

(-1) Something went wrong in the server. Try to reproduce and contact Exit Games. <u>More...</u>

#### const int InvalidAuthentication = 0x7FFF

(32767) Authentication failed. Possible cause: Appld is unknown to **Photon** (in cloud service). More...

## const int **GameIdAlreadyExists** = 0x7FFF - 1

(32766) GameId (name) already in use (can't create another). Change name. More...

#### const int GameFull = 0x7FFF - 2

(32765) Game is full. This rarely happens when some player joined the room before your join completed. More...

#### const int GameClosed = 0x7FFF - 3

(32764) Game is closed and can't be joined. Join another game. More...

#### const int AlreadyMatched = 0x7FFF - 4

#### const int ServerFull = 0x7FFF - 5

(32762) Not in use currently. More...

#### const int UserBlocked = 0x7FFF - 6

(32761) Not in use currently. More...

#### const int NoRandomMatchFound = 0x7FFF - 7

(32760) Random matchmaking only succeeds if a room exists thats neither closed nor full. Repeat in a few seconds or create a new room. More...

#### const int **GameDoesNotExist** = 0x7FFF - 9

(32758) Join can fail if the room (name) is not existing (anymore). This can happen when players leave while you join. More...

#### const int MaxCcuReached = 0x7FFF - 10

(32757) Authorization on the **Photon** Cloud failed becaus the concurrent users (CCU) limit of the app's subscription is reached. More...

## const int InvalidRegion = 0x7FFF - 11

(32756) Authorization on the **Photon** Cloud failed because the app's subscription does not allow to use a particular region's server. More...

#### const int CustomAuthenticationFailed = 0x7FFF - 12

(32755) Custom Authentication of the user failed due to setup reasons (see Cloud Dashboard) or the provided user data (like username or token). Check error message for details. More...

#### const int **AuthenticationTicketExpired** = 0x7FF1

(32753) The Authentication ticket expired. Usually, this is refreshed behind the scenes. Connect (and authorize) again. More...

#### const int PluginReportedError = 0x7FFF - 15

(32752) A server-side plugin (or webhook) failed to execute and reported an error. Check the OperationResponse.DebugMessage. More...

#### const int **PluginMismatch** = 0x7FFF - 16

(32751) CreateRoom/JoinRoom/Join operation fails if expected plugin does not correspond to loaded one. More...

#### const int **JoinFailedPeerAlreadyJoined** = 32750

(32750) for join requests. Indicates the current peer already called join and is joined to the room. More...

#### const int **JoinFailedFoundInactiveJoiner** = 32749

(32749) for join requests. Indicates the list of InactiveActors already contains an actor with the requested ActorNr or UserId. More...

## const int **JoinFailedWithRejoinerNotFound** = 32748

(32748) for join requests. Indicates the list of Actors (active and inactive) did not contain an actor with the requested ActorNr or UserId. More...

### const int **JoinFailedFoundExcludedUserId** = 32747

(32747) for join requests. Note: for future use - Indicates the requested UserId was found in the ExcludedList. More...

#### const int **JoinFailedFoundActiveJoiner** = 32746

(32746) for join requests. Indicates the list of ActiveActors

already contains an actor with the requested ActorNr or UserId. More...

#### const int **HttpLimitReached** = 32745

(32745) for SetProerties and Raisevent (if flag HttpForward is true) requests. Indicates the maximum allowd http requests per minute was reached. More...

#### const int **ExternalHttpCallFailed** = 32744

(32744) for WebRpc requests. Indicates the the call to the external service failed. More...

#### const int **SlotError** = 32742

(32742) Server error during matchmaking with slot reservation. E.g. the reserved slots can not exceed MaxPlayers. More...

#### const int **InvalidEncryptionParameters** = 32741

(32741) Server will react with this error if invalid encryption parameters provided by token <u>More...</u>

## **Detailed Description**

**ErrorCode** defines the default codes associated with **Photon** client/server communication.

## Member Data Documentation

#### const int ErrorCode.AlreadyMatched = 0x7FFF - 4

### const int ErrorCode.AuthenticationTicketExpired = 0x7FF1

(32753) The Authentication ticket expired. Usually, this is refreshed behind the scenes. Connect (and authorize) again.

#### const int ErrorCode.CustomAuthenticationFailed = 0x7FFF - 12

(32755) Custom Authentication of the user failed due to setup reasons (see Cloud Dashboard) or the provided user data (like username or token). Check error message for details.

## const int ErrorCode.ExternalHttpCallFailed = 32744

(32744) for WebRpc requests. Indicates the the call to the external service failed.

#### const int ErrorCode.GameClosed = 0x7FFF - 3

(32764) Game is closed and can't be joined. Join another game.

#### const int ErrorCode.GameDoesNotExist = 0x7FFF - 9

(32758) Join can fail if the room (name) is not existing (anymore). This can happen when players leave while you join.

#### const int ErrorCode.GameFull = 0x7FFF - 2

(32765) Game is full. This rarely happens when some player joined the room before your join completed.

#### const int ErrorCode.GameIdAlreadyExists = 0x7FFF - 1

(32766) GameId (name) already in use (can't create another). Change name.

#### const int ErrorCode.HttpLimitReached = 32745

(32745) for SetProerties and Raisevent (if flag HttpForward is true) requests. Indicates the maximum allowd http requests per minute was reached.

#### const int ErrorCode.InternalServerError = -1

(-1) Something went wrong in the server. Try to reproduce and contact Exit Games.

#### const int ErrorCode.InvalidAuthentication = 0x7FFF

(32767) Authentication failed. Possible cause: Appld is unknown to **Photon** (in cloud service).

## const int ErrorCode.InvalidEncryptionParameters = 32741

(32741) Server will react with this error if invalid encryption parameters provided by token

#### const int ErrorCode.InvalidOperation = -2

(-2) The operation you called could not be executed on the server.

Make sure you are connected to the server you expect.

This code is used in several cases: The arguments/parameters of the operation might be out of range, missing entirely or conflicting. The operation you called is not implemented on the server (application). Server-side plugins affect the available operations.

## const int ErrorCode.InvalidOperationCode = -2

(-2) The operation you called is not implemented on the server (application) you connect to. Make sure you run the fitting applications.

#### const int ErrorCode.InvalidRegion = 0x7FFF - 11

(32756) Authorization on the **Photon** Cloud failed because the app's subscription does not allow to use a particular region's server.

Some subscription plans for the **Photon** Cloud are region-bound. Servers of other regions can't be used then. Check your master server address and compare it with your **Photon** Cloud Dashboard's info. <a href="https://cloud.photonengine.com/dashboard">https://cloud.photonengine.com/dashboard</a>

OpAuthorize is part of connection workflow but only on the **Photon** Cloud, this error can happen. Self-hosted **Photon** servers with a CCU limited license won't let a client connect at all.

#### const int ErrorCode.JoinFailedFoundActiveJoiner = 32746

(32746) for join requests. Indicates the list of ActiveActors already contains an actor with the requested ActorNr or UserId.

#### const int ErrorCode.JoinFailedFoundExcludedUserId = 32747

(32747) for join requests. Note: for future use - Indicates the requested UserId was found in the ExcludedList.

#### const int ErrorCode.JoinFailedFoundInactiveJoiner = 32749

(32749) for join requests. Indicates the list of InactiveActors already contains an actor with the requested ActorNr or UserId.

#### const int ErrorCode.JoinFailedPeerAlreadyJoined = 32750

(32750) for join requests. Indicates the current peer already called join and is joined to the room.

#### const int ErrorCode.JoinFailedWithRejoinerNotFound = 32748

(32748) for join requests. Indicates the list of Actors (active and inactive) did not contain an actor with the requested ActorNr or UserId.

#### const int ErrorCode.MaxCcuReached = 0x7FFF - 10

(32757) Authorization on the **Photon** Cloud failed becaus the concurrent users (CCU) limit of the app's subscription is reached.

Unless you have a plan with "CCU Burst", clients might fail the authentication step during connect. Affected client are unable to call operations. Please note that players who end a game and return to the master server will disconnect and re-connect, which means that they just played and are rejected in the next minute / re-connect. This is a temporary measure. Once the CCU is below the limit, players will be able to connect an play again.

OpAuthorize is part of connection workflow but only on the **Photon** Cloud, this error can happen. Self-hosted **Photon** servers with a CCU limited license won't let a client connect at all.

#### const int ErrorCode.NoRandomMatchFound = 0x7FFF - 7

(32760) Random matchmaking only succeeds if a room exists thats neither closed nor full. Repeat in a few seconds or create a new room.

#### const int ErrorCode.Ok = 0

(0) is always "OK", anything else an error or specific situation.

#### const int ErrorCode.OperationNotAllowedInCurrentState = -3

(-3) Operation can't be executed yet (e.g. OpJoin can't be called before being authenticated, RaiseEvent cant be used before getting into a room).

Before you call any operations on the Cloud servers, the automated client workflow must complete its authorization. In PUN, wait until State is: JoinedLobby (with AutoJoinLobby = true) or ConnectedToMaster (AutoJoinLobby = false)

## const int ErrorCode.PluginMismatch = 0x7FFF - 16

(32751) CreateRoom/JoinRoom/Join operation fails if expected plugin does not correspond to loaded one.

## const int ErrorCode.PluginReportedError = 0x7FFF - 15

(32752) A server-side plugin (or webhook) failed to execute and

reported an error. Check the OperationResponse.DebugMessage.

#### const int ErrorCode.ServerFull = 0x7FFF - 5

(32762) Not in use currently.

#### const int ErrorCode.SlotError = 32742

(32742) Server error during matchmaking with slot reservation. E.g. the reserved slots can not exceed MaxPlayers.

#### const int ErrorCode.UserBlocked = 0x7FFF - 6

(32761) Not in use currently.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Index Class Hie		Clas	ss Members	

Public Attributes | List of all members

# **EventCode Class Reference**

Class for constants. These values are for events defined by **Photon** Loadbalancing. Pun uses these constants internally. <u>More...</u>

## **Public Attributes**

const byte **GameList** = 230

(230) Initial list of RoomInfos (in lobby on Master) More...

const byte **GameListUpdate** = 229

(229) Update of RoomInfos to be merged into "initial" list (in lobby on Master) More...

const byte **QueueState** = 228

(228) Currently not used. State of queueing in case of server-full More...

const byte Match = 227

(227) Currently not used. Event for matchmaking More...

const byte **AppStats** = 226

(226) Event with stats about this application (players, rooms, etc) More...

const byte **LobbyStats** = 224

(224) This event provides a list of lobbies with their player and game counts. <u>More...</u>

const byte AzureNodeInfo = 210

(210) Internally used in case of hosting by Azure More...

const byte **Join** = (byte)255

(255) Event Join: someone joined the game. The new actorNumber is provided as well as the properties of that actor (if set in OpJoin). More...

const byte **Leave** = (byte)254

(254) Event Leave: The player who left the game can be identified by the actorNumber. More...

## const byte **PropertiesChanged** = (byte)253

(253) When you call OpSetProperties with the broadcast option "on", this event is fired. It contains the properties being set. <u>More...</u>

#### const byte **SetProperties** = (byte)253

(253) When you call OpSetProperties with the broadcast option "on", this event is fired. It contains the properties being set. More...

#### const byte **ErrorInfo** = 251

(252) When player left game unexpected and the room has a playerTtl > 0, this event is fired to let everyone know about the timeout. More...

#### const byte **CacheSliceChanged** = 250

(250) Sent by **Photon** whent he event cache slice was changed. Done by OpRaiseEvent. More...

## const byte **AuthEvent** = 223

(223) Sent by **Photon** to update a token before it times out. More...

## **Detailed Description**

Class for constants. These values are for events defined by **Photon** Loadbalancing. Pun uses these constants internally.

They start at 255 and go DOWN. Your own in-game events can start at 0.

## Member Data Documentation

#### const byte EventCode.AppStats = 226

(226) Event with stats about this application (players, rooms, etc)

#### const byte EventCode.AuthEvent = 223

(223) Sent by **Photon** to update a token before it times out.

#### const byte EventCode.AzureNodeInfo = 210

(210) Internally used in case of hosting by Azure

## const byte EventCode.CacheSliceChanged = 250

(250) Sent by **Photon** whent he event cache slice was changed. Done by OpRaiseEvent.

## const byte EventCode.ErrorInfo = 251

(252) When player left game unexpected and the room has a playerTtl event is fired to let everyone know about the timeout.

Obsolete. Replaced by Leave. public const byte Disconnect = LiteEventCode.Disconnect;

(251) Sent by **Photon** Cloud when a plugin-call or webhook-call failed. execution on the server continues, despite the issue. Contains: **ParameterCode.Info**.

#### See also

https://doc.photonengine.com/en/realtime/current/reference/webho

#### const byte EventCode.GameList = 230

(230) Initial list of RoomInfos (in lobby on Master)

### const byte EventCode.GameListUpdate = 229

(229) Update of RoomInfos to be merged into "initial" list (in lobby on Master)

## const byte EventCode.Join = (byte)255

(255) Event Join: someone joined the game. The new actorNumber is provided as well as the properties of that actor (if set in OpJoin).

## const byte EventCode.Leave = (byte)254

(254) Event Leave: The player who left the game can be identified by the actorNumber.

## const byte EventCode.LobbyStats = 224

(224) This event provides a list of lobbies with their player and game counts.

## const byte EventCode.Match = 227

(227) Currently not used. Event for matchmaking

#### const byte EventCode.PropertiesChanged = (byte)253

(253) When you call OpSetProperties with the broadcast option "on", this event is fired. It contains the properties being set.

## const byte EventCode.QueueState = 228

(228) Currently not used. State of queueing in case of server-full

## const byte EventCode.SetProperties = (byte)253

(253) When you call OpSetProperties with the broadcast option "on", this event is fired. It contains the properties being set.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page	e F	Related	Pages	ges Modules		Classes	Files
Class List	Class	Class Index Class Hi		erarchy	Class Members		
Extensions Class Reference			5	<u>Stati</u>		atic Public Membe : Attributes   List o	•

This static class defines some useful extension methods for several existing classes (e.g. Vector3, float and others). More...

## Static Public Member Functions

static ParameterInfo[]	GetCachedParemeters (this MethodInfo mo)
static PhotonView[]	GetPhotonViewsInChildren (this UnityEngine.GameObject go)
static PhotonView	GetPhotonView (this UnityEngine.GameObject go)
static bool	AlmostEquals (this Vector3 target, Vector3 second, float sqrMagnitudePrecision) compares the squared magnitude of target - second to given float value More
static bool	AlmostEquals (this Vector2 target, Vector2 second, float sqrMagnitudePrecision) compares the squared magnitude of target - second to given float value More
static bool	AlmostEquals (this Quaternion target, Quaternion second, float maxAngle) compares the angle between target and second to given float value More
static bool	AlmostEquals (this float target, float second, float floatDiff) compares two floats and returns true of their difference is less than floatDiff More
static void	Merge (this IDictionary target, IDictionary addHash) Merges all keys from addHash into the target. Adds new keys and updates the values of existing keys in target. More
	MergeStringKeys (this IDictionary target,

#### static void IDictionary addHash)

Merges keys of type string to target Hashtable. More...

## static string **ToStringFull** (this IDictionary origin)

Helper method for debugging of IDictionary content, inlcuding type-information. Using this is not performant. More...

#### static string **ToStringFull** (this object[] data)

Helper method for debugging of object[] content. Using this is not performant. More...

#### static **Hashtable StripToStringKeys** (this IDictionary original)

This method copies all string-typed keys of the original into a new Hashtable. More...

#### static void

## StripKeysWithNullValues (this IDictionary original)

This removes all key-value pairs that have a null-reference as value. **Photon** properties are removed by setting their value to null. Changes the original passed IDictionary! More...

## static bool **Contains** (this int∏ target, int nr)

Checks if a particular integer value is in an intarray. More...

### Static Public Attributes

static Dictionary< MethodInfo, ParameterInfo[]>

ParametersOfMethods = new Dictionary<MethodInfo, ParameterInfo[]>()

### **Detailed Description**

This static class defines some useful extension methods for several existing classes (e.g. Vector3, float and others).

### **Member Function Documentation**

compares the squared magnitude of target - second to given float value

compares the squared magnitude of target - second to given float value

```
static bool
Extensions.AlmostEquals (this Quaternion target,
Quaternion second,
float maxAngle
)
```

compares the angle between target and second to given float value

```
static bool Extensions.AlmostEquals (this float target, float second, float floatDiff
) static
```

compares two floats and returns true of their difference is less than floatDiff

Checks if a particular integer value is in an int-array.

This might be useful to look up if a particular actorNumber is in the list of players of a room.

#### **Parameters**

target The array of ints to check.nr The number to lookup in target.

#### **Returns**

True if nr was found in target.

# static ParameterInfo [] Extensions.GetCachedParemeters (this MethodInfo mo) static

static PhotonView

Extensions.GetPhotonView (this UnityEngine.GameObject go)

static PhotonView []

Extensions.GetPhotonViewsInChildren (this UnityEngine.GameOb

```
static void Extensions.Merge (this IDictionary target,
IDictionary addHash
)
```

Merges all keys from addHash into the target. Adds new keys and

updates the values of existing keys in target.

#### **Parameters**

target The IDictionary to update.

addHash The IDictionary containing data to merge into target.

#### static void

Extensions.MergeStringKeys (this IDictionary target, IDictionary addHash)

static

Merges keys of type string to target Hashtable.

Does not remove keys from target (so non-string keys CAN be in target if they were before).

#### **Parameters**

target The target IDicitionary passed in plus all string-typed

keys from the addHash.

**addHash** A IDictionary that should be merged partly into target to update it.

#### static void

Extensions.StripKeysWithNullValues (this IDictionary original)

.

This removes all key-value pairs that have a null-reference as value. **Photon** properties are removed by setting their value to null. Changes original passed IDictionary!

#### **Parameters**

original The IDictionary to strip of keys with null-values.

#### static Hashtable

**Extensions.StripToStringKeys** (this IDictionary original)

static

This method copies all string-typed keys of the original into a new Hashtable.

Does not recurse (!) into hashes that might be values in the roothash. This does not modify the original.

#### **Parameters**

original The original IDictonary to get string-typed keys from.

#### Returns

New Hashtable containing only string-typed keys of the original.

### static string Extensions.ToStringFull

(this IDictionary origin)

static

Helper method for debugging of IDictionary content, inlcuding type-information. Using this is not performant.

Should only be used for debugging as necessary.

#### **Parameters**

origin Some Dictionary or Hashtable.

#### Returns

String of the content of the IDictionary.

### static string Extensions.ToStringFull (this object[] data)

static

Helper method for debugging of object[] content. Using this is not performant.

Should only be used for debugging as necessary.

#### **Parameters**

data Any object[].

### Returns

A comma-separated string containing each value's ToString().

### Member Data Documentation

Dictionary<MethodInfo, ParameterInfo[]> Extensions.ParametersOfMethods = new Dictionary<MethodInfo, ParameterInfo[]>()

static

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



## Photon Unity Networking

v1.81

Main Page		Related Pages		Modules		Classes	Files
Class List	C	lass Index	Class Hierarchy		Class Members		
Friendli Referer				P	Public M	ember Functions List (	<u>Properties</u>   of all members

Used to store info about a friend's online state and in which room he/she is. More...

### **Public Member Functions**

override string ToString ()

### **Properties**

string Name [get, set]

bool IsOnline [get, set]

string Room [get, set]

bool IsInRoom [get]

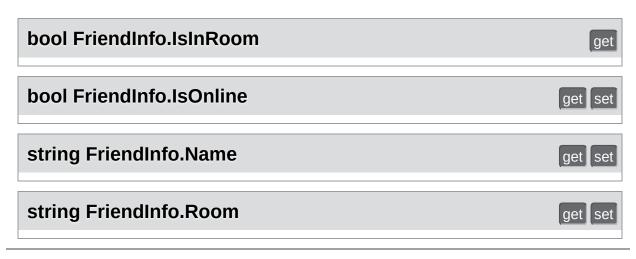
### **Detailed Description**

Used to store info about a friend's online state and in which room he/she is.

### **Member Function Documentation**

override string FriendInfo.ToString ( )

### **Property Documentation**



Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page		Related	Pages	Modules		Classes	Files
Class List	С	Class Index Class Hi		erarchy	Clas	ss Members	
					Sta	atic Public Membe	er Functions

### **GameObjectExtensions Class Reference**

Small number of extension methods that make it easier for PUN to work cross-Unity-versions. <u>More...</u>

### Static Public Member Functions

static bool **GetActive** (this GameObject target)
Unity-version-independent replacement for active GO property. More...

### **Detailed Description**

Small number of extension methods that make it easier for PUN to work cross-Unity-versions.

### **Member Function Documentation**

### static bool

**GameObjectExtensions.GetActive (this GameObject target)** 

static

Unity-version-independent replacement for active GO property.

#### **Returns**

Unity 3.5: active. Any newer Unity: activeInHierarchy.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



## Photon Unity Networking

v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	
				Public	: Attributes   List	of all members

# GamePropertyKey Class Reference

Class for constants. These (byte) values are for "well known" room/game properties used in **Photon** Loadbalancing. Pun uses these constants internally. <u>More...</u>

### **Public Attributes**

### const byte **MaxPlayers** = 255

(255) Max number of players that "fit" into this room. 0 is for "unlimited". More...

### const byte **IsVisible** = 254

(254) Makes this room listed or not in the lobby on master. More...

### const byte **IsOpen** = 253

(253) Allows more players to join a room (or not). More...

### const byte PlayerCount = 252

(252) Current count of players in the room. Used only in the lobby on master. <u>More...</u>

### const byte **Removed** = 251

(251) True if the room is to be removed from room listing (used in update to room list in lobby on master) More...

### const byte **PropsListedInLobby** = 250

(250) A list of the room properties to pass to the **RoomInfo** list in a lobby. This is used in CreateRoom, which defines this list once per room. More...

### const byte CleanupCacheOnLeave = 249

(249) Equivalent of Operation Join parameter CleanupCacheOnLeave. More...

### const byte **MasterClientId** = (byte)248

(248) Code for MasterClientId, which is synced by server. When sent as op-parameter this is (byte)203. As room property this is (byte)248. More...

const byte ExpectedUsers = (byte)247
(247) Code for ExpectedUsers in a room. Matchmaking keeps a slot open for the players with these userIDs. More...

### **Detailed Description**

Class for constants. These (byte) values are for "well known" room/game properties used in **Photon** Loadbalancing. Pun uses these constants internally.

"Custom properties" have to use a string-type as key. They can be assigned at will.

### Member Data Documentation

### const byte GamePropertyKey.CleanupCacheOnLeave = 249

(249) Equivalent of Operation Join parameter CleanupCacheOnLeave.

### const byte GamePropertyKey.ExpectedUsers = (byte)247

(247) Code for ExpectedUsers in a room. Matchmaking keeps a slot open for the players with these userIDs.

### **const byte GamePropertyKey.IsOpen = 253**

(253) Allows more players to join a room (or not).

### const byte GamePropertyKey.IsVisible = 254

(254) Makes this room listed or not in the lobby on master.

### const byte GamePropertyKey.MasterClientId = (byte)248

(248) Code for MasterClientId, which is synced by server. When sent as op-parameter this is (byte)203. As room property this is (byte)248.

Tightly related to ParameterCode.MasterClientId.

### const byte GamePropertyKey.MaxPlayers = 255

(255) Max number of players that "fit" into this room. 0 is for "unlimited".

### const byte GamePropertyKey.PlayerCount = 252

(252) Current count of players in the room. Used only in the lobby on master.

### const byte GamePropertyKey.PropsListedInLobby = 250

(250) A list of the room properties to pass to the **RoomInfo** list in a lobby. This is used in CreateRoom, which defines this list once per room.

### const byte GamePropertyKey.Removed = 251

(251) True if the room is to be removed from room listing (used in update to room list in lobby on master)

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



## Photon Unity Networking

v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Member Functions | List of all members

### HelpURL Class Reference

Empty implementation of the upcoming **HelpURL** of Unity 5.1. This one is only for compatibility of attributes. <u>More...</u>

Inherits Attribute.

### **Public Member Functions**

HelpURL (string url)

### **Detailed Description**

Empty implementation of the upcoming **HelpURL** of Unity 5.1. This one is only for compatibility of attributes.

http://feedback.unity3d.com/suggestions/override-component-documentation-slash-help-link

### Constructor & Destructor Documentation

HelpURL.HelpURL (string url)

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



## Photon Unity Networking

v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Member Functions | List of all members

# IPunPrefabPool Interface Reference

Defines all the methods that a Object Pool must implement, so that PUN can use it. More...

### **Public Member Functions**

### GameObject

**Instantiate** (string prefabld, Vector3 position, Quaternion rotation)

This is called when PUN wants to create a new instance of an entity prefab. Must return valid GameObject with **PhotonView**. More...

### void Destroy (GameObject gameObject)

This is called when PUN wants to destroy the instance of an entity prefab. More...

### **Detailed Description**

Defines all the methods that a Object Pool must implement, so that PUN can use it.

To use a Object Pool for instantiation, you can set PhotonNetwork.ObjectPool. That is used for all objects, as long as ObjectPool is not null. The pool has to return a valid non-null GameObject when PUN calls Instantiate. Also, the position and rotation must be applied.

Please note that pooled GameObjects don't get the usual Awake and Start calls. OnEnable will be called (by your pool) but the networking values are not updated yet when that happens. OnEnable will have outdated values for **PhotonView** (isMine, etc.). You might have to adjust scripts.

PUN will call OnPhotonInstantiate (see **IPunCallbacks**). This should be used to setup the re-used object with regards to networking values / ownership.

### Member Function Documentation

### void IPunPrefabPool.Destroy ( GameObject gameObject )

This is called when PUN wants to destroy the instance of an entity prefab.

A pool needs some way to find out which type of GameObject got returned via **Destroy()**. It could be a tag or name or anything similar.

#### **Parameters**

**gameObject** The instance to destroy.

```
GameObject IPunPrefabPool.Instantiate ( string prefabld, Vector3 position, Quaternion rotation )
```

This is called when PUN wants to create a new instance of an entity prefab. Must return valid GameObject with **PhotonView**.

#### **Parameters**

prefabld The id of this prefab.position The position we want the instance instantiated at.rotation The rotation we want the instance to take.

#### Returns

The newly instantiated object, or null if a prefab with *prefabld* was not found.



## Photon Unity Networking

v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Attributes | List of all members

# **OperationCode Class Reference**

Class for constants. Contains operation codes. Pun uses these constants internally.  $\underline{\text{More...}}$ 

### **Public Attributes**

const byte ExchangeKeysForEncryption = 250

const byte **Join** = 255

(255) Code for OpJoin, to get into a room. More...

const byte **AuthenticateOnce** = 231

(231) Authenticates this peer and connects to a virtual application More...

const byte **Authenticate** = 230

(230) Authenticates this peer and connects to a virtual application More...

const byte **JoinLobby** = 229

(229) Joins lobby (on master) More...

const byte **LeaveLobby** = 228

(228) Leaves lobby (on master) More...

const byte **CreateGame** = 227

(227) Creates a game (or fails if name exists) More...

const byte **JoinGame** = 226

(226) Join game (by name) More...

const byte **JoinRandomGame** = 225

(225) Joins random game (on master) More...

const byte Leave = (byte)254

(254) Code for OpLeave, to get out of a room. More...

const byte RaiseEvent = (byte)253

(253) Raise event (in a room, for other actors/players)

More...

### const byte **SetProperties** = (byte)252

(252) Set Properties (of room or actor/player) More...

### const byte **GetProperties** = (byte)251

(251) Get Properties More...

### const byte **ChangeGroups** = (byte)248

(248) Operation code to change interest groups in Rooms (Lite application and extending ones). More...

### const byte **FindFriends** = 222

(222) Request the rooms and online status for a list of friends (by name, which should be unique). More...

### const byte **GetLobbyStats** = 221

(221) Request statistics about a specific list of lobbies (their user and game count). More...

### const byte **GetRegions** = 220

(220) Get list of regional servers from a NameServer. More...

### const byte WebRpc = 219

(219) WebRpc Operation. More...

### const byte ServerSettings = 218

(218) Operation to set some server settings. Used with different parameters on various servers. <u>More...</u>

### const byte **GetGameList** = 217

(217) Get the game list matching a supplied sql filter (SqlListLobby only) More...

### **Detailed Description**

Class for constants. Contains operation codes. Pun uses these constants internally.

#### Member Data Documentation

#### const byte OperationCode.Authenticate = 230

(230) Authenticates this peer and connects to a virtual application

#### const byte OperationCode.AuthenticateOnce = 231

(231) Authenticates this peer and connects to a virtual application

#### const byte OperationCode.ChangeGroups = (byte)248

(248) Operation code to change interest groups in Rooms (Lite application and extending ones).

#### **const byte OperationCode.CreateGame = 227**

(227) Creates a game (or fails if name exists)

#### **const byte OperationCode.ExchangeKeysForEncryption = 250**

#### **const byte OperationCode.FindFriends = 222**

(222) Request the rooms and online status for a list of friends (by name, which should be unique).

#### const byte OperationCode.GetGameList = 217

(217) Get the game list matching a supplied sql filter (SqlListLobby only)

#### const byte OperationCode.GetLobbyStats = 221

(221) Request statistics about a specific list of lobbies (their user and game count).

#### const byte OperationCode.GetProperties = (byte)251

(251) Get Properties

#### const byte OperationCode.GetRegions = 220

(220) Get list of regional servers from a NameServer.

#### const byte OperationCode.Join = 255

(255) Code for OpJoin, to get into a room.

#### **const byte OperationCode.JoinGame = 226**

(226) Join game (by name)

#### const byte OperationCode.JoinLobby = 229

(229) Joins lobby (on master)

#### **const byte OperationCode.JoinRandomGame = 225**

(225) Joins random game (on master)

#### const byte OperationCode.Leave = (byte)254

(254) Code for OpLeave, to get out of a room.

#### const byte OperationCode.LeaveLobby = 228

(228) Leaves lobby (on master)

#### const byte OperationCode.RaiseEvent = (byte)253

(253) Raise event (in a room, for other actors/players)

#### const byte OperationCode.ServerSettings = 218

(218) Operation to set some server settings. Used with different parameters on various servers.

#### const byte OperationCode.SetProperties = (byte)252

(252) Set Properties (of room or actor/player)

#### const byte OperationCode.WebRpc = 219

(219) WebRpc Operation.



# Photon Unity Networking

v1.81

Main Page	9	Related	Pages	Modu	les	Classes	Files
Class List	CI	lass Index	Class Hi	erarchy	Clas	ss Members	
					Public	Attributes   List	of all members

## ParameterCode Class Reference

Class for constants. Codes for parameters of Operations and Events. Pun uses these constants internally. <u>More...</u>

#### **Public Attributes**

#### const byte **SuppressRoomEvents** = 237

(237) A bool parameter for creating games. If set to true, no room events are sent to the clients on join and leave. Default: false (and not sent). More...

#### const byte **EmptyRoomTTL** = 236

(236) Time To Live (TTL) for a room when the last player leaves. Keeps room in memory for case a player re-joins soon. In milliseconds. More...

#### const byte PlayerTTL = 235

(235) Time To Live (TTL) for an 'actor' in a room. If a client disconnects, this actor is inactive first and removed after this timeout. In milliseconds. More...

#### const byte **EventForward** = 234

(234) Optional parameter of OpRaiseEvent and OpSetCustomProperties to forward the event/operation to a web-service. <u>More...</u>

#### const byte IsComingBack = (byte)233

(233) Optional parameter of OpLeave in async games. If false, the player does abandons the game (forever). By default players become inactive and can re-join. <u>More...</u>

#### const byte Islnactive = (byte)233

(233) Used in EvLeave to describe if a user is inactive (and might come back) or not. In rooms with PlayerTTL, becoming inactive is the default case. More...

#### const byte **CheckUserOnJoin** = (byte)232

(232) Used when creating rooms to define if any userid can join the room only once. <u>More...</u>

const byte **ExpectedValues** = (byte)231

(231) Code for "Check And Swap" (CAS) when changing properties. More...

const byte Address = 230

(230) Address of a (game) server to use. More...

const byte **PeerCount** = 229

(229) Count of players in this application in a rooms (used in stats event) More...

const byte **GameCount** = 228

(228) Count of games in this application (used in stats event) More...

const byte **MasterPeerCount** = 227

(227) Count of players on the master server (in this app, looking for rooms) More...

const byte **UserId** = 225

(225) User's ID More...

const byte **ApplicationId** = 224

(224) Your application's ID: a name on your own **Photon** or a GUID on the **Photon** Cloud More...

const byte **Position** = 223

(223) Not used currently (as "Position"). If you get queued before connect, this is your position More...

const byte **MatchMakingType** = 223

(223) Modifies the matchmaking algorithm used for OpJoinRandom. Allowed parameter values are defined in enum MatchmakingMode. <u>More...</u>

const byte **GameList** = 222

(222) List of RoomInfos about open / listed rooms More...

const byte **Secret** = 221

(221) Internally used to establish encryption More...

const byte **AppVersion** = 220

(220) Version of your application More...

const byte AzureNodeInfo = 210

(210) Internally used in case of hosting by Azure More...

const byte AzureLocalNodeld = 209

(209) Internally used in case of hosting by Azure More...

const byte AzureMasterNodeld = 208

(208) Internally used in case of hosting by Azure More...

const byte **RoomName** = (byte)255

(255) Code for the gameId/roomName (a unique name per room). Used in OpJoin and similar. More...

const byte **Broadcast** = (byte)250

(250) Code for broadcast parameter of OpSetProperties method. More...

const byte **ActorList** = (byte)252

(252) Code for list of players in a room. Currently not used. More...

const byte ActorNr = (byte)254

(254) Code of the Actor of an operation. Used for property get and set. <u>More...</u>

const byte PlayerProperties = (byte)249

(249) Code for property set (Hashtable). More...

const byte **CustomEventContent** = (byte)245

(245) Code of data/custom content of an event. Used in OpRaiseEvent. More...

#### const byte **Data** = (byte)245

(245) Code of data of an event. Used in OpRaiseEvent. More...

#### const byte Code = (byte)244

(244) Code used when sending some code-related parameter, like OpRaiseEvent's event-code. More...

#### const byte GameProperties = (byte)248

(248) Code for property set (Hashtable). More...

#### const byte **Properties** = (byte)251

(251) Code for property-set (Hashtable). This key is used when sending only one set of properties. If either **ActorProperties** or GameProperties are used (or both), check those keys. More...

#### const byte TargetActorNr = (byte)253

(253) Code of the target Actor of an operation. Used for property set. Is 0 for game More...

#### const byte **ReceiverGroup** = (byte)246

(246) Code to select the receivers of events (used in Lite, Operation RaiseEvent). <u>More...</u>

#### const byte **Cache** = (byte)247

(247) Code for caching events while raising them. More...

#### const byte CleanupCacheOnLeave = (byte)241

(241) Bool parameter of CreateRoom Operation. If true, server cleans up roomcache of leaving players (their cached events get removed). <u>More...</u>

#### const byte Group = 240

(240) Code for "group" operation-parameter (as used in Op RaiseEvent). More...

#### const byte **Remove** = 239

(239) The "Remove" operation-parameter can be used to remove something from a list. E.g. remove groups from player's interest groups. <u>More...</u>

#### const byte PublishUserId = 239

(239) Used in Op Join to define if UserIds of the players are broadcast in the room. Useful for FindFriends and reserving slots for expected users. More...

#### const byte Add = 238

(238) The "Add" operation-parameter can be used to add something to some list or set. E.g. add groups to player's interest groups. <u>More...</u>

#### const byte Info = 218

(218) Content for **0**, this event is fired to let **e...">EventCode.ErrorInfo** and internal debug operations. <u>More...</u>

#### const byte **ClientAuthenticationType** = 217

(217) This key's (byte) value defines the target custom authentication type/service the client connects with. Used in OpAuthenticate More...

#### const byte ClientAuthenticationParams = 216

(216) This key's (string) value provides parameters sent to the custom authentication type/service the client connects with. Used in OpAuthenticate More...

#### const byte **JoinMode** = 215

(215) Makes the server create a room if it doesn't exist. OpJoin uses this to always enter a room, unless it exists and is full/closed. More...

#### const byte ClientAuthenticationData = 214

(214) This key's (string or byte[]) value provides parameters sent to the custom authentication service setup in **Photon** Dashboard. Used in OpAuthenticate More...

#### const byte MasterClientId = (byte)203

(203) Code for MasterClientId, which is synced by server. When sent as op-parameter this is code 203. More...

#### const byte FindFriendsRequestList = (byte)1

(1) Used in Op FindFriends request. Value must be string[] of friends to look up. More...

#### const byte FindFriendsResponseOnlineList = (byte)1

(1) Used in Op FindFriends response. Contains bool[] list of online states (false if not online). More...

#### const byte FindFriendsResponseRoomIdList = (byte)2

(2) Used in Op FindFriends response. Contains string[] of room names ("" where not known or no room joined). More...

#### const byte **LobbyName** = (byte)213

(213) Used in matchmaking-related methods and when creating a room to name a lobby (to join or to attach a room to). More...

#### const byte **LobbyType** = (byte)212

(212) Used in matchmaking-related methods and when creating a room to define the type of a lobby. Combined with the lobby name this identifies the lobby. More...

#### const byte **LobbyStats** = (byte)211

(211) This (optional) parameter can be sent in Op Authenticate to turn on Lobby Stats (info about lobby names and their user- and game-counts). See:

#### PhotonNetwork.Lobbies More...

#### const byte **Region** = (byte)210

(210) Used for region values in OpAuth and OpGetRegions. More...

#### const byte **UriPath** = 209

(209) Path of the WebRPC that got called. Also known as "WebRpc Name". Type: string. More...

#### const byte **WebRpcParameters** = 208

(208) Parameters for a WebRPC as: Dictionary<string, object>. This will get serialized to JSon. More...

#### const byte **WebRpcReturnCode** = 207

(207) ReturnCode for the WebRPC, as sent by the web service (not by **Photon**, which uses **ErrorCode**). Type: byte. More...

#### const byte **WebRpcReturnMessage** = 206

(206) Message returned by WebRPC server. Analog to **Photon**'s debug message. Type: string. <u>More...</u>

#### const byte CacheSliceIndex = 205

(205) Used to define a "slice" for cached events. Slices can easily be removed from cache. Type: int. More...

#### const byte **Plugins** = 204

(204) Informs the server of the expected plugin setup. More...

#### const byte **NickName** = 202

(202) Used by the server in Operation Responses, when it sends the nickname of the client (the user's nickname). More...

#### const byte PluginName = 201

(201) Informs user about name of plugin load to game More...

#### const byte **PluginVersion** = 200

(200) Informs user about version of plugin load to game More...

#### const byte **ExpectedProtocol** = 195

(195) Protocol which will be used by client to connect master/game servers. Used for nameserver. More...

#### const byte **CustomInitData** = 194

(194) Set of custom parameters which are sent in auth request. More...

#### const byte **EncryptionMode** = 193

(193) How are we going to encrypt data. More...

#### const byte **EncryptionData** = 192

(192) Parameter of Authentication, which contains encryption keys (depends on AuthMode and EncryptionMode). <u>More...</u>

# **Detailed Description**

Class for constants. Codes for parameters of Operations and Events. Pun uses these constants internally.

#### Member Data Documentation

#### const byte ParameterCode.ActorList = (byte)252

(252) Code for list of players in a room. Currently not used.

#### const byte ParameterCode.ActorNr = (byte)254

(254) Code of the Actor of an operation. Used for property get and set.

#### const byte ParameterCode.Add = 238

(238) The "Add" operation-parameter can be used to add something to some list or set. E.g. add groups to player's interest groups.

#### **const byte ParameterCode.Address = 230**

(230) Address of a (game) server to use.

#### const byte ParameterCode.ApplicationId = 224

(224) Your application's ID: a name on your own **Photon** or a GUID on the **Photon** Cloud

#### const byte ParameterCode.AppVersion = 220

(220) Version of your application

#### const byte ParameterCode.AzureLocalNodeld = 209

(209) Internally used in case of hosting by Azure

#### const byte ParameterCode.AzureMasterNodeId = 208

(208) Internally used in case of hosting by Azure

#### **const byte ParameterCode.AzureNodeInfo = 210**

(210) Internally used in case of hosting by Azure

#### const byte ParameterCode.Broadcast = (byte)250

(250) Code for broadcast parameter of OpSetProperties method.

#### const byte ParameterCode.Cache = (byte)247

(247) Code for caching events while raising them.

#### **const byte ParameterCode.CacheSliceIndex = 205**

(205) Used to define a "slice" for cached events. Slices can easily be removed from cache. Type: int.

#### const byte ParameterCode.CheckUserOnJoin = (byte)232

(232) Used when creating rooms to define if any userid can join the room only once.

#### const byte ParameterCode.CleanupCacheOnLeave = (byte)241

(241) Bool parameter of CreateRoom Operation. If true, server cleans up roomcache of leaving players (their cached events get removed).

#### const byte ParameterCode.ClientAuthenticationData = 214

(214) This key's (string or byte[]) value provides parameters sent to the custom authentication service setup in **Photon** Dashboard. Used in OpAuthenticate

#### **const byte ParameterCode.ClientAuthenticationParams = 216**

(216) This key's (string) value provides parameters sent to the custom authentication type/service the client connects with. Used in OpAuthenticate

#### const byte ParameterCode.ClientAuthenticationType = 217

(217) This key's (byte) value defines the target custom authentication type/service the client connects with. Used in OpAuthenticate

#### const byte ParameterCode.Code = (byte)244

(244) Code used when sending some code-related parameter, like OpRaiseEvent's event-code.

This is not the same as the Operation's code, which is no longer sent as part of the parameter Dictionary in **Photon** 3.

#### const byte ParameterCode.CustomEventContent = (byte)245

(245) Code of data/custom content of an event. Used in OpRaiseEvent.

#### const byte ParameterCode.CustomInitData = 194

(194) Set of custom parameters which are sent in auth request.

#### const byte ParameterCode.Data = (byte)245

(245) Code of data of an event. Used in OpRaiseEvent.

#### const byte ParameterCode.EmptyRoomTTL = 236

(236) Time To Live (TTL) for a room when the last player leaves. Keeps room in memory for case a player re-joins soon. In milliseconds.

#### **const byte ParameterCode.EncryptionData = 192**

(192) Parameter of Authentication, which contains encryption keys (depends on AuthMode and EncryptionMode).

#### **const byte ParameterCode.EncryptionMode = 193**

(193) How are we going to encrypt data.

#### **const byte ParameterCode.EventForward = 234**

(234) Optional parameter of OpRaiseEvent and

OpSetCustomProperties to forward the event/operation to a webservice.

#### **const byte ParameterCode.ExpectedProtocol = 195**

(195) Protocol which will be used by client to connect master/game servers. Used for nameserver.

#### const byte ParameterCode.ExpectedValues = (byte)231

(231) Code for "Check And Swap" (CAS) when changing properties.

#### const byte ParameterCode.FindFriendsRequestList = (byte)1

(1) Used in Op FindFriends request. Value must be string[] of friends to look up.

# const byte ParameterCode.FindFriendsResponseOnlineList = (byte)1

(1) Used in Op FindFriends response. Contains bool[] list of online states (false if not online).

# const byte ParameterCode.FindFriendsResponseRoomIdList = (byte)2

(2) Used in Op FindFriends response. Contains string[] of room names ("" where not known or no room joined).

#### **const byte ParameterCode.GameCount = 228**

(228) Count of games in this application (used in stats event)

#### const byte ParameterCode.GameList = 222

(222) List of RoomInfos about open / listed rooms

#### const byte ParameterCode.GameProperties = (byte)248

(248) Code for property set (Hashtable).

#### const byte ParameterCode.Group = 240

(240) Code for "group" operation-parameter (as used in Op RaiseEvent).

#### **const byte ParameterCode.Info = 218**

(218) Content for **0**, **this event is fired to let e...">EventCode.ErrorInfo** and internal debug operations.

#### const byte ParameterCode.IsComingBack = (byte)233

(233) Optional parameter of OpLeave in async games. If false, the player does abandons the game (forever). By default players become inactive and can re-join.

#### const byte ParameterCode.IsInactive = (byte)233

(233) Used in EvLeave to describe if a user is inactive (and might come back) or not. In rooms with PlayerTTL, becoming inactive is

the default case.

#### const byte ParameterCode.JoinMode = 215

- (215) Makes the server create a room if it doesn't exist. OpJoin uses this to always enter a room, unless it exists and is full/closed.
- (215) The JoinMode enum defines which variant of joining a room will be executed: Join only if available, create if not exists or re-join.

Replaces CreatelfNotExists which was only a bool-value.

#### const byte ParameterCode.LobbyName = (byte)213

(213) Used in matchmaking-related methods and when creating a room to name a lobby (to join or to attach a room to).

#### const byte ParameterCode.LobbyStats = (byte)211

(211) This (optional) parameter can be sent in Op Authenticate to turn on Lobby Stats (info about lobby names and their user- and game-counts). See: PhotonNetwork.Lobbies

#### const byte ParameterCode.LobbyType = (byte)212

(212) Used in matchmaking-related methods and when creating a room to define the type of a lobby. Combined with the lobby name this identifies the lobby.

#### const byte ParameterCode.MasterClientId = (byte)203

(203) Code for MasterClientId, which is synced by server. When sent as op-parameter this is code 203.

Tightly related to **GamePropertyKey.MasterClientId**.

#### **const byte ParameterCode.MasterPeerCount = 227**

(227) Count of players on the master server (in this app, looking for rooms)

#### const byte ParameterCode.MatchMakingType = 223

(223) Modifies the matchmaking algorithm used for OpJoinRandom. Allowed parameter values are defined in enum MatchmakingMode.

#### const byte ParameterCode.NickName = 202

(202) Used by the server in Operation Responses, when it sends the nickname of the client (the user's nickname).

#### const byte ParameterCode.PeerCount = 229

(229) Count of players in this application in a rooms (used in stats event)

#### const byte ParameterCode.PlayerProperties = (byte)249

(249) Code for property set (Hashtable).

#### const byte ParameterCode.PlayerTTL = 235

(235) Time To Live (TTL) for an 'actor' in a room. If a client disconnects, this actor is inactive first and removed after this timeout. In milliseconds.

#### const byte ParameterCode.PluginName = 201

(201) Informs user about name of plugin load to game

#### const byte ParameterCode.Plugins = 204

(204) Informs the server of the expected plugin setup.

The operation will fail in case of a plugin mismatch returning error code PluginMismatch 32751(0x7FFF - 16). Setting string[]{} means the client expects no plugin to be setup. Note: for backwards compatibility null omits any check.

#### **const byte ParameterCode.PluginVersion = 200**

(200) Informs user about version of plugin load to game

#### **const byte ParameterCode.Position = 223**

(223) Not used currently (as "Position"). If you get queued before connect, this is your position

#### const byte ParameterCode.Properties = (byte)251

(251) Code for property-set (Hashtable). This key is used when sending only one set of properties. If either **ActorProperties** or GameProperties are used (or both), check those keys.

#### const byte ParameterCode.PublishUserId = 239

(239) Used in Op Join to define if Userlds of the players are

broadcast in the room. Useful for FindFriends and reserving slots for expected users.

#### const byte ParameterCode.ReceiverGroup = (byte)246

(246) Code to select the receivers of events (used in Lite, Operation RaiseEvent).

#### const byte ParameterCode.Region = (byte)210

(210) Used for region values in OpAuth and OpGetRegions.

#### const byte ParameterCode.Remove = 239

(239) The "Remove" operation-parameter can be used to remove something from a list. E.g. remove groups from player's interest groups.

#### const byte ParameterCode.RoomName = (byte)255

(255) Code for the gameId/roomName (a unique name per room). Used in OpJoin and similar.

#### **const byte ParameterCode.Secret = 221**

(221) Internally used to establish encryption

#### **const byte ParameterCode.SuppressRoomEvents = 237**

(237) A bool parameter for creating games. If set to true, no room events are sent to the clients on join and leave. Default: false (and

not sent).

#### const byte ParameterCode.TargetActorNr = (byte)253

(253) Code of the target Actor of an operation. Used for property set. Is 0 for game

#### const byte ParameterCode.UriPath = 209

(209) Path of the WebRPC that got called. Also known as "WebRpc Name". Type: string.

#### **const byte ParameterCode.UserId = 225**

(225) User's ID

#### **const byte ParameterCode.WebRpcParameters = 208**

(208) Parameters for a WebRPC as: Dictionary<string, object>. This will get serialized to JSon.

#### const byte ParameterCode.WebRpcReturnCode = 207

(207) ReturnCode for the WebRPC, as sent by the web service (not by **Photon**, which uses **ErrorCode**). Type: byte.

#### const byte ParameterCode.WebRpcReturnMessage = 206

(206) Message returned by WebRPC server. Analog to **Photon**'s debug message. Type: string.



# Photon Unity Networking

v1.81

Main Page	e Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	
	Animator eference		<u>Public N</u>	Member	Classes   F Functions   List (	Public Types   of all members

This class helps you to synchronize Mecanim animations Simply add the component to your GameObject and make sure that the **PhotonAnimatorView** is added to the list of observed components More...

Inherits MonoBehaviour, and IPunObservable.

## Classes

class **SynchronizedLayer** 

class SynchronizedParameter

### Public Types

```
enum ParameterType { ParameterType.Float = 1,
ParameterType.Int = 3, ParameterType.Bool = 4,
ParameterType.Trigger = 9 }

SynchronizeType { SynchronizeType.Disabled = 0,
SynchronizeType.Discrete = 1,
SynchronizeType.Continuous = 2 }
```

## **Public Member Functions**

Fublic Mellibel Fullctions	
void	CacheDiscreteTriggers () Caches the discrete triggers values f keeping track of raised triggers, and be reseted after the sync routine got performed More
	DoesLayerSynchronizeTypeExist (
bool	layerIndex) Check if a specific layer is configured be synchronize More
bool	DoesParameterSynchronizeTypeE (string name) Check if the specified parameter is configured to be synchronized More.
List< SynchronizedLayer >	GetSynchronizedLayers ()
List v Oynom om zeazayer >	Get a list of all synchronized layers More
List< SynchronizedParameter >	GetSynchronizedParameters () Get a list of all synchronized parameters More
SynchronizeType	GetLayerSynchronizeType (int layerIndex) Gets the type how the layer is synchronized More
SynchronizeType	GetParameterSynchronizeType (st name) Gets the type how the parameter is synchronized More

void **SetLayerSynchronized** (int layerInc **SynchronizeType** synchronizeType) Sets the how a layer should be synchronized <u>More...</u>

void name, ParameterType type,
SynchronizeType synchronizeType)
Sets the how a parameter should be synchronized More...

void (PhotonStream stream, PhotonMessageInfo info)

Called by PUN several times per second, so that your script can write and read synchronization data for the **PhotonView**. More...

## **Detailed Description**

This class helps you to synchronize Mecanim animations Simply add the component to your GameObject and make sure that the **PhotonAnimatorView** is added to the list of observed components

When Using Trigger Parameters, make sure the component that sets the trigger is higher in the stack of Components on the GameObject than 'PhotonAnimatorView' Triggers are raised true during one frame only.

# Member Enumeration Documentation

um Phot	tonAnimatorView.ParameterType
Enumerator	
Float	
Int	
Bool	
Trigger	

enum PhotonAnimatorView.SynchronizeType		

#### **Member Function Documentation**

#### void PhotonAnimatorView.CacheDiscreteTriggers ( )

Caches the discrete triggers values for keeping track of raised triggers, and will be reseted after the sync routine got performed

#### bool

PhotonAnimatorView.DoesLayerSynchronizeTypeExist (int layerIr

Check if a specific layer is configured to be synchronize

#### **Parameters**

layerIndex Index of the layer.

#### **Returns**

True if the layer is synchronized

#### bool

PhotonAnimatorView.DoesParameterSynchronizeTypeExist ( string

Check if the specified parameter is configured to be synchronized

#### **Parameters**

name The name of the parameter.

#### **Returns**

True if the parameter is synchronized

#### SynchronizeType

PhotonAnimatorView.GetLayerSynchronizeType (int layerIndex)

Gets the type how the layer is synchronized

#### **Parameters**

layerIndex Index of the layer.

#### **Returns**

Disabled/Discrete/Continuous

#### **SynchronizeType**

PhotonAnimatorView.GetParameterSynchronizeType (string name

Gets the type how the parameter is synchronized

#### **Parameters**

**name** The name of the parameter.

#### **Returns**

Disabled/Discrete/Continuous

#### List<SynchronizedLayer>

PhotonAnimatorView.GetSynchronizedLayers

()

Get a list of all synchronized layers

#### Returns

List of SynchronizedLayer objects

#### List<SynchronizedParameter>

PhotonAnimatorView.GetSynchronizedParameters

()

Get a list of all synchronized parameters

#### Returns

List of SynchronizedParameter objects

# void PhotonAnimatorView.OnPhotonSerializeView ( PhotonStream PhotonMessageInfo )

Called by PUN several times per second, so that your script can write a synchronization data for the **PhotonView**.

This method will be called in scripts that are assigned as Observed con of a **PhotonView**.

PhotonNetwork.sendRateOnSerialize affects how often this method i PhotonNetwork.sendRate affects how often packages are sent by this

Implementing this method, you can customize which data a **PhotonVie** regularly synchronizes. Your code defines what is being sent (content) a your data is used by receiving clients.

Unlike other callbacks, *OnPhotonSerializeView only gets called when it assigned to a PhotonView* as PhotonView.observed script.

To make use of this method, the **PhotonStream** is essential. It will be in "writing" mode" on the client that controls a PhotonView (PhotonStream.isWriting == true) and in "reading mode" on the remote that just receive that the controlling client sends.

If you skip writing any value into the stream, PUN will skip the update. L carefully, this can conserve bandwidth and messages (which have a lim room/second).

Note that OnPhotonSerializeView is not called on remote clients when t sender does not send any update. This can't be used as "x-times per se Update()".

Implements IPunObservable.

```
void
PhotonAnimatorView.SetLayerSynchronized ( int SynchronizeType sy
```

Sets the how a layer should be synchronized

#### **Parameters**

layerIndex Index of the layer.

synchronizeType Disabled/Discrete/Continuous

#### void

PhotonAnimatorView.SetParameterSynchronized (string

ParameterType SynchronizeType

Sets the how a parameter should be synchronized

#### **Parameters**

name The name of the parameter.type The type of the parameter.synchronizeType Disabled/Discrete/Continuous

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



v1.81

Main Page	Related	Related Pages		lles	Classes	Files	
Class List	Class Index	Class Hierarchy		Class Members			
PhotonAnimatorView SynchronizedLayer							
				<u>Public</u>	Attributes   List o	of all members	

# PhotonAnimatorView.SynchronizedLayer Class Reference

### **Public Attributes**

### SynchronizeType SynchronizeType

### int LayerIndex

### Member Data Documentation

int PhotonAnimatorView.SynchronizedLayer.LayerIndex

SynchronizeType PhotonAnimatorView.SynchronizedLayer.SynchronizeType

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



v1.81

Main Page	е	Related Pages		Modu	les	Classes	Files
Class List	Clas	ss Index	Class Hierarchy		Class Members		
PhotonAnimatorView > SynchronizedParameter >							
					Public	: Attributes   List	of all members

# PhotonAnimatorView.SynchronizedParameter Class Reference

### **Public Attributes**

ParameterType Type

SynchronizeType SynchronizeType

string Name

### Member Data Documentation

string PhotonAnimatorView.SynchronizedParameter.Name

**SynchronizeType** 

PhotonAnimatorView.SynchronizedParameter.SynchronizeType

**ParameterType** 

PhotonAnimatorView.SynchronizedParameter.Type

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



v1.81

Main Page	ge Related Pages		Modules		Classes	Files
Class List	Class Index	Class Hierarchy		Class Members		
					Public Member	er Functions
Photon	PingMana	ager		<u>Sta</u>	atic Public Membe	er Functions
	eference	_	<u>Pub</u>	lic Attrib	utes   Static Pub	lic Attributes
Class R	CICICIICE				Properties   List of	of all members

### **Public Member Functions**

IEnumerator PingSocket (Region region)

### Static Public Member Functions

### static string ResolveHost (string hostName)

Attempts to resolve a hostname into an IP string or returns empty string if that fails. More...

### **Public Attributes**

### bool UseNative

### Static Public Attributes

static int **Attempts** = 5

static bool IgnoreInitialAttempt = true

static int MaxMilliseconsPerPing = 800

### Properties

Region BestRegion [get]

bool Done [get]

### **Member Function Documentation**

### IEnumerator PhotonPingManager.PingSocket (Region region)

Affected by frame-rate of app, as this Coroutine checks the socket for a result once per frame.

### static string PhotonPingManager.ResolveHost

(string hostName) static



Attempts to resolve a hostname into an IP string or returns empty string if that fails.

To be compatible with most platforms, the address family is checked like this: if (ipAddress.AddressFamily.ToString().Contains("6")) // ipv6... </reamrks>

#### **Parameters**

hostName Hostname to resolve.

#### Returns

IP string or empty string if resolution fails

### Member Data Documentation

int PhotonPingManager.Attemp	ts = 5
------------------------------	--------

static

bool PhotonPingManager.IgnoreInitialAttempt = true

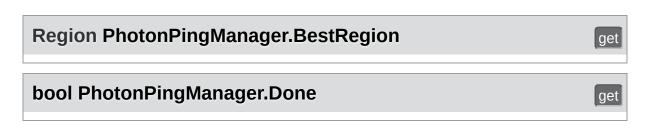
static

int PhotonPingManager.MaxMilliseconsPerPing = 800

static

bool PhotonPingManager.UseNative

### **Property Documentation**



Online Documentation - Dashboard - Support Forum

Exit Games GmbH



v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Member Functions | List of all members

### PhotonRigidbody2DView Class Reference

This class helps you to synchronize the velocities of a 2d physics RigidBody. Note that only the velocities are synchronized and because Unitys physics engine is not deterministic (ie. the results aren't always the same on all computers) - the actual positions of the objects may go out of sync. If you want to have the position of this object the same on all clients, you should also add a **PhotonTransformView** to synchronize the position. Simply add the component to your GameObject and make sure that the **PhotonRigidbody2DView** is added to the list of observed components <u>More...</u>

Inherits MonoBehaviour, and IPunObservable.

### **Public Member Functions**

### void

### OnPhotonSerializeView (PhotonStream stream, PhotonMessageInfo info)

Called by PUN several times per second, so that your script can write and read synchronization data for the **PhotonView**. More...

### **Detailed Description**

This class helps you to synchronize the velocities of a 2d physics RigidBody. Note that only the velocities are synchronized and because Unitys physics engine is not deterministic (ie. the results aren't always the same on all computers) - the actual positions of the objects may go out of sync. If you want to have the position of this object the same on all clients, you should also add a **PhotonTransformView** to synchronize the position. Simply add the component to your GameObject and make sure that the **PhotonRigidbody2DView** is added to the list of observed components

### Member Function Documentation

### void PhotonRigidbody2DView.OnPhotonSerializeView ( PhotonStream PhotonMessage )

Called by PUN several times per second, so that your script can write a synchronization data for the **PhotonView**.

This method will be called in scripts that are assigned as Observed con **PhotonView**.

PhotonNetwork.sendRateOnSerialize affects how often this method i PhotonNetwork.sendRate affects how often packages are sent by this

Implementing this method, you can customize which data a **PhotonVie** synchronizes. Your code defines what is being sent (content) and how y used by receiving clients.

Unlike other callbacks, *OnPhotonSerializeView only gets called when it to a PhotonView* as PhotonView.observed script.

To make use of this method, the **PhotonStream** is essential. It will be it mode" on the client that controls a PhotonView (PhotonStream.isWriting and in "reading mode" on the remote clients that just receive that the coclient sends.

If you skip writing any value into the stream, PUN will skip the update. It carefully, this can conserve bandwidth and messages (which have a lim room/second).

Note that OnPhotonSerializeView is not called on remote clients when t does not send any update. This can't be used as "x-times per second U

Implements IPunObservable.



v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Member Functions | List of all members

# PhotonRigidbodyView Class Reference

This class helps you to synchronize the velocities of a physics RigidBody. Note that only the velocities are synchronized and because Unitys physics engine is not deterministic (ie. the results aren't always the same on all computers) - the actual positions of the objects may go out of sync. If you want to have the position of this object the same on all clients, you should also add a **PhotonTransformView** to synchronize the position. Simply add the component to your GameObject and make sure that the **PhotonRigidbodyView** is added to the list of observed components More...

Inherits MonoBehaviour, and IPunObservable.

### **Public Member Functions**

### void

### OnPhotonSerializeView (PhotonStream stream, PhotonMessageInfo info)

Called by PUN several times per second, so that your script can write and read synchronization data for the **PhotonView**. More...

### **Detailed Description**

This class helps you to synchronize the velocities of a physics RigidBody. Note that only the velocities are synchronized and because Unitys physics engine is not deterministic (ie. the results aren't always the same on all computers) - the actual positions of the objects may go out of sync. If you want to have the position of this object the same on all clients, you should also add a **PhotonTransformView** to synchronize the position. Simply add the component to your GameObject and make sure that the **PhotonRigidbodyView** is added to the list of observed components

### **Member Function Documentation**

# void PhotonRigidbodyView.OnPhotonSerializeView ( PhotonStream PhotonMessageIn )

Called by PUN several times per second, so that your script can write a synchronization data for the **PhotonView**.

This method will be called in scripts that are assigned as Observed con a **PhotonView**.

PhotonNetwork.sendRateOnSerialize affects how often this method i PhotonNetwork.sendRate affects how often packages are sent by this

Implementing this method, you can customize which data a **PhotonVie** regularly synchronizes. Your code defines what is being sent (content) a your data is used by receiving clients.

Unlike other callbacks, *OnPhotonSerializeView only gets called when it assigned to a PhotonView* as PhotonView.observed script.

To make use of this method, the **PhotonStream** is essential. It will be it mode" on the client that controls a PhotonView (PhotonStream.isWriting and in "reading mode" on the remote clients that just receive that the coclient sends.

If you skip writing any value into the stream, PUN will skip the update. It carefully, this can conserve bandwidth and messages (which have a lim room/second).

Note that OnPhotonSerializeView is not called on remote clients when t does not send any update. This can't be used as "x-times per second U

Implements IPunObservable.



v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Member Functions | List of all members

## PhotonStreamQueue Class Reference

The **PhotonStreamQueue** helps you poll object states at higher frequencies then what **PhotonNetwork.sendRate** dictates and then sends all those states at once when **Serialize()** is called. On the receiving end you can call **Deserialize()** and then the stream will roll out the received object states in the same order and timeStep they were recorded in. More...

### **Public Member Functions**

### PhotonStreamQueue (int sampleRate)

Initializes a new instance of the **PhotonStreamQueue** class. More...

### void Reset ()

Resets the **PhotonStreamQueue**. You need to do this whenever the amount of objects you are observing changes More...

### void **SendNext** (object obj)

Adds the next object to the queue. This works just like **PhotonStream.SendNext** More...

### bool HasQueuedObjects ()

Determines whether the queue has stored any objects More...

### object ReceiveNext ()

Receives the next object from the queue. This works just like **PhotonStream.ReceiveNext** More...

### void Serialize (PhotonStream stream)

Serializes the specified stream. Call this in your OnPhotonSerializeView method to send the whole recorded stream. More...

### void **Deserialize** (**PhotonStream** stream)

Deserializes the specified stream. Call this in your OnPhotonSerializeView method to receive the whole recorded stream. More...

### **Detailed Description**

The **PhotonStreamQueue** helps you poll object states at higher frequencies then what **PhotonNetwork.sendRate** dictates and then sends all those states at once when **Serialize()** is called. On the receiving end you can call **Deserialize()** and then the stream will roll out the received object states in the same order and timeStep they were recorded in.

### Constructor & Destructor Documentation

### PhotonStreamQueue.PhotonStreamQueue (int sampleRate)

Initializes a new instance of the **PhotonStreamQueue** class.

### **Parameters**

**sampleRate** How many times per second should the object states be sampled

### **Member Function Documentation**

### void PhotonStreamQueue.Deserialize (PhotonStream stream)

Deserializes the specified stream. Call this in your OnPhotonSerializeView method to receive the whole recorded stream.

#### **Parameters**

**stream** The **PhotonStream** you receive as a parameter in OnPhotonSerializeView

### bool PhotonStreamQueue.HasQueuedObjects ( )

Determines whether the queue has stored any objects

### object PhotonStreamQueue.ReceiveNext ( )

Receives the next object from the queue. This works just like **PhotonStream.ReceiveNext** 

#### Returns

### void PhotonStreamQueue.Reset ()

Resets the **PhotonStreamQueue**. You need to do this whenever the amount of objects you are observing changes

### void PhotonStreamQueue.SendNext ( object obj )

Adds the next object to the queue. This works just like **PhotonStream.SendNext** 

#### **Parameters**

obj The object you want to add to the queue

### void PhotonStreamQueue.Serialize (PhotonStream stream)

Serializes the specified stream. Call this in your OnPhotonSerializeView method to send the whole recorded stream.

#### **Parameters**

**stream** The **PhotonStream** you receive as a parameter in OnPhotonSerializeView

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Member Functions | List of all members

## PhotonTransformView Class Reference

This class helps you to synchronize position, rotation and scale of a GameObject. It also gives you many different options to make the synchronized values appear smooth, even when the data is only send a couple of times per second. Simply add the component to your GameObject and make sure that the **PhotonTransformView** is added to the list of observed components <u>More...</u>

Inherits MonoBehaviour, and IPunObservable.

### **Public Member Functions**

### void **SetSynchronizedValues** (Vector3 speed, float turnSpeed)

These values are synchronized to the remote objects if the interpolation mode or the extrapolation mode SynchronizeValues is used. Your movement script should pass on the current speed (in units/second) and turning speed (in angles/second) so the remote object can use them to predict the objects movement. More...

### void

### OnPhotonSerializeView (PhotonStream stream, PhotonMessageInfo info)

Called by PUN several times per second, so that your script can write and read synchronization data for the **PhotonView**. More...

### **Detailed Description**

This class helps you to synchronize position, rotation and scale of a GameObject. It also gives you many different options to make the synchronized values appear smooth, even when the data is only send a couple of times per second. Simply add the component to your GameObject and make sure that the **PhotonTransformView** is added to the list of observed components

### Member Function Documentation

# void PhotonTransformView.OnPhotonSerializeView ( PhotonStream PhotonMessageIn )

Called by PUN several times per second, so that your script can write a synchronization data for the **PhotonView**.

This method will be called in scripts that are assigned as Observed con a **PhotonView**.

PhotonNetwork.sendRateOnSerialize affects how often this method i PhotonNetwork.sendRate affects how often packages are sent by this

Implementing this method, you can customize which data a **PhotonVie** regularly synchronizes. Your code defines what is being sent (content) a your data is used by receiving clients.

Unlike other callbacks, *OnPhotonSerializeView only gets called when it assigned to a PhotonView* as PhotonView.observed script.

To make use of this method, the **PhotonStream** is essential. It will be it mode" on the client that controls a PhotonView (PhotonStream.isWriting and in "reading mode" on the remote clients that just receive that the coclient sends.

If you skip writing any value into the stream, PUN will skip the update. It carefully, this can conserve bandwidth and messages (which have a lim room/second).

Note that OnPhotonSerializeView is not called on remote clients when t does not send any update. This can't be used as "x-times per second U

Implements IPunObservable.

# void PhotonTransformView.SetSynchronizedValues ( Vector3 speed, float turnSpeed)

These values are synchronized to the remote objects if the interpolatior mode or the extrapolation mode SynchronizeValues is used. Your movement script should pass on the current speed (in units/second) and turning speed (in angles/second) so the remote object can use them to predict the objects movement.

#### **Parameters**

**speed** The current movement vector of the object in

units/second.

turnSpeed The current turn speed of the object in angles/second.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Member Functions | List of all members

# PhotonTransformViewPositionControl Class Reference

### **Public Member Functions**

# PhotonTransformViewPositionControl (PhotonTransformViewPositionModel model)

void **SetSynchronizedValues** (Vector3 speed, float turnSpeed)
These values are synchronized to the remote objects if the interpolation mode or the extrapolation mode
SynchronizeValues is used. Your movement script should pass on the current speed (in units/second) and turning speed (in angles/second) so the remote object can use them to predict the objects movement. More...

### Vector3 UpdatePosition (Vector3 currentPosition)

Calculates the new position based on the values setup in the inspector <u>More...</u>

#### Vector3 GetNetworkPosition ()

Gets the last position that was received through the network More...

### Vector3 GetExtrapolatedPositionOffset ()

Calculates an estimated position based on the last synchronized position, the time when the last position was received and the movement speed of the object More...

void OnPhotonSerializeView (Vector3 currentPosition, PhotonStream stream, PhotonMessageInfo info)

# Constructor & Destructor Documentation

PhotonTransformViewPositionControl.PhotonTransformViewPosit

## **Member Function Documentation**

#### Vector3

#### PhotonTransformViewPositionControl.GetExtrapolatedPositionOff

Calculates an estimated position based on the last synchronized positic the time when the last position was received and the movement speed object

#### Returns

Estimated position of the remote object

#### Vector3

PhotonTransformViewPositionControl.GetNetworkPosition

()

Gets the last position that was received through the network

#### Returns

#### void

PhotonTransformViewPositionControl.OnPhotonSerializeView (Ve

Ph

Ph

#### void

)

These values are synchronized to the remote objects if the interpolation

extrapolation mode SynchronizeValues is used. Your movement script the current speed (in units/second) and turning speed (in angles/second object can use them to predict the objects movement.

#### **Parameters**

**speed** The current movement vector of the object in units/sec **turnSpeed** The current turn speed of the object in angles/second.

#### Vector3

PhotonTransformViewPositionControl.UpdatePosition (Vector3 cu

Calculates the new position based on the values setup in the inspector

#### **Parameters**

currentPosition The current position.

#### Returns

The new position.

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	d Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	ierarchy	Clas	ss Members	

Public Types | Public Attributes | List of all members

# PhotonTransformViewPositionModel Class Reference

## **Public Types**

```
InterpolateOptions {
    InterpolateOptions.Disabled,
    InterpolateOptions.FixedSpeed,
    InterpolateOptions.EstimatedSpeed,
    InterpolateOptions.SynchronizeValues,
    InterpolateOptions.Lerp
}

ExtrapolateOptions { ExtrapolateOptions.Disabled,
    ExtrapolateOptions.SynchronizeValues,
    ExtrapolateOptions.EstimateSpeedAndTurn,
    ExtrapolateOptions.FixedSpeed }
```

# **Public Attributes**

bool	SynchronizeEnabled
bool	TeleportEnabled = true
float	TeleportIfDistanceGreaterThan = 3f
InterpolateOptions	InterpolateOption = InterpolateOptions.EstimatedSpeed
float	InterpolateMoveTowardsSpeed = 1f
float	InterpolateLerpSpeed = 1f
float	InterpolateMoveTowardsAcceleration = 2
float	InterpolateMoveTowardsDeceleration = 2
AnimationCurve	InterpolateSpeedCurve
ExtrapolateOptions	ExtrapolateOption = ExtrapolateOptions.Disabled
float	ExtrapolateSpeed = 1f
bool	ExtrapolateIncludingRoundTripTime = true
int	ExtrapolateNumberOfStoredPositions = 1
bool	DrawErrorGizmo = true

# Member Enumeration Documentation

## ${\bf enum\ Photon Transform View Position Model. Extrapolate Options}$

Enumerator	
Disabled	
SynchronizeValues	
EstimateSpeedAndTurn	
FixedSpeed	

# enum PhotonTransformViewPositionModel.InterpolateOptions

Enumerator	
Disabled	
FixedSpeed	
EstimatedSpeed	
SynchronizeValues	
Lerp	

## Member Data Documentation

bool PhotonTransformViewPositionModel.DrawErrorGizmo = true

bool

PhotonTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionModel.ExtrapolateIncludingRoundTransformViewPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPositionPos

int

PhotonTransformViewPositionModel.ExtrapolateNumberOfStoredI = 1

#### **ExtrapolateOptions**

PhotonTransformViewPositionModel.ExtrapolateOption = ExtrapolateOptions.Disabled

float PhotonTransformViewPositionModel.ExtrapolateSpeed = 1f

float PhotonTransformViewPositionModel.InterpolateLerpSpeed = 1f

float

PhotonTransformViewPositionModel.InterpolateMoveTowardsAcco = 2

float

PhotonTransformViewPositionModel.InterpolateMoveTowardsDece = 2

float

PhotonTransformViewPositionModel.InterpolateMoveTowardsSpec = 1f

#### **InterpolateOptions**

PhotonTransformViewPositionModel.InterpolateOption = InterpolateOptions.EstimatedSpeed

# AnimationCurve PhotonTransformViewPositionModel.InterpolateSpeedCurve

#### **Initial value:**

```
= new AnimationCurve( new Keyframe[] {
  new Keyframe( -1, 0, 0, Mathf.Infinity ),
  new Keyframe( 0, 1, 0, 0 ),
  new Keyframe( 1, 1, 0, 1 ),
  new Keyframe( 4, 4, 1, 0 ) } )
```

bool PhotonTransformViewPositionModel.SynchronizeEnabled

bool PhotonTransformViewPositionModel.TeleportEnabled = true

#### float

PhotonTransformViewPositionModel.TeleportIfDistanceGreaterThat = 3f

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Member Functions | List of all members

# PhotonTransformViewRotationControl Class Reference

## **Public Member Functions**

PhotonTransformViewRotationControl
(PhotonTransformViewRotationModel model)

Quaternion

GetNetworkRotation ()
Gets the last rotation that was received through the network More...

Quaternion

GetRotation (Quaternion currentRotation)

Void

OnPhotonSerializeView (Quaternion currentRotation, PhotonStream stream, PhotonMessageInfo info)

# Constructor & Destructor Documentation

PhotonTransformViewRotationControl.PhotonTransformViewRotat

## **Member Function Documentation**

# Quaternion PhotonTransformViewRotationControl.GetNetworkRotation () Gets the last rotation that was received through the network

#### **Returns**

# Quaternion PhotonTransformViewRotationControl.GetRotation ( Quaternion c



Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Types | Public Attributes |
List of all members

# PhotonTransformViewRotationModel Class Reference

# Public Types

InterpolateOptions { InterpolateOptions.Disabled, enum InterpolateOptions.RotateTowards, InterpolateOptions.Lerp }

# **Public Attributes**

bool	SynchronizeEnabled
InterpolateOptions	InterpolateOption = InterpolateOptions.RotateTowards
float	InterpolateRotateTowardsSpeed = 180
float	InterpolateLerpSpeed = 5

# **Member Enumeration Documentation**

# enum PhotonTransformViewRotationModel.InterpolateOptions Enumerator Disabled RotateTowards Lerp

## Member Data Documentation

float PhotonTransformViewRotationModel.InterpolateLerpSpeed = 5

#### **InterpolateOptions**

PhotonTransformViewRotationModel.InterpolateOption = InterpolateOptions.RotateTowards

#### float

PhotonTransformViewRotationModel.InterpolateRotateTowardsSp = 180

bool PhotonTransformViewRotationModel.SynchronizeEnabled

Online Documentation - Dashboard - Support Forum



v1.81

Class List Class Index Class Hierarchy Class Members	Main Page	e Related	Pages	Modu	les	Classes	Files
Old33 Filet Old33 Members	Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Member Functions | List of all members

# PhotonTransformViewScaleControl Class Reference

## **Public Member Functions**

PhotonTransformViewScaleControl (PhotonTransformViewScaleModel model)

Vector3 GetNetworkScale ()

Gets the last scale that was received through the network More...

Vector3 GetScale (Vector3 currentScale)

void OnPhotonSerializeView (Vector3 currentScale, PhotonStream stream, PhotonMessageInfo info)

# Constructor & Destructor Documentation

PhotonTransformViewScaleControl.PhotonTransformViewScaleCo

## **Member Function Documentation**

## Vector3 PhotonTransformViewScaleControl.GetNetworkScale()

Gets the last scale that was received through the network

#### **Returns**

#### Vector3

PhotonTransformViewScaleControl.GetScale (Vector3 currentSca

#### void

PhotonTransformViewScaleControl.OnPhotonSerializeView ( Vector Photon Ph

Pho

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Types | Public Attributes |
List of all members

# PhotonTransformViewScaleModel Class Reference

# Public Types

InterpolateOptions { InterpolateOptions.Disabled, enum InterpolateOptions.MoveTowards, InterpolateOptions.Lerp }

# **Public Attributes**

bool	SynchronizeEnabled
InterpolateOptions	InterpolateOption = InterpolateOptions.Disabled
float	InterpolateMoveTowardsSpeed = 1f
float	InterpolateLerpSpeed

# Member Enumeration Documentation

enum PhotonTr	ansformViewScaleModel.InterpolateOptions
Enumerator	
Disabled	
MoveTowards	
Lerp	

## Member Data Documentation

### float PhotonTransformViewScaleModel.InterpolateLerpSpeed

#### float

PhotonTransformViewScaleModel.InterpolateMoveTowardsSpeed = 1f

#### **InterpolateOptions**

PhotonTransformViewScaleModel.InterpolateOption = InterpolateOptions.Disabled

bool PhotonTransformViewScaleModel.SynchronizeEnabled

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

Public Member Functions | List of all members

# PingMonoEditor Class Reference

Uses C# Socket class from System.Net.Sockets (as Unity usually does). <u>More...</u>

Inherits PhotonPing.

## **Public Member Functions**

override bool StartPing (string ip)
Sends a "Photon Ping" to a server. More...

override bool Done ()

override void Dispose ()

# **Detailed Description**

Uses C# Socket class from System.Net.Sockets (as Unity usually does).

Incompatible with Windows 8 Store/Phone API.

## **Member Function Documentation**

### override void PingMonoEditor.Dispose ()

## override bool PingMonoEditor.Done ()

#### override bool PingMonoEditor.StartPing (string ip)

Sends a "Photon Ping" to a server.

#### **Parameters**

ip Address in IPv4 or IPv6 format. An address containing a '.' will be interpreted as IPv4.

#### Returns

True if the **Photon** Ping could be sent.

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Related	Related Pages		les	Classes	Files
Class List	Class Index Class Hid		erarchy Clas		ss Members	

## **PunRPC Class Reference**

Replacement for RPC attribute with different name. Used to flag methods as remote-callable. <u>More...</u>

Inherits Attribute.

# **Detailed Description**

Replacement for RPC attribute with different name. Used to flag methods as remote-callable.

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Relate		Pages	Modules		Classes	Files
Class List	Class Index	Class Hierarchy		Class Members		
Public Attributes I Static Public Attributes I						

# RaiseEventOptions Class Reference

Public Attributes | Static Public Attributes | List of all members

Aggregates several less-often used options for operation RaiseEvent. See field descriptions for usage details. <u>More...</u>

#### **Public Attributes**

#### **EventCaching CachingOption**

Defines if the server should simply send the event, put it in the cache or remove events that are like this one. More...

#### byte InterestGroup

The number of the Interest Group to send this to. 0 goes to all users but to get 1 and up, clients must subscribe to the group first. More...

#### int[] TargetActors

A list of PhotonPlayer.IDs to send this event to. You can implement events that just go to specific users this way. <u>More...</u>

#### ReceiverGroup Receivers

Sends the event to All, MasterClient or Others (default). Be careful with MasterClient, as the client might disconnect before it got the event and it gets lost. More...

#### byte SequenceChannel

Events are ordered per "channel". If you have events that are independent of others, they can go into another sequence or channel. <u>More...</u>

#### bool ForwardToWebhook

Events can be forwarded to Webhooks, which can evaluate and use the events to follow the game's state. More...

#### bool Encrypt

### Static Public Attributes

### static readonly RaiseEventOptions

# Default = new RaiseEventOptions()

Default options:

Caching Option: DoNotCache, InterestGroup: 0, targetActors:

null, receivers: Others,

sequenceChannel: 0. More...

## **Detailed Description**

Aggregates several less-often used options for operation RaiseEvent. See field descriptions for usage details.

#### Member Data Documentation

#### **EventCaching RaiseEventOptions.CachingOption**

Defines if the server should simply send the event, put it in the cache or remove events that are like this one.

When using option: SliceSetIndex, SlicePurgeIndex or SlicePurgeUpToIndex, set a CacheSliceIndex. All other options except SequenceChannel get ignored.

## readonly RaiseEventOptions RaiseEventOptions.Default = new RaiseEventOptions()

stati<u>c</u>

Default options: CachingOption: DoNotCache, InterestGroup: 0, targetActors: null, receivers: Others, sequenceChannel: 0.

#### bool RaiseEventOptions.Encrypt

#### bool RaiseEventOptions.ForwardToWebhook

Events can be forwarded to Webhooks, which can evaluate and use the events to follow the game's state.

#### byte RaiseEventOptions.InterestGroup

The number of the Interest Group to send this to. 0 goes to all users but to get 1 and up, clients must subscribe to the group first.

#### ReceiverGroup RaiseEventOptions.Receivers

Sends the event to All, MasterClient or Others (default). Be careful with MasterClient, as the client might disconnect before it got the event and it gets lost.

#### byte RaiseEventOptions.SequenceChannel

Events are ordered per "channel". If you have events that are independent of others, they can go into another sequence or channel.

#### int [] RaiseEventOptions.TargetActors

A list of PhotonPlayer.IDs to send this event to. You can implement events that just go to specific users this way.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page	e Related	Pages	Modu	ıles	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	
					Public Member	er Functions
Region	Class			Sta	<u>atic Public Membe</u>	er Functions
Referen	000			<u>Public</u>	Attributes   List o	of all members
Kelelel	ICE					

## **Public Member Functions**

override string ToString ()

## Static Public Member Functions

static CloudRegionCode Parse (string codeAsString)

## Public Attributes

### CloudRegionCode Code

### string HostAndPort

### int **Ping**

## **Member Function Documentation**

static CloudRegionCode Region.Parse

(string codeAsString) static



override string Region.ToString ( )

### Member Data Documentation

CloudRegionCode Region.Code

string Region.HostAndPort

int Region.Ping

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page	е	Related	Pages	Modu	les	Classes	Files
Class List	С	lass Index	Class Hi	erarchy	Clas	ss Members	
RoomO	pt	ions Cl	ass			Public Attributes List	of all members
Referen	-						

Wraps up common room properties needed when you create rooms. Read the individual entries for more details. <u>More...</u>

#### **Public Attributes**

#### byte MaxPlayers

Max number of players that can be in the room at any time. 0 means "no limit". More...

#### int PlayerTtl

Time To Live (TTL) for an 'actor' in a room. If a client disconnects, this actor is inactive first and removed after this timeout. In milliseconds. More...

#### int EmptyRoomTtl

Time To Live (TTL) for a room when the last player leaves. Keeps room in memory for case a player re-joins soon. In milliseconds. More...

#### **Hashtable CustomRoomProperties**

The room's custom properties to set. Use string keys! More...

#### string[] CustomRoomPropertiesForLobby = new string[0]

Defines the custom room properties that get listed in the lobby. <u>More...</u>

#### string[] Plugins

Informs the server of the expected plugin setup. More...

## **Properties**

#### bool IsVisible [get, set]

Defines if this room is listed in the lobby. If not, it also is not joined randomly. <u>More...</u>

#### bool IsOpen [get, set]

Defines if this room can be joined at all. More...

#### bool CleanupCacheOnLeave [get, set]

Removes a user's events and properties from the room when a user leaves. More...

#### bool SuppressRoomEvents [get]

Tells the server to skip room events for joining and leaving players. <u>More...</u>

#### bool PublishUserId [get, set]

Defines if the UserIds of players get "published" in the room. Useful for FindFriends, if players want to play another game together. <u>More...</u>

bool isVisible [get, set]

bool isOpen [get, set]

byte maxPlayers [get, set]

bool cleanupCacheOnLeave [get, set]

Hashtable customRoomProperties [get, set]

string[] customRoomPropertiesForLobby [get, set]

```
string[] plugins [get, set]

bool suppressRoomEvents [get]

bool publishUserId [get, set]
```

## **Detailed Description**

Wraps up common room properties needed when you create rooms. Read the individual entries for more details.

This directly maps to the fields in the **Room** class.

#### Member Data Documentation

#### Hashtable RoomOptions.CustomRoomProperties

The room's custom properties to set. Use string keys!

Custom room properties are any key-values you need to define the game's setup. The shorter your keys are, the better. Example: Map, Mode (could be "m" when used with "Map"), TileSet (could be "t").

#### string [] RoomOptions.CustomRoomPropertiesForLobby = new string[0]

Defines the custom room properties that get listed in the lobby.

Name the custom room properties that should be available to clients that are in a lobby. Use with care. Unless a custom property is essential for matchmaking or user info, it should not be sent to the lobby, which causes traffic and delays for clients in the lobby.

Default: No custom properties are sent to the lobby.

#### int RoomOptions.EmptyRoomTtl

Time To Live (TTL) for a room when the last player leaves. Keeps room in memory for case a player re-joins soon. In milliseconds.

#### byte RoomOptions.MaxPlayers

Max number of players that can be in the room at any time. 0 means "no limit".

#### int RoomOptions.PlayerTtl

Time To Live (TTL) for an 'actor' in a room. If a client disconnects, this actor is inactive first and removed after this timeout. In milliseconds.

#### string [] RoomOptions.Plugins

Informs the server of the expected plugin setup.

The operation will fail in case of a plugin missmatch returning error code PluginMismatch 32757(0x7FFF - 10). Setting string[]{} means the client expects no plugin to be setup. Note: for backwards compatibility null omits any check.

### **Property Documentation**

#### bool RoomOptions.CleanupCacheOnLeave

get set

Removes a user's events and properties from the room when a user leaves.

This makes sense when in rooms where players can't place items in the room and just vanish entirely. When you disable this, the event history can become too long to load if the room stays in use indefinitely. Default: true. Cleans up the cache and props of leaving users.

#### bool RoomOptions.cleanupCacheOnLeave





Hashtable RoomOptions.customRoomProperties





string [] RoomOptions.customRoomPropertiesForLobby



#### bool RoomOptions.IsOpen





Defines if this room can be joined at all.

If a room is closed, no player can join this. As example this makes sense when 3 of 4 possible players start their gameplay early and don't want anyone to join during the game. The room can still be listed in the lobby (set IsVisible to control lobby-visibility).

#### bool RoomOptions.isOpen





bool RoomOptions.IsVisible





Defines if this room is listed in the lobby. If not, it also is not joined randomly.

A room that is not visible will be excluded from the room lists that are sent to the clients in lobbies. An invisible room can be joined by name but is excluded from random matchmaking.

Use this to "hide" a room and simulate "private rooms". Players can exchange a roomname and create it invisble to avoid anyone else joining it.

#### bool RoomOptions.isVisible



#### byte RoomOptions.maxPlayers









#### bool RoomOptions.PublishUserId





Defines if the UserIds of players get "published" in the room. Useful for FindFriends, if players want to play another game together.

When you set this to true, **Photon** will publish the UserIds of the players in that room. In that case, you can use **PhotonPlayer.UserId**, to access any player's userID. This is useful for FindFriends and to set "expected users" to reserve slots in a room (see PhotonNetwork.JoinRoom e.g.).

#### bool RoomOptions.publishUserId





#### bool RoomOptions.SuppressRoomEvents



Tells the server to skip room events for joining and leaving players.

Using this makes the client unaware of the other players in a room. That can save some traffic if you have some server logic that updates players but it can also limit the client's usability.

PUN will break if you use this, so it's not settable.

#### bool RoomOptions.suppressRoomEvents



Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

SceneManagerHelper Class Reference Properties | List of all members

## Properties

static string ActiveSceneName [get]

static int ActiveSceneBuildIndex [get]

## **Property Documentation**

int SceneManagerHelper.ActiveSceneBuildIndex

static get

string SceneManagerHelper.ActiveSceneName

static get

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



# Photon Unity Networking

v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

## ServerSettings Class Reference

Public Types | Public Member Functions |
Static Public Member Functions |
Public Attributes | Properties |
List of all members

Collection of connection-relevant settings, used internally by **PhotonNetwork.ConnectUsingSettings**. More...

Inherits ScriptableObject.

## **Public Types**

```
enum HostingOption {
    HostingOption.NotSet = 0, HostingOption.PhotonCloud =
    1, HostingOption.SelfHosted = 2,
    HostingOption.OfflineMode = 3,
    HostingOption.BestRegion = 4
}
```

## **Public Member Functions**

void	UseCloudBestRegion (string cloudAppid)
void	UseCloud (string cloudAppid)
void	<b>UseCloud</b> (string cloudAppid, <b>CloudRegionCode</b> code)
void	<b>UseMyServer</b> (string serverAddress, int serverPort, string application)
override string	ToString ()

### Static Public Member Functions

static bool IsAppId (string val)
Checks if a string is a Guid by attempting to create one. More...

static void ResetBestRegionCodeInPreferences ()

#### **Public Attributes**

string AppID = "" string VoiceAppID = "" string ChatAppID = "" **HostingOption HostType** = HostingOption.NotSet CloudRegionCode PreferredRegion CloudRegionFlag EnabledRegions = (CloudRegionFlag)(-1) ConnectionProtocol Protocol = ConnectionProtocol.Udp string ServerAddress = "" int **ServerPort** = 5055 int VoiceServerPort = 5055 bool JoinLobby bool EnableLobbyStatistics PhotonLogLevel PunLogging = PhotonLogLevel.ErrorsOnly DebugLevel NetworkLogging = DebugLevel.ERROR bool RunInBackground = true List< string > RpcList = new List<string>() bool DisableAutoOpenWizard

### **Properties**

#### static CloudRegionCode BestRegionCodeInPreferences [get]

Gets the best region code in preferences. This composes the PhotonHandler, since its Internal and can not be accessed by the custom inspector More...

#### static CloudRegionCode BestRegionCodeCurrently [get]

Gets the best region code currently. This composes the PhotonHandler, since its Internal and can not be accessed by the custom inspector <u>More...</u>

## **Detailed Description**

Collection of connection-relevant settings, used internally by **PhotonNetwork.ConnectUsingSettings**.

## Member Enumeration Documentation

enum ServerSe
Enumerator
NotSet
PhotonCloud
SelfHosted
OfflineMode
BestRegion

#### **Member Function Documentation**

## static bool ServerSettings.IsAppld (string val) static Checks if a string is a Guid by attempting to create one. **Parameters** val The potential guid to check. Returns True if new Guid(val) did not fail. static void ServerSettings.ResetBestRegionCodeInPreferences ( ) static override string ServerSettings.ToString ( ) void ServerSettings.UseCloud (string cloudAppid) void ServerSettings.UseCloud (string cloudAppid, CloudRegionCode code void ServerSettings.UseCloudBestRegion (string cloudAppid) void ServerSettings.UseMyServer (string serverAddress, int serverPort. string application

#### Member Data Documentation

string ServerSettings.AppID = ""

string ServerSettings.ChatAppID = ""

bool ServerSettings.DisableAutoOpenWizard

CloudRegionFlag ServerSettings.EnabledRegions = (CloudRegionFlag)(-1)

bool ServerSettings.EnableLobbyStatistics

**HostingOption ServerSettings.HostType = HostingOption.NotSet** 

bool ServerSettings.JoinLobby

DebugLevel ServerSettings.NetworkLogging = DebugLevel.ERROR

CloudRegionCode ServerSettings.PreferredRegion

ConnectionProtocol ServerSettings.Protocol = ConnectionProtocol.Udp

PhotonLogLevel ServerSettings.PunLogging = PhotonLogLevel.ErrorsOnly

List<string> ServerSettings.RpcList = new List<string>()

**bool ServerSettings.RunInBackground = true** 

string ServerSettings.ServerAddress = ""

int ServerSettings.ServerPort = 5055

string ServerSettings.VoiceAppID = ""

int ServerSettings.VoiceServerPort = 5055

### **Property Documentation**

#### CloudRegionCode ServerSettings.BestRegionCodeCurrently

static get

Gets the best region code currently. This composes the PhotonHandler, since its Internal and can not be accessed by the custom inspector

The best region code currently.

#### CloudRegionCode ServerSettings.BestRegionCodeInPreferences



Gets the best region code in preferences. This composes the PhotonHandler, since its Internal and can not be accessed by the custom inspector

The best region code in preferences.

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



v1.81

Main Page	e Relate	d Pages	Modu	les	Classes	Files
Class List	Class Index	Class H	ierarchy	Clas	ss Members	

# TypedLobby Class Reference

Public Member Functions | Public Attributes |
Static Public Attributes | Properties |
List of all members

Refers to a specific lobby (and type) on the server. More...
Inherited by **TypedLobbyInfo**.

# **Public Member Functions**

TypedLobby ()

TypedLobby (string name, LobbyType type)

override string ToString ()

### **Public Attributes**

#### string Name

Name of the lobby this game gets added to. Default: null, attached to default lobby. Lobbies are unique per lobbyName plus lobbyType, so the same name can be used when several types are existing. More...

#### LobbyType Type

Type of the (named)lobby this game gets added to More...

# Static Public Attributes

static readonly TypedLobby Default = new TypedLobby()

# Properties

bool IsDefault [get]

# **Detailed Description**

Refers to a specific lobby (and type) on the server.

The name and type are the unique identifier for a lobby. Join a lobby via **PhotonNetwork.JoinLobby(TypedLobby lobby)**. The current lobby is stored in **PhotonNetwork.lobby**.

# Constructor & Destructor Documentation

```
TypedLobby.TypedLobby()
```

```
TypedLobby.TypedLobby ( string name, LobbyType type )
```

# **Member Function Documentation**

override string TypedLobby.ToString ( )

#### Member Data Documentation

#### readonly TypedLobby TypedLobby.Default = new TypedLobby()

static

#### string TypedLobby.Name

Name of the lobby this game gets added to. Default: null, attached to default lobby. Lobbies are unique per lobbyName plus lobbyType, so the same name can be used when several types are existing.

#### LobbyType TypedLobby.Type

Type of the (named)lobby this game gets added to

# **Property Documentation**

### bool TypedLobby.IsDefault



Online Documentation - Dashboard - Support Forum

Exit Games GmbH



v1.81

Main Page	Related	d Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	ierarchy	Clas	ss Members	

# TypedLobbyInfo Class Reference

Public Member Functions | Public Attributes |
List of all members

Inherits **TypedLobby**.

## **Public Member Functions**

override string ToString ()

► Public Member Functions inherited from TypedLobby TypedLobby ()

TypedLobby (string name, LobbyType type)

override string ToString ()

#### **Public Attributes**

#### int PlayerCount

#### int RoomCount

#### ▶ Public Attributes inherited from TypedLobby

#### string Name

Name of the lobby this game gets added to. Default: null, attached to default lobby. Lobbies are unique per lobbyName plus lobbyType, so the same name can be used when several types are existing. More...

#### LobbyType Type

Type of the (named)lobby this game gets added to More...

## **Additional Inherited Members**

- Static Public Attributes inherited from TypedLobby static readonly TypedLobby Default = new TypedLobby()
- Properties inherited from TypedLobby bool IsDefault [get]

# **Member Function Documentation**

override string TypedLobbyInfo.ToString ( )

## Member Data Documentation

int TypedLobbyInfo.PlayerCount

int TypedLobbyInfo.RoomCount

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



v1.81

Main Page	Related	d Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	ierarchy	Clas	ss Members	

# WebRpcResponse Class Reference

Public Member Functions | Properties | List of all members

Reads an operation response of a WebRpc and provides convenient access to most common values. More...

## **Public Member Functions**

### WebRpcResponse (OperationResponse response)

An OperationResponse for a WebRpc is needed to read it's values. More...

### string ToStringFull ()

Turns the response into an easier to read string. More...

# **Properties**

#### string Name [get, set]

Name of the WebRpc that was called. More...

#### int ReturnCode [get, set]

ReturnCode of the WebService that answered the WebRpc. More...

#### string DebugMessage [get, set]

Might be empty or null. More...

#### Dictionary< string, object > Parameters [get, set]

Other key/values returned by the webservice that answered the WebRpc. More...

# **Detailed Description**

Reads an operation response of a WebRpc and provides convenient access to most common values.

See method **PhotonNetwork.WebRpc**. Create a **WebRpcResponse** to access common result values. The operationResponse.OperationCode should be: **OperationCode.WebRpc**.

# Constructor & Destructor Documentation

### WebRpcResponse.WebRpcResponse (OperationResponse response response

An OperationResponse for a WebRpc is needed to read it's values.

# **Member Function Documentation**

#### string WebRpcResponse.ToStringFull ( )

Turns the response into an easier to read string.

#### **Returns**

String resembling the result.

## **Property Documentation**

#### string WebRpcResponse.DebugMessage

get set

Might be empty or null.

#### string WebRpcResponse.Name

get set

Name of the WebRpc that was called.

#### **Dictionary**<string, object> WebRpcResponse.Parameters





Other key/values returned by the webservice that answered the WebRpc.

#### int WebRpcResponse.ReturnCode





ReturnCode of the WebService that answered the WebRpc.

0 is commonly used to signal success.

-1 tells you: Got no ReturnCode from WebRpc service. Other ReturnCodes are defined by the individual WebRpc and service.



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### Class Index

#### $A \mid C \mid E \mid F \mid G \mid H \mid I \mid M \mid O \mid P \mid R \mid S \mid T \mid W$



# ActorProperties AuthenticationValues (ExitGames.Client.Photon.Chat) AuthenticationValues





ChatChannel (ExitGames.Client.Photon.Chat)
ChatClient (ExitGames.Client.Photon.Chat)
ChatEventCode (ExitGames.Client.Photon.Chat)
ChatOperationCode (ExitGames.Client.Photon.Chat)
ChatParameterCode (ExitGames.Client.Photon.Chat)
ChatPeer (ExitGames.Client.Photon.Chat)
ChatUserStatus (ExitGames.Client.Photon.Chat)







**IChatClier** 



**ErrorCode** 

#### $A \mid C \mid E \mid F \mid G \mid H \mid I \mid M \mid O \mid P \mid R \mid S \mid T \mid W$



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

# **Class Hierarchy**

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

	il level 123]
	_
♠ ActorProperties	Class for define "we Actor / Plainternally."
<b>▼                                    </b>	
(7) HelpURL	Empty implements implementation impl
PunRPC	Replacem different n remote-ca
■ ExitGames.Client.Photon.Chat.AuthenticationValues	Container Photon. S connect -
AuthenticationValues	Container Photon. S connect -
© ExitGames.Client.Photon.Chat.ChatChannel	A channel <b>Chat</b> , upd provided a
■ ExitGames.Client.Photon.Chat.ChatEventCode	Wraps up

	Photon C use them
■ ExitGames.Client.Photon.Chat.ChatOperationCode	Wraps up internally to use the
♠ ExitGames.Client.Photon.Chat.ChatParameterCode	Wraps up operations Photon C directly us
♠ ExitGames.Client.Photon.Chat.ChatUserStatus	Contains of for SetOn own.
EncryptionDataParameters	
© ExitGames.Client.Photon.Chat.ErrorCode	ErrorCod associated communic
♠ ErrorCode	ErrorCod associated communic
	Class for events de Pun uses
© Extensions	This static extension classes (e
FriendInfo	Used to state and
<b>⊚</b> GameObjectExtensions	Small nun make it ea Unity-vers
<b>⊚</b> GamePropertyKey	Class for care for "wo properties Loadbalar internally.

<b>⊚</b> ExitGames.Client.GUI.GizmoTypeDrawer	
■ ExitGames.Client.Photon.Chat.IChatClientListener	Callback i Contains ( app about new <b>Chat</b>
<b>▼                                    </b>	
PhotonPlayer	Summariz identified
<b>▼                                    </b>	
PhotonPlayer	Summariz identified
<b>▼                                    </b>	
PhotonPlayer	Summariz identified
<b>▼                                    </b>	
PhotonPlayer	Summariz identified
<b>▼ ©</b> IPhotonPeerListener	
© ExitGames.Client.Photon.Chat.ChatClient	Central cla connect, h
▼	This interf callback n OnPhoton implemen
Photon.PunBehaviour	This class callbacks/ Override t use.
<b>▼ ©</b> IPunObservable	Defines th method to correctly f
PhotonAnimatorView	This class Mecanim componer

	_
	make sure
	is added t
	componer
PhotonRigidbody2DView	This class
	velocities
	that only t
	and becau
	determinis
	the same
	positions (
	sync. If yc
	this object
	should als
	PhotonTr
	the positic
	your Gam
	PhotonRi
	the list of
PhotonRigidbodyView	This class
	velocities
	that only t
	and becau
	determinis
	the same
	positions
	sync. If yo
	this object
	should als
	PhotonTr
	the positic
	your Gam
	PhotonRi
	list of obs
PhotonTransformView	This class
	position, r
	GameObj
	different o

	synchronic even when of times p componer make sure <b>PhotonTr</b> list of obse Defines al Pool must use it.
<b>▼                                    </b>	
	This class velocities that only that only the and becaute determinis the same positions of sync. If you this object should als <b>PhotonTr</b> the position your Gam <b>PhotonRi</b> the list of the second should als the list of the second should als the second should shoul
	This class velocities that only t and becaudeterminis the same positions caync. If yo this object should als

	PhotonTr the positic your Gam PhotonRi list of obse
PhotonStatsGui	Basic GUI statistics (toggled by
♠ PhotonTransformView	This class position, r GameObjectiferent of synchronic even when of times promponer make sure <b>PhotonTr</b> list of observables.
<b>▼ ©</b> MonoBehaviour	
<b>▼                                    </b>	This class while logg still uses t
Photon.PunBehaviour	This class callbacks/ Override tuse.
PhotonView	PUN's Ne for networ
<b>○</b> PhotonAnimatorView	This class Mecanim componer make sure is added t componer
PhotonLagSimulationGui	This Mond

	the <b>Photo</b> feature. It jitter (rand
OperationCode	Class for codes. Pu internally.
ExitGames.Client.Photon.Chat.ParameterCode	
ParameterCode	Class for oparameter Pun uses
PhotonMessageInfo	Container message,
PhotonNetwork	The main <b>PhotonN</b> static.
<b>▼ (a)</b> PhotonPeer	
© ExitGames.Client.Photon.Chat.ChatPeer	Provides I Chat serv public Cha
▼	
<b>☐</b> PingMonoEditor	Uses C# S System.N does).
PhotonPingManager	
PhotonStream	This conta OnPhoto provide in or for you
PhotonStreamQueue	The <b>Phot</b> object star what <b>Pho</b> and then swhen <b>Ser</b> receiving and then t

	received c
	and timeS
PhotonTransformViewPositionControl	
PhotonTransformViewPositionModel	
PhotonTransformViewRotationControl	
PhotonTransformViewRotationModel	
PhotonTransformViewScaleControl	
PhotonTransformViewScaleModel	
<b>□</b> RaiseEventOptions	Aggregate options fo field descri
© Region	
▼	A simplifie required to listing in the settable (c
Room	This class joins (or joins ettable a <b>RoomInfo</b> "your" roo
<b>©</b> RoomOptions	Wraps up needed w individual
■ UnityEngine.SceneManagement.SceneManager	Minimal in <b>SceneMa</b> v5.2.
<b>☐</b> SceneManagerHelper	
<b>▼                                    </b>	
G ServerSettings	Collection used inter
PhotonAnimatorView.SynchronizedLayer	
PhotonAnimatorView.SynchronizedParameter	

<b>▼                                    </b>	Refers to the server
<b>☐</b> TypedLobbyInfo	
<b>@</b> WebRpcResponse	Reads an WebRpc & to most co
Online Documentation - Dashboard - Support Forum	Exit Games GmbH

Online Documentation - Dashboard - Support Forum Exit Games GmbH



v1.81

Main Page R						Rel	ate	d F	Paç	jes	5	Modules					Cla	SS	es		File	es
Class List Class					s In	dex		Class Hierarchy Class Members														
P	AII		Fund	ction	าร		Variables E					numerations					Enumerator					
Properties																						
a	b	С	d	е	f	g	h	i	j	1	m	n	O	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

#### - a -

- ActiveSceneBuildIndex : SceneManagerHelper
- ActiveSceneName : SceneManagerHelper
- ActorList : ParameterCode
- ActorNr : ParameterCode
- Add(): ExitGames.Client.Photon.Chat.ChatChannel , ParameterCode
- AddAuthParameter(): AuthenticationValues ,
   ExitGames.Client.Photon.Chat.AuthenticationValues
- AddFriends(): ExitGames.Client.Photon.Chat.ChatClient, ExitGames.Client.Photon.Chat.ChatOperationCode
- Address: ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode
- AllocateSceneViewID(): PhotonNetwork
- AllocateViewID(): PhotonNetwork
- AllProperties : PhotonPlayer
- allProperties : PhotonPlayer
- AlmostEquals(): Extensions
- AlreadyMatched : ErrorCode
- Appld : ExitGames.Client.Photon.Chat.ChatClient
- AppID : ServerSettings

- ApplicationId : ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode
- AppStats : EventCode
- AppVersion: ExitGames.Client.Photon.Chat.ChatClient, ExitGames.Client.Photon.Chat.ParameterCode, ParameterCode
- Attempts : PhotonPingManager
- Authenticate : ExitGames.Client.Photon.Chat.ChatOperationCode , OperationCode
- AuthenticateOnce : OperationCode
- AuthenticateOnNameServer():
   ExitGames.Client.Photon.Chat.ChatPeer
- AuthenticationTicketExpired : ErrorCode
- AuthenticationValues(): AuthenticationValues ,
   ExitGames.Client.Photon.Chat.AuthenticationValues
- AuthEvent : EventCode
- AuthGetParameters: AuthenticationValues,
   ExitGames.Client.Photon.Chat.AuthenticationValues
- AuthPostData: AuthenticationValues,
   ExitGames.Client.Photon.Chat.AuthenticationValues
- AuthType: AuthenticationValues,
   ExitGames.Client.Photon.Chat.AuthenticationValues
- AuthValues: ExitGames.Client.Photon.Chat.ChatClient, PhotonNetwork
- AutoCleanUp : Room
- autoCleanUp : Room
- autoCleanUpField : RoomInfo
- autoCleanUpPlayerObjects : PhotonNetwork
- autoJoinLobby : PhotonNetwork
- automaticallySyncScene : **PhotonNetwork**
- Away : ExitGames.Client.Photon.Chat.ChatUserStatus
- AzureLocalNodeld: ParameterCode
- AzureMasterNodeld : ParameterCode
- AzureNodeInfo : EventCode , ParameterCode



v1.81

Main Page R						Rel	Related Pages						Modules					SS	es		File	es
Class List Class					s In	dex	Class Hierarchy Class Members								5							
P	AII		Fund	ction	าร	Variables Enumerations Enu							ıme	rato	r							
Properties																						
a	b	С	d	е	f	g	h	i	j	I	m	n	o	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

#### - b -

- BackgroundTimeout : PhotonNetwork
- BestRegion : PhotonPingManager , ServerSettings
- BestRegionCodeCurrently: ServerSettings
- BestRegionCodeInPreferences : ServerSettings
- Bool : PhotonAnimatorView
   Broadcast : ParameterCode
   buttonsOn : PhotonStatsGui

Online Documentation - Dashboard - Support Forum

Exit Games GmbH



v1.81

ı	Mai	n P	age	Э	ı	Rel	ate	d F	Paç	jes	5	M	odı	ıles	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	Class Index ctions Varial					Cl	ass	Hie	rarc	hy		Clas	ss M	leml	oers	5			
F	All	ı	Fund	ction	าร		Vari	ablo	es		Enu	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	1	m	n	O	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

#### - C -

- Cache: ParameterCode
- CacheDiscreteTriggers(): PhotonAnimatorView
- CacheSendMonoMessageTargets(): PhotonNetwork
- CacheSliceChanged : EventCode
- CacheSliceIndex: ParameterCode
- CachingOption : RaiseEventOptions
- CanChat: ExitGames.Client.Photon.Chat.ChatClient
- CanChatInChannel():
  - ExitGames.Client.Photon.Chat.ChatClient
- ChangeGroups : OperationCode
- Channel : ExitGames.Client.Photon.Chat.ChatParameterCode
- ChannelHistory:
  - ${\bf Exit Games. Client. Photon. Chat. Chat Operation Code}$
- Channels : ExitGames.Client.Photon.Chat.ChatParameterCode
- ChannelUserCount:
  - ExitGames.Client.Photon.Chat.ChatParameterCode
- ChatAppID : ServerSettings
- ChatChannel(): ExitGames.Client.Photon.Chat.ChatChannel
- ChatClient(): ExitGames.Client.Photon.Chat.ChatClient
- ChatMessages:

### ExitGames.Client.Photon.Chat.ChatEventCode

- chatPeer : ExitGames.Client.Photon.Chat.ChatClient
- ChatPeer(): ExitGames.Client.Photon.Chat.ChatPeer
- ChatRegion : ExitGames.Client.Photon.Chat.ChatClient
- CheckUserOnJoin: ParameterCode
- CleanupCacheOnLeave : GamePropertyKey , ParameterCode , RoomOptions
- cleanupCacheOnLeave : RoomOptions
- ClearExpectedUsers(): Room
- ClearMessages(): ExitGames.Client.Photon.Chat.ChatChannel
- ClientAuthenticationData :

ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode

ClientAuthenticationParams:

ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode

ClientAuthenticationType :

ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode

- CloseConnection(): PhotonNetwork
- Code : ParameterCode , Region
- CompareTo(): PhotonPlayer
- Connect(): ExitGames.Client.Photon.Chat.ChatClient, ExitGames.Client.Photon.Chat.ChatPeer
- connected : PhotonNetwork
- connectedAndReady : PhotonNetwork
- connecting : PhotonNetwork
- connectionState : PhotonNetwork
- connectionStateDetailed : PhotonNetwork
- ConnectToBestCloudServer(): PhotonNetwork
- ConnectToMaster(): PhotonNetwork
- ConnectToRegion(): PhotonNetwork
- ConnectUsingSettings(): PhotonNetwork
- Contains(): Extensions
- Continuous : PhotonAnimatorView
- Count : PhotonStream
- countOfPlayers : PhotonNetwork
- countOfPlayersInRooms : PhotonNetwork
- countOfPlayersOnMaster : PhotonNetwork
- countOfRooms : PhotonNetwork

- CrcCheckEnabled : PhotonNetwork
- CreateGame : OperationCode
- CreateRoom(): PhotonNetwork
- CreatorActorNr : PhotonView
- currentMasterID : PhotonView
- CustomAuthenticationFailed : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- CustomEventContent : ParameterCode
- CustomInitData: ParameterCode
- CustomProperties : PhotonPlayer
- customProperties : PhotonPlayer
- CustomProperties : RoomInfo
- customProperties : RoomInfo
- CustomRoomProperties : RoomOptions
- customRoomProperties : RoomOptions
- CustomRoomPropertiesForLobby : RoomOptions
- customRoomPropertiesForLobby : RoomOptions

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	5	M	odı	ıle	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	Class Index					CI	lass	Hie	rarc	hy		Clas	ss M	leml	oers	5			
P	AII		Fund	ction	าร		Vari	abl	es		Enu	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	1	m	n	0	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

### - d -

- Data : ParameterCode
- DebugMessage : WebRpcResponse
- DebugOut : ExitGames.Client.Photon.Chat.ChatClient
- DebugReturn():
  - ExitGames.Client.Photon.Chat.IChatClientListener
- Default : RaiseEventOptions , TypedLobby
- Deserialize(): PhotonStreamQueue
- DeserializeView(): PhotonView
- Destroy(): IPunPrefabPool, PhotonNetwork
- DestroyAll(): PhotonNetwork
- DestroyPlayerObjects(): PhotonNetwork
- DisableAutoOpenWizard : ServerSettings
- Disabled : PhotonAnimatorView ,
  - PhotonTransformViewPositionModel .
  - PhotonTransformViewRotationModel .
  - **PhotonTransformViewScaleModel**
- Disconnect(): ExitGames.Client.Photon.Chat.ChatClient, PhotonNetwork
- DisconnectedCause : ExitGames.Client.Photon.Chat.ChatClient
- Discrete: PhotonAnimatorView

- Dispose(): PingMonoEditor
- DND : ExitGames.Client.Photon.Chat.ChatUserStatus
- DoesLayerSynchronizeTypeExist(): PhotonAnimatorView
- DoesParameterSynchronizeTypeExist(): PhotonAnimatorView
- Done: PhotonPingManager, PingMonoEditor
- Draw(): ExitGames.Client.GUI.GizmoTypeDrawer
- DrawErrorGizmo: PhotonTransformViewPositionModel

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	ag	е	F	Rel	ate	d F	Paç	jes	5	M	odı	ıles	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	C	Clas	s In	dex		CI	ass	Hie	rarc	hy		Clas	s M	eml	oers	3			
P	AII		Fund	ction	าร		Vari	abl	es		Enu	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	es																			
a	b	С	d	е	f	g	h	i	j	I	m	n	o	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

#### - e -

- EmptyRoomTTL : ParameterCode
- EmptyRoomTtl : RoomOptions
- EnabledRegions : ServerSettings
- EnableLobbyStatistics : PhotonNetwork , ServerSettings
- Encrypt : RaiseEventOptions
- EncryptionData : ParameterCode
- EncryptionMode : ParameterCode
- Equals(): PhotonPlayer, RoomInfo
- ErrorInfo : EventCode
- EstimatedSpeed : PhotonTransformViewPositionModel
- EstimateSpeedAndTurn : PhotonTransformViewPositionModel
- EventCallback() : PhotonNetwork
- EventForward : ParameterCode
- ExchangeKeysForEncryption : **OperationCode**
- ExpectedProtocol : ParameterCode
- ExpectedUsers : **GamePropertyKey**
- expectedUsers : Room
- ExpectedUsers : Room
- expectedUsersField : RoomInfo
- ExpectedValues : ParameterCode

- ExternalHttpCallFailed : ErrorCode
- ExtrapolateIncludingRoundTripTime : PhotonTransformViewPositionModel
- ExtrapolateNumberOfStoredPositions:
   PhotonTransformViewPositionModel
- ExtrapolateOption: PhotonTransformViewPositionModel
   ExtrapolateOptions: PhotonTransformViewPositionModel
   ExtrapolateSpeed: PhotonTransformViewPositionModel

Online Decumentation Dechhoord Cuppert Forum Evit Comes Cr

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	5	M	odı	ıle	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	Class Index unctions Varial					CI	ass	Hie	rarc	hy		Clas	ss M	leml	oers	•			
P	AII		Fund	ction	าร		Vari	ablo	es		Enu	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	1	m	n	О	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

### - f -

- FetchServerTimestamp(): PhotonNetwork
- Find(): PhotonPlayer, PhotonView
- FindFriends : OperationCode , PhotonNetwork
- FindFriendsRequestList : ParameterCode
- FindFriendsResponseOnlineList : ParameterCode
- FindFriendsResponseRoomIdList : ParameterCode
- FindGameObjectsWithComponent(): PhotonNetwork
- FixedSpeed : PhotonTransformViewPositionModel
- Float : PhotonAnimatorView
- ForwardToWebhook : RaiseEventOptions
- Friends: ExitGames.Client.Photon.Chat.ChatParameterCode, PhotonNetwork
- FriendsList : ExitGames.Client.Photon.Chat.ChatEventCode
- FriendsListAge : **PhotonNetwork**
- FrontendAddress : ExitGames.Client.Photon.Chat.ChatClient



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Pag	jes	5	M	odı	ıle	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	C	Clas	s In	dex		CI	ass	Hie	rarc	hy		Clas	s M	leml	oers	3			
P	All	ı	Fund	ction	าร		Vari	abl	es		Enι	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s							,												
a	b	С	d	е	f	g	h	i	j	ı	m	n	0	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

### - g -

- GameClosed : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- GameCount : ParameterCode
- GameDoesNotExist : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- GameFull : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- GameIdAlreadyExists: ErrorCode, ExitGames.Client.Photon.Chat.ErrorCode
- GameList : EventCode , ParameterCode
- GameListUpdate : EventCode
- GameProperties : ParameterCode
- gameVersion : PhotonNetwork
- Get(): PhotonPlayer, PhotonView
- GetActive(): GameObjectExtensions
- GetCachedParemeters(): Extensions
- GetCustomRoomList(): PhotonNetwork
- GetExtrapolatedPositionOffset():
   PhotonTransformViewPositionControl
- GetGameList : OperationCode

- GetHashCode(): PhotonPlayer , RoomInfo
- GetLayerSynchronizeType(): PhotonAnimatorView
- GetLobbyStats : OperationCode
- GetNetworkPosition(): PhotonTransformViewPositionControl
- GetNetworkRotation(): PhotonTransformViewRotationControl
- GetNetworkScale(): PhotonTransformViewScaleControl
- GetNext(): PhotonPlayer
- GetNextFor(): PhotonPlayer
- GetParameterSynchronizeType(): PhotonAnimatorView
- GetPhotonView(): Extensions
- GetPhotonViewsInChildren(): Extensions
- GetPing(): PhotonNetwork
- GetPrivateChannelNameByUser() : ExitGames.Client.Photon.Chat.ChatClient
- GetProperties : OperationCode
- GetRegions : OperationCode
- GetRoomList(): PhotonNetwork
- GetRotation(): PhotonTransformViewRotationControl
- GetScale(): PhotonTransformViewScaleControl
- GetSynchronizedLayers(): PhotonAnimatorView
- GetSynchronizedParameters(): PhotonAnimatorView
- Group : ParameterCode
- group : PhotonView

Online Documentation - Dashboard - Support Forum



v1.81

ı	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	5	M	odı	ıles	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	C	Clas	s In	dex		Cl	ass	Hie	rarc	hy		Clas	s M	leml	oers	5			
F	AII		Fund	ction	าร		Vari	ablo	es		Enu	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	1	m	n	o	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

### - h -

• HasQueuedObjects(): PhotonStreamQueue

• healthStatsVisible : PhotonStatsGui

• HelpURL(): HelpURL

• HistoryLength :

ExitGames.Client.Photon.Chat.ChatParameterCode

• HostAndPort : Region

HostingOption : ServerSettings

• HostType : ServerSettings

• HttpLimitReached : ErrorCode

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Pag	jes	5	M	odı	ıles	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	Class Index					CI	ass	Hie	rarc	hy		Clas	ss M	leml	oers	5			
P	AII		Fund	ction	าร		Vari	abl	es		Enu	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	I	m	n	О	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

### - i -

- ID : PhotonPlayer
- IgnoreInitialAttempt : PhotonPingManager
- Info: ParameterCode
- InitializeSecurity(): PhotonNetwork
- inRoom : PhotonNetwork
- insideLobby : PhotonNetwork
- Instantiate(): IPunPrefabPool, PhotonNetwork
- InstantiateInRoomOnly : PhotonNetwork
- InstantiateSceneObject(): PhotonNetwork
- instantiationData : PhotonView
- instantiationId : PhotonView
- Int : PhotonAnimatorView
- InterestGroup : RaiseEventOptions
- InternalServerError: ErrorCode,
   ExitGames.Client.Photon.Chat.ErrorCode
- InterpolateLerpSpeed: PhotonTransformViewPositionModel, PhotonTransformViewRotationModel, PhotonTransformViewScaleModel
- InterpolateMoveTowardsAcceleration : PhotonTransformViewPositionModel

- InterpolateMoveTowardsDeceleration : PhotonTransformViewPositionModel
- InterpolateMoveTowardsSpeed : PhotonTransformViewPositionModel , PhotonTransformViewScaleModel
- InterpolateOption: PhotonTransformViewPositionModel, PhotonTransformViewRotationModel, PhotonTransformViewScaleModel
- InterpolateOptions: PhotonTransformViewPositionModel, PhotonTransformViewRotationModel, PhotonTransformViewScaleModel
- InterpolateRotateTowardsSpeed : PhotonTransformViewRotationModel
- InterpolateSpeedCurve : PhotonTransformViewPositionModel
- InvalidAuthentication : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- InvalidEncryptionParameters : ErrorCode
- InvalidOperation : ErrorCode
- InvalidOperationCode : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- InvalidRegion : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- Invisible : ExitGames.Client.Photon.Chat.ChatUserStatus
- IsAppld(): ServerSettings
- IsComingBack : ParameterCode
- IsDefault : TypedLobby
- Islnactive : ActorProperties , ParameterCode
- isInactive : PhotonPlayer
- Islnactive : PhotonPlayer
- IsInRoom : FriendInfo
- isLocal : PhotonPlayer
- IsLocal : PhotonPlayer
- isLocalClientInside : RoomInfo
- IsLocalClientInside : RoomInfo
- isMasterClient : PhotonNetwork , PhotonPlayer
- IsMasterClient : PhotonPlayer
- isMessageQueueRunning : PhotonNetwork
- isMine : PhotonView
- isNonMasterClientInRoom : **PhotonNetwork**
- IsOnline: FriendInfo

• IsOpen: GamePropertyKey, Room, RoomInfo

isOpen : RoomOptionsIsOpen : RoomOptions

• isOwnerActive : PhotonView

• IsPrivate: ExitGames.Client.Photon.Chat.ChatChannel

isReading : PhotonStreamisSceneView : PhotonView

• IsVisible : GamePropertyKey , Room , RoomInfo

isVisible : RoomOptions
 IsVisible : RoomOptions
 isWriting : PhotonStream

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	6	M	odı	ıle	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	Class Index					C	lass	Hie	rarc	hy		Clas	ss M	leml	oers	5			
P	AII		Fund	ction	าร		Vari	ablo	es		Enι	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	0	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

### - j -

- Join : EventCode , OperationCode
- JoinFailedFoundActiveJoiner : ErrorCode
- JoinFailedFoundExcludedUserId : ErrorCode
- JoinFailedFoundInactiveJoiner : ErrorCode
- JoinFailedPeerAlreadyJoined : ErrorCode
- JoinFailedWithRejoinerNotFound : ErrorCode
- JoinGame : OperationCode
- JoinLobby : OperationCode , PhotonNetwork , ServerSettings
- JoinMode : ParameterCode
- JoinOrCreateRoom(): PhotonNetwork
- JoinRandomGame : OperationCode
- JoinRandomRoom(): PhotonNetwork
- JoinRoom(): PhotonNetwork



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	5	M	odı	ıle	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	Class Index Inctions Varial					Cl	ass	Hie	rarc	hy		Clas	ss M	leml	oers	•			
P	AII		Fund	ction	าร		Vari	ablo	es		Enu	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	I	m	n	0	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

### - 1 -

- LayerIndex : PhotonAnimatorView.SynchronizedLayer
- Leave : EventCode , OperationCode
- LeaveLobby : OperationCode , PhotonNetwork
- LeaveRoom(): PhotonNetwork
- Lerp: PhotonTransformViewPositionModel, PhotonTransformViewRotationModel, PhotonTransformViewScaleModel
- LFG : ExitGames.Client.Photon.Chat.ChatUserStatus
- LoadLevel(): PhotonNetwork
- LoadScene(): UnityEngine.SceneManagement.SceneManager
- lobby : PhotonNetwork
- LobbyName : ParameterCode
- LobbyStatistics : PhotonNetwork
- LobbyStats : EventCode , ParameterCode
- LobbyType : ParameterCode
- logLevel : PhotonNetwork



v1.81

N	Mai	n P	ag	е	ı	Rel	ate	d F	Paç	jes	5	М	odı	ıle	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	Class Index unctions Varial				CI	ass	Hie	rarc	hy		Clas	s M	lemb	oers	3				
P	AII		Fund	ction	าร		Vari	abl	es		Enu	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	es																			
a	b	С	d	е	f	g	h	i	j	I	m	n	О	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

#### - m -

- masterClient : PhotonNetwork
- MasterClientId : GamePropertyKey , ParameterCode
- MasterPeerCount : ParameterCode
- Match : EventCode
- MatchMakingType : ParameterCode
- MAX\_VIEW\_IDS : PhotonNetwork
- MaxCcuReached : ErrorCode ,
  - ExitGames.Client.Photon.Chat.ErrorCode
- maxConnections : PhotonNetwork
- MaxMilliseconsPerPing : PhotonPingManager
- MaxPlayers : GamePropertyKey
- maxPlayers : Room
- MaxPlayers : Room
- maxPlayers : RoomInfo
- MaxPlayers : RoomInfo
- maxPlayers : RoomOptions
- MaxPlayers : RoomOptions
- maxPlayersField : RoomInfo
- MaxResendsBeforeDisconnect : PhotonNetwork
- Merge(): Extensions

- MergeStringKeys(): Extensions
- Message : ExitGames.Client.Photon.Chat.ChatParameterCode
- MessageCount : ExitGames.Client.Photon.Chat.ChatChannel
- MessageLimit: ExitGames.Client.Photon.Chat.ChatChannel, ExitGames.Client.Photon.Chat.ChatClient
- Messages: ExitGames.Client.Photon.Chat.ChatChannel, ExitGames.Client.Photon.Chat.ChatParameterCode
- Mode : EncryptionDataParameters
- MoveTowards : PhotonTransformViewScaleModel
- Msgld : ExitGames.Client.Photon.Chat.ChatParameterCode
- Msglds: ExitGames.Client.Photon.Chat.ChatParameterCode

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	5	М	odı	ıle	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	Class Index					Cl	ass	Hie	rarc	hy		Clas	ss M	leml	oers	•			
P	AII		Fund	ction	าร		Vari	abl	es		Enu	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	1	m	n	0	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

#### - n -

- Name: ExitGames.Client.Photon.Chat.ChatChannel,
   FriendInfo, PhotonAnimatorView.SynchronizedParameter
- name : PhotonPlayer , Room
- Name: Room, RoomInfo
- name: RoomInfo
- Name : TypedLobby , WebRpcResponse
- nameField : RoomInfo
- NameServerAddress :
  - ExitGames.Client.Photon.Chat.ChatClient, ExitGames.Client.Photon.Chat.ChatPeer
- NameServerHost : ExitGames.Client.Photon.Chat.ChatPeer
- NameServerHttp: ExitGames.Client.Photon.Chat.ChatPeer
- NetworkLogging : ServerSettings
- NetworkStatisticsEnabled : PhotonNetwork
- NetworkStatisticsReset(): PhotonNetwork
- NetworkStatisticsToString() : PhotonNetwork
- networkView : Photon.MonoBehaviour
- NickName: ParameterCode, PhotonPlayer
- NoRandomMatchFound : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode

• NotSet : ServerSettings

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	5	M	odı	ıle	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	Class Index					CI	lass	Hie	rarc	hy		Clas	ss M	leml	oers	•			
P	AII		Fund	ction	าร		Vari	abl	es		Enu	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	1	m	n	О	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

#### - 0 -

- ObservedComponents : PhotonView
- Offline: ExitGames.Client.Photon.Chat.ChatUserStatus
- offlineMode : PhotonNetwork
- OfflineMode : ServerSettings
- $\bullet \quad \mathsf{Ok} : \textbf{ErrorCode} \;, \; \textbf{ExitGames.Client.Photon.Chat.ErrorCode}$
- OnChatStateChange():
  - ExitGames.Client.Photon.Chat.IChatClientListener
- OnConnected():
  - ExitGames.Client.Photon.Chat.IChatClientListener
- OnConnectedToMaster(): IPunCallbacks , Photon.PunBehaviour
- OnConnectedToPhoton(): IPunCallbacks, Photon.PunBehaviour
- OnConnectionFail(): IPunCallbacks, Photon.PunBehaviour
- OnCreatedRoom(): IPunCallbacks, Photon.PunBehaviour
- OnCustomAuthenticationFailed(): IPunCallbacks , Photon.PunBehaviour
- OnCustomAuthenticationResponse(): IPunCallbacks , Photon.PunBehaviour
- OnDisconnected():

### ExitGames.Client.Photon.Chat.IChatClientListener

- OnDisconnectedFromPhoton(): IPunCallbacks , Photon.PunBehaviour
- OnEventCall : PhotonNetwork
- OnFailedToConnectToPhoton(): IPunCallbacks , Photon.PunBehaviour
- OnGetMessages():
   ExitGames.Client.Photon.Chat.IChatClientListener
- OnGUI(): PhotonLagSimulationGui, PhotonStatsGui
- OnJoinedLobby(): IPunCallbacks, Photon.PunBehaviour
- OnJoinedRoom(): IPunCallbacks, Photon.PunBehaviour
- OnLeftLobby(): IPunCallbacks, Photon.PunBehaviour
- OnLeftRoom(): IPunCallbacks, Photon.PunBehaviour
- Online: ExitGames.Client.Photon.Chat.ChatUserStatus
- OnLobbyStatisticsUpdate(): IPunCallbacks ,
   Photon.PunBehaviour
- OnMasterClientSwitched(): IPunCallbacks ,
   Photon.PunBehaviour , PhotonView
- OnOwnershipRequest(): IPunCallbacks, Photon.PunBehaviour
- OnOwnershipTransfered(): IPunCallbacks, Photon.PunBehaviour
- OnPhotonCreateRoomFailed(): IPunCallbacks,
   Photon.PunBehaviour
- OnPhotonCustomRoomPropertiesChanged(): IPunCallbacks ,
   Photon.PunBehaviour
- OnPhotonInstantiate(): IPunCallbacks, Photon.PunBehaviour
- OnPhotonJoinRoomFailed(): IPunCallbacks,
   Photon.PunBehaviour
- OnPhotonMaxCccuReached(): IPunCallbacks, Photon.PunBehaviour
- OnPhotonPlayerActivityChanged(): IPunCallbacks , Photon.PunBehaviour
- OnPhotonPlayerConnected(): IPunCallbacks,
   Photon.PunBehaviour
- OnPhotonPlayerDisconnected(): IPunCallbacks, Photon.PunBehaviour
- OnPhotonPlayerPropertiesChanged(): IPunCallbacks , Photon.PunBehaviour
- OnPhotonRandomJoinFailed(): IPunCallbacks, Photon.PunBehaviour

- OnPhotonSerializeView(): IPunObservable, PhotonAnimatorView, PhotonRigidbody2DView, PhotonRigidbodyView, PhotonTransformView, PhotonTransformViewPositionControl, PhotonTransformViewRotationControl, PhotonTransformViewScaleControl
- OnPrivateMessage():

ExitGames.Client.Photon.Chat.IChatClientListener

- OnReceivedRoomListUpdate(): IPunCallbacks , Photon.PunBehaviour
- onSerializeRigidBodyOption : PhotonView
- onSerializeTransformOption : PhotonView
- OnStatusUpdate():

ExitGames.Client.Photon.Chat.IChatClientListener

- OnSubscribed():
  - ExitGames.Client.Photon.Chat.IChatClientListener
- OnUnsubscribed():

ExitGames.Client.Photon.Chat.IChatClientListener

- OnUpdatedFriendList(): IPunCallbacks, Photon.PunBehaviour
- OnWebRpcResponse(): IPunCallbacks, Photon.PunBehaviour
- open : Room , RoomInfo
- openField : RoomInfo
- OperationNotAllowedInCurrentState : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- otherPlayers : **PhotonNetwork**
- OverrideBestCloudServer(): PhotonNetwork
- owner : PhotonView
- OwnerActorNr : PhotonView
- ownerld : PhotonView
- ownershipTransfer : PhotonView
- OwnerShipWasTransfered : PhotonView



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	5	M	odı	ıle	S	(	Cla	SS	es		File	es
C	Class List Clas					s In	dex		CI	lass	Hie	rarc	hy		Clas	ss M	leml	oers	•			
P	AII	II Functions					Vari	abl	es		Enι	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	perties																				
a	b	c d e f		g	h	i	j	1	m	n	0	р	q	r	s	t	u	v	w			

Here is a list of all class members with links to the classes they belong to:

### - p -

- PacketLossByCrcCheck : PhotonNetwork
- Parameters : WebRpcResponse
- ParametersOfMethods : Extensions
- ParameterType : PhotonAnimatorView
- Parse(): Region
- PeekNext(): PhotonStream
- Peer: PhotonLagSimulationGui
- PeerCount : ParameterCode
- PhotonCloud : ServerSettings
- PhotonMessageInfo(): PhotonMessageInfo
- PhotonPlayer(): PhotonPlayer
- PhotonServerSettings : PhotonNetwork
- PhotonStream(): PhotonStream
- PhotonStreamQueue(): PhotonStreamQueue
- PhotonTransformViewPositionControl():
   PhotonTransformViewPositionControl
- PhotonTransformViewRotationControl():
   PhotonTransformViewRotationControl
- PhotonTransformViewScaleControl():
   PhotonTransformViewScaleControl

- photonView : Photon.MonoBehaviour , PhotonMessageInfo
- Ping: Region
- PingSocket(): PhotonPingManager
- player : PhotonNetwork
- PlayerCount : GamePropertyKey
- playerCount : Room
- PlayerCount : Room
- playerCount : RoomInfo
- PlayerCount : RoomInfo , TypedLobbyInfo
- playerList : PhotonNetwork
- PlayerName : ActorProperties
- playerName : PhotonNetwork
- PlayerProperties : ParameterCode
- PlayerTTL : ParameterCode
- PlayerTtl : RoomOptions
- Playing : ExitGames.Client.Photon.Chat.ChatUserStatus
- PluginMismatch : **ErrorCode**
- PluginName : ParameterCode
- PluginReportedError : ErrorCode
- Plugins : ParameterCode , RoomOptions
- plugins : RoomOptions
- PluginVersion : ParameterCode
- Position : ParameterCode
- precisionForFloatSynchronization : PhotonNetwork
- precisionForQuaternionSynchronization : PhotonNetwork
- precisionForVectorSynchronization : PhotonNetwork
- PrefabCache : PhotonNetwork
- PrefabPool : PhotonNetwork
- PreferredRegion : ServerSettings
- prefix : PhotonView
- prefixBackup : PhotonView
- PrivateChannels : ExitGames.Client.Photon.Chat.ChatClient
- PrivateMessage :
  - ExitGames.Client.Photon.Chat.ChatEventCode
- Properties : ParameterCode
- PropertiesChanged : EventCode
- PropertiesListedInLobby : Room
- propertiesListedInLobby : Room
- PropsListedInLobby : GamePropertyKey
- Protocol : ServerSettings

- PublicChannels : ExitGames.Client.Photon.Chat.ChatClient
- Publish : ExitGames.Client.Photon.Chat.ChatOperationCode
- PublishMessage(): ExitGames.Client.Photon.Chat.ChatClient
- PublishUserId : ParameterCode , RoomOptions
- publishUserId : RoomOptions

• PunLogging : ServerSettings

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	age	Э	Related Pages							Modules					Cla	SS	es		File	es
C	Class	s Lis	t	C	Class	s Inc	dex		CI	ass	Hie	rarc	hy		Clas	ss M	leml	oers				
P	All	ı	=unc	ction	าร		Varia	able	:S		Enι	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
a	b	С	d	е	g	h	i	j	1	m	n	O	р	q	r	s	t	u	v	w		

Here is a list of all class members with links to the classes they belong to:

- q -

• QueueState : EventCode

• QuickResends : PhotonNetwork

Online Documentation - Dashboard - Support Forum



v1.81

ı	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	5	M	odı	ıles	S	Classes					File	es
C	Class List Clas					s In	dex		Cl	ass	Hie	rarc	hy		Clas	s M	leml	oers	5			
F	II Functions						Vari	ablo	es		Enι	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	perties																				
a	b	С	c d e f		g	h	i	j	1	m	n	O	р	q	r	s	t	u	v	w		

Here is a list of all class members with links to the classes they belong to:

#### - r -

- RaiseEvent : OperationCode , PhotonNetwork
- ReceiveNext(): PhotonStream, PhotonStreamQueue
- ReceiverGroup : ParameterCode
- Receivers : RaiseEventOptions
- Reconnect(): PhotonNetwork
- ReconnectAndRejoin(): PhotonNetwork
- RefreshCloudServerRating() : PhotonNetwork
- RefreshRpcMonoBehaviourCache(): PhotonView
- Region : ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode
- ReJoinRoom(): PhotonNetwork
- Remove : ParameterCode
- Removed : **GamePropertyKey**
- removedFromList : RoomInfo
- RemoveFriends(): ExitGames.Client.Photon.Chat.ChatClient, ExitGames.Client.Photon.Chat.ChatOperationCode
- RemovePlayerCustomProperties(): PhotonNetwork
- RemoveRPCs(): PhotonNetwork
- RemoveRPCsInGroup(): PhotonNetwork
- RequestOwnership(): PhotonView

ResentReliableCommands: PhotonNetwork

• Reset(): PhotonStreamQueue

• ResetBestRegionCodeInPreferences(): ServerSettings

• ResolveHost(): PhotonPingManager

• ReturnCode : WebRpcResponse

• Room : FriendInfo

• room : PhotonNetwork

RoomCount : TypedLobbyInfoRoomName : ParameterCode

• RotateTowards: PhotonTransformViewRotationModel

RPC(): PhotonView
RpcList: ServerSettings
RpcSecure(): PhotonView

• RunInBackground : ServerSettings

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	5	M	odı	ıle	S	(	Cla	SS	es		File	es
C	Class List Clas					s In	dex		CI	lass	Hie	rarc	hy		Clas	ss M	leml	bers	•			
P	AII	II Functions					Vari	abl	les Enumerati					ns		Enι	ıme	rato	r			
F	Prop	operties																				
a	b	b c d e f		g	h	i	j	1	m	n	0	р	q	r	s	t	u	v	w			

Here is a list of all class members with links to the classes they belong to:

### - S -

- Secret : ExitGames.Client.Photon.Chat.ChatParameterCode , ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode
- Secret1 : EncryptionDataParameters
- Secret2 : EncryptionDataParameters
- SelfHosted : ServerSettings
- SendAcksOnly(): ExitGames.Client.Photon.Chat.ChatClient
- Sender : ExitGames.Client.Photon.Chat.ChatParameterCode
- sender : PhotonMessageInfo
- Senders: ExitGames.Client.Photon.Chat.ChatChannel, ExitGames.Client.Photon.Chat.ChatParameterCode
- SendMonoMessageTargets: PhotonNetwork
- SendMonoMessageTargetType : PhotonNetwork
- SendNext(): PhotonStream, PhotonStreamQueue
- SendOutgoingCommands(): PhotonNetwork
- SendPrivate :
  - ExitGames.Client.Photon.Chat.ChatOperationCode
- SendPrivateMessage():
   ExitGames.Client.Photon.Chat.ChatClient
- sendRate : PhotonNetwork

- sendRateOnSerialize : PhotonNetwork
- SequenceChannel : RaiseEventOptions
- Serialize(): PhotonStream, PhotonStreamQueue
- SerializeView(): PhotonView
- Server : PhotonNetwork
- ServerAddress: PhotonNetwork, ServerSettings
- ServerFull : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- ServerPort : ServerSettings
- ServerSettings : OperationCode
- ServerTimestamp : PhotonNetwork
- Service(): ExitGames.Client.Photon.Chat.ChatClient
- SetAuthPostData(): AuthenticationValues ,
   ExitGames.Client.Photon.Chat.AuthenticationValues
- SetCustomProperties(): PhotonPlayer, Room
- SetLayerSynchronized(): PhotonAnimatorView
- SetLevelPrefix(): PhotonNetwork
- SetMasterClient(): PhotonNetwork
- SetOnlineStatus(): ExitGames.Client.Photon.Chat.ChatClient
- SetParameterSynchronized(): PhotonAnimatorView
- SetPlayerCustomProperties(): PhotonNetwork
- SetProperties : EventCode , OperationCode
- SetPropertiesListedInLobby(): Room
- SetReadStream(): PhotonStream
- SetReceivingEnabled(): PhotonNetwork
- SetSendingEnabled(): PhotonNetwork
- SetSynchronizedValues(): PhotonTransformView, PhotonTransformViewPositionControl
- SkipMessage :
  - ${\bf Exit Games. Client. Photon. Chat. Chat Parameter Code}$
- SlotError : **ErrorCode**
- Start(): PhotonLagSimulationGui, PhotonStatsGui
- StartPing(): PingMonoEditor
- StartRpcsAsCoroutine : PhotonNetwork
- State: ExitGames.Client.Photon.Chat.ChatClient
- statsOn: PhotonStatsGui
- statsRect : PhotonStatsGui
- statsWindowOn : PhotonStatsGui
- Status: ExitGames.Client.Photon.Chat.ChatParameterCode
- StatusUpdate: ExitGames.Client.Photon.Chat.ChatEventCode

- StopThread(): ExitGames.Client.Photon.Chat.ChatClient
- StripKeysWithNullValues(): Extensions
- StripToStringKeys(): Extensions
- Subscribe(): ExitGames.Client.Photon.Chat.ChatClient, ExitGames.Client.Photon.Chat.ChatEventCode, ExitGames.Client.Photon.Chat.ChatOperationCode
- SubscribeResults:
  - ExitGames.Client.Photon.Chat.ChatParameterCode
- SuppressRoomEvents : ParameterCode , RoomOptions
- suppressRoomEvents : RoomOptions
- SwitchToProtocol(): PhotonNetwork
- synchronization : PhotonView
- SynchronizeEnabled: PhotonTransformViewPositionModel, PhotonTransformViewRotationModel, PhotonTransformViewScaleModel
- SynchronizeType: PhotonAnimatorView.SynchronizedLayer, PhotonAnimatorView.SynchronizedParameter, PhotonAnimatorView
- SynchronizeValues : PhotonTransformViewPositionModel

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	age	9	F	Related Pages						M	odı	ıle	S	(	Cla	SS	es		File	es
C	Class List Clas					s In	dex		Cl	ass	Hie	rarc	hy		Clas	ss M	leml	bers	5			
P	AII	II Functions					Vari	ablo	es		Enι	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	erties																				
a	b	c d e f		g	h	i	j	1	m	n	0	р	q	r	s	t	u	v	w			

Here is a list of all class members with links to the classes they belong to:

### - t -

- TagObject : PhotonPlayer
- TargetActorNr : ParameterCode
- TargetActors : RaiseEventOptions
- TeleportEnabled : PhotonTransformViewPositionModel
- TeleportIfDistanceGreaterThan : PhotonTransformViewPositionModel
- time : PhotonNetwork
- timestamp : PhotonMessageInfo
- ToArray(): PhotonStream
- Token : AuthenticationValues , ExitGames.Client.Photon.Chat.AuthenticationValues
- ToString(): AuthenticationValues ,
   ExitGames.Client.Photon.Chat.AuthenticationValues ,
   FriendInfo , PhotonMessageInfo , PhotonPlayer , PhotonView ,
   Region , Room , RoomInfo , ServerSettings , TypedLobby ,
   TypedLobbyInfo
- ToStringFull(): Extensions, PhotonPlayer, Room, RoomInfo, WebRpcResponse
- ToStringMessages():
   ExitGames.Client.Photon.Chat.ChatChannel

- trafficStatsOn : PhotonStatsGui
- TrafficStatsWindow(): PhotonStatsGui
- TransferOwnership() : **PhotonView**
- Trigger : PhotonAnimatorView
- TruncateMessages(): ExitGames.Client.Photon.Chat.ChatChannel
- TryGetChannel(): ExitGames.Client.Photon.Chat.ChatClient
- Type: PhotonAnimatorView.SynchronizedParameter,
   TypedLobby
- TypedLobby(): TypedLobby

Online Documentation - Dashboard - Support Forum



v1.81

ı	Mai	n P	age	9	F	Related Pages						M	odı	ıles	S	(	Cla	SS	es		File	es
C	Class List Clas					s In	dex		Cl	ass	Hie	rarc	hy		Clas	ss M	leml	oers	•			
F	AII	II Functions					Vari	ablo	es		Enι	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	perties																				
a	b	С	c d e f		g	h	i	j	1	m	n	O	р	q	r	s	t	u	v	w		

Here is a list of all class members with links to the classes they belong to:

#### - u -

- UnAllocateViewID(): PhotonNetwork
- unreliableCommandsLimit : PhotonNetwork
- Unsubscribe(): ExitGames.Client.Photon.Chat.ChatClient, ExitGames.Client.Photon.Chat.ChatEventCode, ExitGames.Client.Photon.Chat.ChatOperationCode
- Update(): PhotonStatsGui
- UpdatePosition(): PhotonTransformViewPositionControl
- UpdateStatus:
  - ExitGames.Client.Photon.Chat.ChatOperationCode
- UriPath : ParameterCode
- UseCloud(): ServerSettings
- UseCloudBestRegion(): ServerSettings
- UseMyServer(): ServerSettings
- UseNative : PhotonPingManager
- UsePrefabCache: PhotonNetwork
- UserBlocked : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- Userld: ActorProperties, AuthenticationValues, ExitGames.Client.Photon.Chat.AuthenticationValues, ExitGames.Client.Photon.Chat.ChatClient,

ExitGames.Client.Photon.Chat.ChatParameterCode, ExitGames.Client.Photon.Chat.ParameterCode, ParameterCode

userId : PhotonPlayerUserId : PhotonPlayer

• UseRpcMonoBehaviourCache : PhotonNetwork

• Users : ExitGames.Client.Photon.Chat.ChatEventCode

Online Documentation - Dashboard - Support Forum



v1.81

ı	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	5	M	odı	ıle	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	C	Clas	s In	dex		Cl	ass	Hie	rarc	hy		Clas	s M	leml	oers	5			
F	AII		Fund	ction	าร		Vari	ablo	es		Enι	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	1	m	n	O	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

#### - V -

versionPUN : PhotonNetwork

viewID : PhotonView

• Visible : PhotonLagSimulationGui

visible : Room , RoomInfovisibleField : RoomInfo

• VoiceAppID : ServerSettings

• VoiceServerPort : ServerSettings

Online Documentation - Dashboard - Support Forum



v1.81

ı	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	•	M	odı	ıles	S	(	Cla	SSE	es		File	es
C	Class	s Lis	st	C	Clas	s In	dex		C	lass	Hie	rarc	hy		Clas	s M	lemb	ers				
F	All	-	Fund	ction	าร		Varia	ablo	es		Enu	ımeı	ratio	ns		Eηι	ıme	rato	r			
F	Prop	ertie	s																			
а	b	С	d	е	f	g	h	i	j	1	m	n	o	р	q	r	s	t	u	v	w	

Here is a list of all class members with links to the classes they belong to:

#### - W -

- WebRpc : OperationCode , PhotonNetwork
- WebRpcParameters : ParameterCode
- WebRpcResponse(): WebRpcResponse
- WebRpcReturnCode : ParameterCode
- WebRpcReturnMessage : ParameterCode
- Windowld : PhotonLagSimulationGui , PhotonStatsGui
- WindowRect : PhotonLagSimulationGui

Online Documentation - Dashboard - Support Forum



v1.81

M	laiı	n P	ag	е	F	Rel	ate	d	Pa	ges	5	М	odı	ıle	S		Cla	เรร	es	Fil	es	
CI	lass	Lis	t	(	Clas	s In	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	6			
Al	II	F	-und	ctio	ns		Vari	iab	les		Enι	ımeı	ratio	าร		En	ume	erato	r			
Pr	rope	ertie	:S																			
a	С	d	е	f	g	h	i	j	1	m	n	O	р	r	s	t	u	w				

#### - a -

- Add(): ExitGames.Client.Photon.Chat.ChatChannel
- AddAuthParameter(): AuthenticationValues, ExitGames.Client.Photon.Chat.AuthenticationValues
- AddFriends(): ExitGames.Client.Photon.Chat.ChatClient
- AllocateSceneViewID(): PhotonNetwork
- AllocateViewID(): PhotonNetwork
- AlmostEquals() : **Extensions**
- AuthenticateOnNameServer():
   ExitGames.Client.Photon.Chat.ChatPeer
- AuthenticationValues(): AuthenticationValues, ExitGames.Client.Photon.Chat.AuthenticationValues



v1.81

N	Иai	n P	ag	е	F	Rel	ate	d	Pa	ges		M	odı	ıle	S		Cla	เรร	es	Fil	es			
C	Class	s Lis	st	(	Clas	s Ind	dex		C	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	6	Files				
A	All		Fund	ctio	ns		Var	iab	les		Enι	ımeı	ratio	ns		En	ume	erato	r					
P	rop	ertie	es							,														
a	С	d	е	f	g	h	i	j	ı	m	n	o	р	r	s	t	u	w		Files				

#### - C -

- CacheDiscreteTriggers(): PhotonAnimatorView
- CacheSendMonoMessageTargets(): PhotonNetwork
- CanChatInChannel():
   ExitGames.Client.Photon.Chat.ChatClient
- ChatChannel(): ExitGames.Client.Photon.Chat.ChatChannel
- ChatClient(): ExitGames.Client.Photon.Chat.ChatClient
- ChatPeer() : ExitGames.Client.Photon.Chat.ChatPeer
- ClearExpectedUsers(): Room
- ClearMessages(): ExitGames.Client.Photon.Chat.ChatChannel
- CloseConnection(): PhotonNetwork
- CompareTo(): PhotonPlayer
- Connect(): ExitGames.Client.Photon.Chat.ChatClient, ExitGames.Client.Photon.Chat.ChatPeer
- ConnectToBestCloudServer() : **PhotonNetwork**
- ConnectToMaster() : PhotonNetwork
- ConnectToRegion(): PhotonNetwork
- ConnectUsingSettings(): PhotonNetwork
- Contains(): Extensions
- CreateRoom(): PhotonNetwork



v1.81

ľ	Иai	n P	ag	е	F	Rel	ate	ed	Pa	ges		M	odı	ıle	S		Cla	เรร	es	F	iles	
C	Clas	s Lis	st	(	Clas	s In	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	3			
F	AII		Fun	ctio	ns		Var	iab	les		Enι	ımeı	atio	ns		En	ume	erato	r			
F	Prop	ertie	es																			
a	С	d	е	f	g	h	i	j	1	m	n	О	р	r	s	t	u	w				

### - d -

- DebugReturn():
  - ExitGames.Client.Photon.Chat.IChatClientListener
- Deserialize(): PhotonStreamQueue
- DeserializeView(): PhotonView
- Destroy(): IPunPrefabPool, PhotonNetwork
- DestroyAll(): PhotonNetwork
- DestroyPlayerObjects(): PhotonNetwork
- Disconnect(): ExitGames.Client.Photon.Chat.ChatClient, PhotonNetwork
- Dispose(): PingMonoEditor
- DoesLayerSynchronizeTypeExist(): PhotonAnimatorView
- DoesParameterSynchronizeTypeExist(): PhotonAnimatorView
- Done(): PingMonoEditor
- Draw(): ExitGames.Client.GUI.GizmoTypeDrawer



v1.81

ı	Mai	n P	ag	е	ı	Rel	ate	d	Pa	ges		M	odı	ıle	S		Cla	SSE	es	File	S
C	Class	s Lis	st		Clas	s In	dex		C	lass	Hie	rarc	hy		Cla	ss N	/lem	bers			
P	AII		Fun	ctio	ns		Var	iab	les		Enu	ımeı	atio	าร		En	ume	rato	r		
F	Prop	ertie	s																		
a	С	d	е	f	g	h	i	j	1	m	n	o	р	r	s	t	u	w			

### - e -

• Equals(): PhotonPlayer, RoomInfo

EventCallback() : PhotonNetwork

Online Documentation - Dashboard - Support Forum



v1.81

Mai	n P	age	9	F	Rel	ate	d I	Paç	ges	5	M	odı	ıles	S		Cla	เรร	es	File	es
Clas	s Lis	t	(	Class	s Inc	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	6		
All	F	unc	ctio	ns		Vari	abl	les		Enu	ımeı	atio	ns		En	ume	erato	r		
Prop	ertie	s																		
a c	d	е	f	g	h	i	j	ı	m	n	O	р	r	s	t	u	w			

### - f -

- FetchServerTimestamp(): PhotonNetwork
- Find(): PhotonPlayer, PhotonView
- FindFriends(): PhotonNetwork
- FindGameObjectsWithComponent(): PhotonNetwork

Online Documentation - Dashboard - Support Forum



v1.81

N	Иai	n P	ag	е	F	Rel	ate	d	Pag	ges		M	odı	ıle	S		Cla	เรร	es		Fi	les			
C	Class	s Lis	st	(	Clas	s Ind	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	6		Files				
P	AII		Fund	ctio	ns		Vari	iab	les		Enι	ımeı	ratio	ns		En	ume	erato	r	Files					
F	Prop	ertie	es			,				,															
a	С	d	е	f	g	h	i	j	ı	m	n	O	р	r	s	t	u	w							

### - g -

- Get(): PhotonPlayer, PhotonView
- GetActive(): GameObjectExtensions
- GetCachedParemeters(): Extensions
- GetCustomRoomList(): PhotonNetwork
- GetExtrapolatedPositionOffset():
   PhotonTransformViewPositionControl
- GetHashCode(): PhotonPlayer, RoomInfo
- GetLayerSynchronizeType(): PhotonAnimatorView
- GetNetworkPosition(): PhotonTransformViewPositionControl
- GetNetworkRotation(): PhotonTransformViewRotationControl
- GetNetworkScale(): PhotonTransformViewScaleControl
- GetNext(): PhotonPlayer
- GetNextFor(): PhotonPlayer
- GetParameterSynchronizeType(): PhotonAnimatorView
- GetPhotonView(): Extensions
- GetPhotonViewsInChildren(): Extensions
- GetPing(): PhotonNetwork
- GetPrivateChannelNameByUser() : ExitGames.Client.Photon.Chat.ChatClient
- GetRoomList(): PhotonNetwork
- GetRotation(): PhotonTransformViewRotationControl

- GetScale(): PhotonTransformViewScaleControl
- GetSynchronizedLayers(): PhotonAnimatorView
   GetSynchronizedParameters(): PhotonAnimatorView

Online Documentation - Dashboard - Support Forum



v1.81

Mai	n Pa	age	<b>,</b>	F	Rel	ate	d	Pa	ges		М	odı	ıle	S		Cla	เรร	es	Fil	es	
Clas	s List	t	(	Class	s In	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	6	Files		
All	F	unc	tio	ns		Var	iab	les		Enι	ımeı	atio	าร		En	ume	erato	r			
Prop	ertie	S																			
a c	d	е	f	g	h	i	j	ı	m	n	O	р	r	s	t	u	w				

### - h -

• HasQueuedObjects(): PhotonStreamQueue

• HelpURL(): HelpURL

Online Documentation - Dashboard - Support Forum



v1.81

N	Иai	n P	ag	е	F	Rela	ate	d	Paç	ges		M	odı	ıles	S		Cla	ISS	es	File	S			
C	Class	s Lis	st	(	Class	s Inc	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	3					
P	All	ı	Fund	ctio	ns		Vari	iab	les		Enu	ımeı	ratio	ns		En	ume	erato	r	Files				
F	Prop	ertie	s																					
a	С	d	е	f	g	h	i	j	ı	m	n	o	р	r	s	t	u	w						

- i -

• InitializeSecurity(): PhotonNetwork

• Instantiate(): IPunPrefabPool, PhotonNetwork

• InstantiateSceneObject(): PhotonNetwork

• IsAppld(): ServerSettings

Online Documentation - Dashboard - Support Forum



v1.81

I	Иai	n P	ag	е	F	Rel	ate	ed	Pa	ges		M	odı	ıle	S		Cla	SSE	es	F	iles	,
C	Clas	s Lis	st	(	Clas	s In	dex		C	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	3			
F	AII		Fun	ctio	ns		Var	iab	les		Enu	ımeı	atio	ns		En	ume	erato	r			
F	Prop	ertie	es							,												
a	С	d	е	f	g	h	i	j	1	m	n	o	р	r	s	t	u	w				

- j -

• JoinLobby(): PhotonNetwork

• JoinOrCreateRoom(): PhotonNetwork

• JoinRandomRoom() : PhotonNetwork

• JoinRoom(): PhotonNetwork

Online Documentation - Dashboard - Support Forum



v1.81

N	Иai	n P	ag	е	F	Rel	ate	d	Pa	ges	,	M	odı	ıle	S		Cla	เรร	es	Fil	es	
C	Class	s Lis	st	(	Clas	s Ind	dex		C	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	6			
P	All		Fund	ctio	ns		Vari	iab	les		Enι	ımeı	ratio	ns		En	ume	erato	r			
F	Prop	ertie	es			,				,												
a	С	d	е	f	g	h	i	j	ı	m	n	o	р	r	s	t	u	w				

- 1 -

LeaveLobby(): PhotonNetwork
 LeaveRoom(): PhotonNetwork
 LoadLevel(): PhotonNetwork

• LoadScene(): UnityEngine.SceneManagement.SceneManager

Online Documentation - Dashboard - Support Forum



v1.81

Ma	ain I	Pag	е	F	Rela	ate	d	Paç	ges		М	odı	ıles	5		Cla	ISS	es	File	s
Cla	ass Li	st	(	Class	Inc	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	6		
All		Fund	ctio	ns	,	Vari	iab	les		Enι	ımeı	atio	ns		En	ume	erato	or		
Pro	operti	es																		
a (	c d	е	f	g	h	i	j	ı	m	n	O	р	r	s	t	u	w			

### - m -

• Merge(): Extensions

• MergeStringKeys(): Extensions

Online Documentation - Dashboard - Support Forum



v1.81

Mai	n Pa	age	<b>;</b>	F	Rel	ate	d	Pa	ges		M	odı	ıle	S		Cla	เรร	es	Fil	es
Class	s List	t	(	Clas	s Inc	dex		C	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	6		
All	F	unc	tio	ns		Vari	iab	les		Enι	ıme	ratio	ns		En	ume	erato	r		
Prop	ertie	S																		
ас	d	е	f	g	h	i	j	ı	m	n	o	р	r	s	t	u	w			

### - n -

- NetworkStatisticsReset(): PhotonNetwork
   NetworkStatisticsToString(): PhotonNetwork

Online Documentation - Dashboard - Support Forum



v1.81

N	Иai	n P	ag	е	ı	Rel	ate	ed	Pa	ges		M	odı	ıle	S		Cla	SSE	es	File	s
C	Class	s Lis	st	(	Clas	s In	dex		C	lass	Hie	rarc	hy		Cla	ss N	/lem	bers			
P	AII		Fun	ctio	ns		Var	iab	les		Enu	ımeı	atio	าร		En	ume	rato	r		
F	Prop	ertie	es																		
a	С	d	е	f	g	h	i	j	1	m	n	o	р	r	s	t	u	w			

#### - 0 -

- OnChatStateChange(): ExitGames.Client.Photon.Chat.IChatClientListener
- OnConnected():

ExitGames.Client.Photon.Chat.IChatClientListener

- OnConnectedToMaster(): IPunCallbacks , Photon.PunBehaviour
- OnConnectedToPhoton(): IPunCallbacks, Photon.PunBehaviour
- OnConnectionFail(): IPunCallbacks, Photon.PunBehaviour
- OnCreatedRoom(): IPunCallbacks, Photon.PunBehaviour
- OnCustomAuthenticationFailed(): IPunCallbacks , Photon.PunBehaviour
- OnCustomAuthenticationResponse(): IPunCallbacks, Photon.PunBehaviour
- OnDisconnected():

ExitGames.Client.Photon.Chat.IChatClientListener

- OnDisconnectedFromPhoton(): IPunCallbacks , Photon.PunBehaviour
- OnFailedToConnectToPhoton(): IPunCallbacks , Photon.PunBehaviour
- OnGetMessages():

- ExitGames.Client.Photon.Chat.IChatClientListener
- OnGUI(): PhotonLagSimulationGui, PhotonStatsGui
- OnJoinedLobby(): IPunCallbacks, Photon.PunBehaviour
- OnJoinedRoom(): IPunCallbacks, Photon.PunBehaviour
- OnLeftLobby(): IPunCallbacks, Photon.PunBehaviour
- OnLeftRoom(): IPunCallbacks, Photon.PunBehaviour
- OnLobbyStatisticsUpdate(): IPunCallbacks , Photon.PunBehaviour
- OnMasterClientSwitched(): IPunCallbacks ,
   Photon.PunBehaviour , PhotonView
- OnOwnershipRequest(): IPunCallbacks, Photon.PunBehaviour
- OnOwnershipTransfered(): IPunCallbacks, Photon.PunBehaviour
- OnPhotonCreateRoomFailed(): IPunCallbacks , Photon.PunBehaviour
- OnPhotonCustomRoomPropertiesChanged(): IPunCallbacks , Photon.PunBehaviour
- OnPhotonInstantiate(): IPunCallbacks, Photon.PunBehaviour
- OnPhotonJoinRoomFailed(): IPunCallbacks,
   Photon.PunBehaviour
- OnPhotonMaxCccuReached(): IPunCallbacks, Photon.PunBehaviour
- OnPhotonPlayerActivityChanged(): IPunCallbacks , Photon.PunBehaviour
- OnPhotonPlayerConnected(): IPunCallbacks , Photon.PunBehaviour
- OnPhotonPlayerDisconnected(): IPunCallbacks, Photon.PunBehaviour
- OnPhotonPlayerPropertiesChanged(): IPunCallbacks, Photon.PunBehaviour
- OnPhotonRandomJoinFailed(): IPunCallbacks,
   Photon.PunBehaviour
- OnPhotonSerializeView(): IPunObservable, PhotonAnimatorView, PhotonRigidbody2DView, PhotonRigidbodyView, PhotonTransformView, PhotonTransformViewPositionControl, PhotonTransformViewRotationControl, PhotonTransformViewScaleControl
- OnPrivateMessage():
   ExitGames.Client.Photon.Chat.IChatClientListener

- OnReceivedRoomListUpdate(): IPunCallbacks , Photon.PunBehaviour
- OnStatusUpdate() : ExitGames.Client.Photon.Chat.IChatClientListener
- OnSubscribed(): ExitGames.Client.Photon.Chat.IChatClientListener
- OnUnsubscribed(): ExitGames.Client.Photon.Chat.IChatClientListener
- OnUpdatedFriendList(): IPunCallbacks, Photon.PunBehaviour
- OnWebRpcResponse(): IPunCallbacks, Photon.PunBehaviour
- OverrideBestCloudServer(): PhotonNetwork

Online Documentation - Dashboard - Support Forum



v1.81

ı	Иai	n P	ag	е	F	Rel	ate	ed	Pa	ges		M	odı	ıle	S		Cla	เรร	es	F	iles	
C	Clas	s Lis	st	(	Clas	s In	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	3			
F	AII		Fund	ctio	ns		Var	iab	les		Enu	ımeı	atio	าร		En	ume	erato	r			
F	Prop	ertie	es																			
a	С	d	е	f	g	h	i	j	1	m	n	o	р	r	s	t	u	w				

### - p -

- Parse(): Region
- PeekNext(): PhotonStream
- PhotonMessageInfo(): PhotonMessageInfo
- PhotonPlayer(): PhotonPlayer
- PhotonStream(): PhotonStream
- PhotonStreamQueue(): PhotonStreamQueue
- PhotonTransformViewPositionControl():
   PhotonTransformViewPositionControl
- PhotonTransformViewRotationControl():
   PhotonTransformViewRotationControl
- PhotonTransformViewScaleControl():
   PhotonTransformViewScaleControl
- PingSocket(): PhotonPingManager
- PublishMessage(): ExitGames.Client.Photon.Chat.ChatClient



v1.81

I	Иai	n P	ag	е	ı	Rel	ate	ed	Pa	ges		M	odı	ıle	S		Cla	เรร	es	F	iles	
C	Clas	s Lis	st	(	Clas	s In	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	3			
F	AII		Fun	ctio	ns		Var	iabl	les		Enι	ımeı	atio	ns		En	ume	erato	r			
F	Prop	ertie	es																			
a	С	d	е	f	g	h	i	j	1	m	n	o	р	r	s	t	u	w				

#### - r -

- RaiseEvent(): PhotonNetwork
- ReceiveNext(): PhotonStream, PhotonStreamQueue
- Reconnect(): PhotonNetwork
- ReconnectAndRejoin(): PhotonNetwork
- RefreshCloudServerRating(): PhotonNetwork
- RefreshRpcMonoBehaviourCache(): PhotonView
- ReJoinRoom(): PhotonNetwork
- RemoveFriends(): ExitGames.Client.Photon.Chat.ChatClient
- RemovePlayerCustomProperties(): PhotonNetwork
- RemoveRPCs(): PhotonNetwork
- RemoveRPCsInGroup(): PhotonNetwork
- RequestOwnership(): PhotonView
- Reset(): PhotonStreamQueue
- ResetBestRegionCodeInPreferences(): ServerSettings
- ResolveHost(): PhotonPingManager
- RPC(): PhotonView
- RpcSecure(): PhotonView



v1.81

ľ	Иai	n P	ag	е	F	Rel	ate	ed	Pa	ges		M	odı	ıle	S		Cla	SSE	es	F	-iles	5
C	Clas	s Lis	st	(	Clas	s In	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	5			
F	AII		Fund	ctio	ns		Var	iab	les		Enu	ımeı	atio	ns		En	ume	erato	r			
F	Prop	ertie	es																			
a	С	d	е	f	g	h	i	j	ı	m	n	o	р	r	s	t	u	w				

#### - S -

- SendAcksOnly(): ExitGames.Client.Photon.Chat.ChatClient
- SendNext(): PhotonStream, PhotonStreamQueue
- SendOutgoingCommands(): PhotonNetwork
- SendPrivateMessage() : ExitGames.Client.Photon.Chat.ChatClient
- Serialize(): PhotonStream, PhotonStreamQueue
- SerializeView(): PhotonView
- Service(): ExitGames.Client.Photon.Chat.ChatClient
- SetAuthPostData(): AuthenticationValues, ExitGames.Client.Photon.Chat.AuthenticationValues
- SetCustomProperties(): PhotonPlayer, Room
- SetLayerSynchronized(): PhotonAnimatorView
- SetLevelPrefix(): PhotonNetwork
- SetMasterClient(): PhotonNetwork
- SetOnlineStatus(): ExitGames.Client.Photon.Chat.ChatClient
- SetParameterSynchronized(): PhotonAnimatorView
- SetPlayerCustomProperties(): PhotonNetwork
- SetPropertiesListedInLobby(): Room
- SetReadStream(): PhotonStream
- SetReceivingEnabled(): PhotonNetwork
- SetSendingEnabled(): PhotonNetwork

- SetSynchronizedValues(): PhotonTransformView,
   PhotonTransformViewPositionControl
- Start(): PhotonLagSimulationGui, PhotonStatsGui
- StartPing(): PingMonoEditor
- StopThread(): ExitGames.Client.Photon.Chat.ChatClient
- StripKeysWithNullValues(): Extensions
- StripToStringKeys(): Extensions
- Subscribe(): ExitGames.Client.Photon.Chat.ChatClient
- SwitchToProtocol(): PhotonNetwork

Online Documentation - Dashboard - Support Forum



v1.81

I	Иai	n P	ag	е	F	Rel	ate	ed	Pa	ges		M	odı	ıle	S		Cla	SSE	es	F	iles	
C	Clas	s Lis	st	(	Clas	s In	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers				
F	AII		Fund	ctio	ns		Var	iab	les		Enu	ımeı	atio	ns		En	ume	erato	r			
F	Prop	ertie	es																			
a	С	d	е	f	g	h	i	j	ı	m	n	o	р	r	s	t	u	w				

### - t -

- ToArray(): PhotonStream
- ToString(): AuthenticationValues,
   ExitGames.Client.Photon.Chat.AuthenticationValues,
   FriendInfo, PhotonMessageInfo, PhotonPlayer, PhotonView,
   Region, Room, RoomInfo, ServerSettings, TypedLobby,
   TypedLobbyInfo
- ToStringFull(): Extensions, PhotonPlayer, Room, RoomInfo, WebRpcResponse
- ToStringMessages():
  - ExitGames.Client.Photon.Chat.ChatChannel
- TrafficStatsWindow(): PhotonStatsGui
- TransferOwnership(): PhotonView
- TruncateMessages():
  - ExitGames.Client.Photon.Chat.ChatChannel
- TryGetChannel(): ExitGames.Client.Photon.Chat.ChatClient
- TypedLobby(): TypedLobby



v1.81

ı	Иai	n P	ag	е	F	Rel	ate	ed	Pa	ges		M	odı	ıle	S		Cla	เรร	es	F	iles	
C	Clas	s Lis	st	(	Clas	s In	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	3			-
F	AII		Fund	ctio	ns		Var	iab	les		Enu	ımeı	atio	าร		En	ume	erato	r			
F	Prop	ertie	es																			
a	С	d	е	f	g	h	i	j	1	m	n	o	р	r	s	t	u	w				

### - u -

- UnAllocateViewID(): PhotonNetwork
- Unsubscribe(): ExitGames.Client.Photon.Chat.ChatClient
- Update(): PhotonStatsGui
- UpdatePosition(): PhotonTransformViewPositionControl
- UseCloud(): ServerSettings
- UseCloudBestRegion(): ServerSettings
- UseMyServer() : ServerSettings

Online Documentation - Dashboard - Support Forum



v1.81

Mai	n P	age	Э	F	Rel	ate	d	Pa	ges		М	odı	ıle	S		Cla	เรร	es	File	es
Clas	s Lis	t	(	Clas	s In	dex		С	lass	Hie	rarc	hy		Cla	ss N	/lem	bers	5		
All	F	unc	ctio	ns		Vari	iab	les		Enι	ımeı	atio	ns		En	ume	erato	r		
Prop	ertie	S																		
a c	d	е	f	g	h	i	j	1	m	n	O	р	r	s	t	u	w			

- W -

WebRpc(): PhotonNetworkWebRpcResponse(): WebRpcResponse

Online Documentation - Dashboard - Support Forum



v1.81

ľ	Mai	n P	ag	е	F	Rel	ate	d F	Paç	ges	5	M	odı	ıles	5	(	Cla	SSE	es		File	es
C	Class	s Lis	st	C	Clas	s In	dex		С	lass	Hie	rarc	hy		Clas	s M	emk	ers				
A	AII		Fund	ction	าร		Vari	abl	es		Enu	ımeı	atio	าร		Enu	ımeı	rato	r			
F	Prop	ertie	es																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	s	t	u	v	w	

#### - a -

- ActorList : ParameterCode
- ActorNr : ParameterCode
- Add : ParameterCode
- AddFriends :
  - ${\bf Exit Games. Client. Photon. Chat. Chat Operation Code}$
- Address: ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode
- AlreadyMatched : ErrorCode
- ApplD : ServerSettings
- ApplicationId : ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode
- AppStats : **EventCode**
- AppVersion : ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode
- Attempts : PhotonPingManager
- Authenticate :
  - ExitGames.Client.Photon.Chat.ChatOperationCode , OperationCode
- AuthenticateOnce : OperationCode
- AuthenticationTicketExpired : ErrorCode
- AuthEvent : EventCode

• autoCleanUpField : RoomInfo

• Away : ExitGames.Client.Photon.Chat.ChatUserStatus

AzureLocalNodeld : ParameterCode
 AzureMasterNodeld : ParameterCode

• AzureNodeInfo : EventCode , ParameterCode

Online Documentation - Dashboard - Support Forum



v1.81

N	Иai	n P	age	е	F	Rel	ate	d F	Paç	jes	6	M	odu	ıles	5	(	Cla	SSE	es		File	es
C	Class	s Lis	st	C	Clas	s Ind	dex		С	lass	s Hie	rarc	hy		Clas	s M	lemb	ers	•			
P	All		Fund	ction	าร		Vari	ablo	es		En	ıme	ratio	ns		Enι	ıme	rato	r			
F	Prop	ertie	es																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	0	р	q	r	s	t	u	v	w	

### - b -

• BackgroundTimeout : **PhotonNetwork** 

Broadcast : ParameterCodebuttonsOn : PhotonStatsGui

Online Documentation - Dashboard - Support Forum



v1.81

ľ	Mai	n P	ago	е	F	Rel	ate	d F	Paç	jes	6	M	odu	ıles	5	(	Cla	SSE	es		File	es
C	Class	s Lis	st	C	Clas	s Ind	dex		С	lass	Hie	rarc	hy		Clas	s M	emb	ers	•			
F	AII		Fund	ction	าร		Vari	ablo	es		Enu	ıme	ratio	ns		Enu	ıme	rato	r			
F	Prop	ertie	es							,												
a	b	С	d	е	f	g	h	i	j	ı	m	n	0	р	q	r	s	t	u	v	w	

#### - C -

• Cache: ParameterCode

• CacheSliceChanged : **EventCode** 

• CacheSliceIndex : ParameterCode

• CachingOption : RaiseEventOptions

• ChangeGroups : OperationCode

• Channel : ExitGames.Client.Photon.Chat.ChatParameterCode

• ChannelHistory:

ExitGames.Client.Photon.Chat.ChatOperationCode

- Channels: ExitGames.Client.Photon.Chat.ChatParameterCode
- ChannelUserCount:

ExitGames.Client.Photon.Chat.ChatParameterCode

- ChatAppID : ServerSettings
- ChatMessages :

ExitGames.Client.Photon.Chat.ChatEventCode

- chatPeer : ExitGames.Client.Photon.Chat.ChatClient
- CheckUserOnJoin: ParameterCode
- CleanupCacheOnLeave : GamePropertyKey , ParameterCode
- ClientAuthenticationData :

ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode

• ClientAuthenticationParams :

ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode

 ClientAuthenticationType : ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode

• Code : ParameterCode , Region

• CreateGame : OperationCode

• currentMasterID : PhotonView

• CustomAuthenticationFailed : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode

• CustomEventContent : ParameterCode

• CustomInitData : ParameterCode

• CustomRoomProperties : RoomOptions

• CustomRoomPropertiesForLobby : RoomOptions

Online Documentation - Dashboard - Support Forum



v1.81

ľ	Mai	n P	Э	F	Rel	ate	d F	Paç	ges	5	Modules					Cla	SS	es		File	es	
C	Class	C	Class	s In	dex		С	lass	Hie	rarc	hy		Class Members									
A	AII		Fund	ction	าร		Variables					Enumerations					Enumerator					
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	s	t	u	v	w	

### - d -

• Data: ParameterCode

• Default : RaiseEventOptions , TypedLobby

• DisableAutoOpenWizard : ServerSettings

• DND : ExitGames.Client.Photon.Chat.ChatUserStatus

• DrawErrorGizmo: PhotonTransformViewPositionModel

Online Documentation - Dashboard - Support Forum



v1.81

ľ	Mai	n P	age	Э	F	Rel	ate	d F	Paç	ges	5	Modules					Cla	SS	es		es	
C	Class	C	Clas	s In	dex		Class Hierarchy						Class Members									
F	AII		Fund	ction	าร	Variables					Enumerations					Enumerator						
F	Prop	ertie	es																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	0	р	q	r	s	t	u	v	w	

#### - e -

- EmptyRoomTTL : ParameterCode
- EmptyRoomTtl : RoomOptions
- EnabledRegions : ServerSettings
- EnableLobbyStatistics : ServerSettings
- Encrypt : RaiseEventOptions
- EncryptionData : ParameterCode
- EncryptionMode : ParameterCode
- ErrorInfo : EventCode
- EventForward : ParameterCode
- ExchangeKeysForEncryption : OperationCode
- ExpectedProtocol : ParameterCode
- ExpectedUsers : GamePropertyKey
- expectedUsersField: RoomInfo
- ExpectedValues : ParameterCode
- ExternalHttpCallFailed : ErrorCode
- ExtrapolateIncludingRoundTripTime : PhotonTransformViewPositionModel
- ExtrapolateNumberOfStoredPositions:
   PhotonTransformViewPositionModel
- ExtrapolateOption : PhotonTransformViewPositionModel
- ExtrapolateSpeed : PhotonTransformViewPositionModel

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	5	M	odı	ıles	6	(	Cla	SSE	es		es	
C	Class	C	lass	s Inc	dex		Class Hierarchy						Class Members									
P	AII	ı	=unc	ction	ıs		Variables					Enumerations					ımeı	rato				
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	s	t	u	v	w	

### - f -

- FindFriends : OperationCode
- FindFriendsRequestList : ParameterCode
- FindFriendsResponseOnlineList : ParameterCode
- FindFriendsResponseRoomIdList : ParameterCode
- ForwardToWebhook : RaiseEventOptions
- Friends : ExitGames.Client.Photon.Chat.ChatParameterCode
- FriendsList: ExitGames.Client.Photon.Chat.ChatEventCode

Online Documentation - Dashboard - Support Forum



v1.81

ľ	Mai	n P	Э	F	Rel	ate	d F	Paç	jes	5	Modules					Cla	SS	es		File	es	
C	Class	C	Clas	s In	dex		Class Hierarchy						Class Members									
F	AII		Fund	ction	าร	Variables					Enumerations					Enumerator						
F	Prop	ertie	es							,												
a	b	С	d	е	f	g	h	i	j	1	m	n	0	р	q	r	s	t	u	v	w	

### - g -

- GameClosed : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- GameCount : ParameterCode
- GameDoesNotExist: ErrorCode, ExitGames.Client.Photon.Chat.ErrorCode
- GameFull : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- GameIdAlreadyExists : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- GameList : EventCode , ParameterCode
- GameListUpdate : EventCode
- GameProperties : ParameterCode
- GetGameList : OperationCode
- GetLobbyStats : OperationCode
- GetProperties : OperationCode
- GetRegions : OperationCode
- Group : ParameterCode
- group : PhotonView



v1.81

N	Mai	n P	age	е	F	Rel	ate	d F	Paç	jes	•	M	odı	ıles	5	(	Cla	SSE	es		File	es
C	Class	s Lis	List Class Index						C	ass	Hie	rarc	hy		Clas	s M	emk	ers	•			
P	AII		Functions Variate					able	es		Enu	ıme	ratio	ns		Enu	ımeı	rato	r			
F	Prop	ertie	es																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	s	t	u	v	w	

### - h -

• healthStatsVisible : PhotonStatsGui

• HistoryLength:

ExitGames.Client.Photon.Chat.ChatParameterCode

• HostAndPort : Region

• HostType : ServerSettings

• HttpLimitReached : ErrorCode

Online Documentation - Dashboard - Support Forum



v1.81

ı	Mai	n P	age	9	F	Rel	ate	d F	Paç	ges	6	M	odu	ıles	S	(	Cla	SS	es		File	es
C	Class	s Lis	st	Class Index					С	lass	s Hie	rarc	hy		Clas	s M	lemb	oers	3			
F	AII	ı	Func	Class Index ctions Variab					es		En	ıme	ratio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
а	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	s	t	u	v	w	

### - i -

- IgnoreInitialAttempt : PhotonPingManager
- Info : ParameterCode
- InstantiateInRoomOnly : PhotonNetwork
- instantiationId : PhotonView
- InterestGroup : RaiseEventOptions
- InternalServerError: ErrorCode,
   ExitGames.Client.Photon.Chat.ErrorCode
- InterpolateLerpSpeed: PhotonTransformViewPositionModel, PhotonTransformViewRotationModel, PhotonTransformViewScaleModel
- InterpolateMoveTowardsAcceleration : PhotonTransformViewPositionModel
- InterpolateMoveTowardsDeceleration : PhotonTransformViewPositionModel
- InterpolateMoveTowardsSpeed:
   PhotonTransformViewPositionModel,
   PhotonTransformViewScaleModel
- InterpolateOption: PhotonTransformViewPositionModel, PhotonTransformViewRotationModel, PhotonTransformViewScaleModel
- InterpolateRotateTowardsSpeed:

### **PhotonTransformViewRotationModel**

- InterpolateSpeedCurve : PhotonTransformViewPositionModel
- InvalidAuthentication : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- InvalidEncryptionParameters : **ErrorCode**
- InvalidOperation : ErrorCode
- InvalidOperationCode : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- InvalidRegion : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- Invisible : ExitGames.Client.Photon.Chat.ChatUserStatus
- IsComingBack : ParameterCode
- IsInactive : ActorProperties , ParameterCode
- IsLocal : PhotonPlayer
- IsOpen : GamePropertyKeyIsVisible : GamePropertyKey

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	6	M	odı	ıles	6	(	Cla	SSE	es		File	es
C	Class	Lis	t	C	Class Index					lass	s Hie	rarc	hy		Clas	s M	eml	ers	3			
P	All	ı	=unc	tior	Class Index tions Variab						En	ıme	ratio	ns		Enu	ımeı	rato	r			
F	rop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	s	t	u	v	w	

### - j -

- Join : EventCode , OperationCode
- JoinFailedFoundActiveJoiner : ErrorCode
- JoinFailedFoundExcludedUserId : ErrorCode
- JoinFailedFoundInactiveJoiner : ErrorCode
- JoinFailedPeerAlreadyJoined : **ErrorCode**
- JoinFailedWithRejoinerNotFound : ErrorCode
- JoinGame : OperationCode
- JoinLobby : OperationCode , ServerSettings
- JoinMode : ParameterCode
- JoinRandomGame : OperationCode

Online Documentation - Dashboard - Support Forum



v1.81

I	Mai	n P	age	е	F	Rel	ate	d F	Paç	jes	5	M	odı	ıles	S	(	Cla	SS	es		File	es
C	Class	s Lis							C	lass	Hie	rarc	hy		Clas	s M	lemb	ers	•			
F	AII		Class Index Functions Varia					abl	es		Enu	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s							,												
a	b	С	d	е	f	g	h	i	j	1	m	n	o	р	q	r	s	t	u	v	w	

### - | -

- LayerIndex : PhotonAnimatorView.SynchronizedLayer
- Leave : EventCode , OperationCode
- LeaveLobby : OperationCode
- LFG: ExitGames.Client.Photon.Chat.ChatUserStatus
- LobbyName : ParameterCode
- LobbyStats : **EventCode** , **ParameterCode**
- LobbyType : ParameterCodelogLevel : PhotonNetwork

Online Documentation - Dashboard - Support Forum



v1.81

ı	Mai	n P	age	е	ı	Rel	ate	d F	Paç	jes	5	M	odı	ıles	5	(	Cla	SSE	es		File	es
C	Class	s Lis							C	lass	Hie	rarc	hy		Clas	s M	emk	ers				
P	AII		Fund	Class Index inctions Varial					es		Enι	ımeı	atio	าร		Enu	ımeı	rato	r			
F	Prop	ertie	es							,												
a	b	С	d	е	f	g	h	i	j	1	m	n	o	р	q	r	s	t	u	v	w	

#### - m -

- MasterClientId : GamePropertyKey , ParameterCode
- MasterPeerCount : ParameterCode
- Match : EventCode
- MatchMakingType : ParameterCode
- MAX\_VIEW\_IDS : PhotonNetwork
- MaxCcuReached : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- maxConnections : PhotonNetwork
- MaxMilliseconsPerPing : PhotonPingManager
- MaxPlayers : GamePropertyKey , RoomOptions
- maxPlayersField : RoomInfo
- Message : ExitGames.Client.Photon.Chat.ChatParameterCode
- MessageLimit: ExitGames.Client.Photon.Chat.ChatChannel, ExitGames.Client.Photon.Chat.ChatClient
- Messages: ExitGames.Client.Photon.Chat.ChatChannel, ExitGames.Client.Photon.Chat.ChatParameterCode
- Mode : EncryptionDataParameters
- Msgld : ExitGames.Client.Photon.Chat.ChatParameterCode
- Msglds: ExitGames.Client.Photon.Chat.ChatParameterCode



v1.81

ı	Mai	n P	age	9	F	Rel	ate	d F	Paç	ges	6	M	odı	ıles	5	(	Cla	SSE	es		File	es
C	Class	s Lis	st	Class Index					С	lass	s Hie	rarc	hy		Clas	s M	emb	ers	3			
F	All	ı	Func	ction	Class Index tions Variab				es		Enu	ıme	ratio	ns		Enι	ımeı	rato	r			
F	Prop	ertie	s																			
а	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	s	t	u	v	w	

#### - n -

- Name: ExitGames.Client.Photon.Chat.ChatChannel, PhotonAnimatorView.SynchronizedParameter, TypedLobby
- nameField : RoomInfo
- NameServerHost: ExitGames.Client.Photon.Chat.ChatPeer
- NameServerHttp : ExitGames.Client.Photon.Chat.ChatPeer
- NetworkLogging : ServerSettings
- NickName : ParameterCode
- NoRandomMatchFound : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode

Online Documentation - Dashboard - Support Forum



v1.81

N	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	6	M	odı	ıles	6	(	Cla	SSE	es		File	es
C	Class	Lis	t	Class Index					C	lass	s Hie	rarc	hy		Clas	s M	eml	ers	3			
P	All	ı	=unc	tior	Class Index tions Variab				es		En	ıme	ratio	ns		Enu	ımeı	rato	r			
F	rop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	s	t	u	v	w	

#### **- 0 -**

- ObservedComponents : PhotonView
- Offline : ExitGames.Client.Photon.Chat.ChatUserStatus
- Ok : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- OnEventCall : PhotonNetwork
- Online: ExitGames.Client.Photon.Chat.ChatUserStatus
- onSerializeRigidBodyOption : PhotonView
- onSerializeTransformOption : PhotonView
- openField : RoomInfo
- OperationNotAllowedInCurrentState : **ErrorCode** , **ExitGames.Client.Photon.Chat.ErrorCode**
- ownerld : PhotonView
- ownershipTransfer : **PhotonView**
- OwnerShipWasTransfered : PhotonView



v1.81

ľ	Mai	n P	age	Э	F	Rel	ate	d F	Paç	ges	5	M	odı	ıles	S	(	Cla	SS	es		File	es
C	Class	s Lis	List Class Index						C	lass	Hie	rarc	hy		Clas	s M	lemb	ers	•			
F	AII		Class Index Functions Varia					abl	es		Enu	ımeı	atio	ns		Enι	ıme	rato	r			
F	Prop	ertie	es																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	s	t	u	v	w	

### - p -

- ParametersOfMethods : Extensions
- PeerCount : ParameterCode
- PhotonServerSettings : PhotonNetwork
- photonView : PhotonMessageInfo
- Ping : Region
- PlayerCount : **GamePropertyKey** , **TypedLobbyInfo**
- PlayerName : ActorProperties
- PlayerProperties : ParameterCode
- PlayerTTL : ParameterCode
- PlayerTtl : RoomOptions
- Playing : ExitGames.Client.Photon.Chat.ChatUserStatus
- PluginMismatch : ErrorCode
- PluginName : ParameterCode
- PluginReportedError : ErrorCode
- Plugins : ParameterCode , RoomOptions
- PluginVersion : ParameterCode
- Position : ParameterCode
- precisionForFloatSynchronization : **PhotonNetwork**
- precisionForQuaternionSynchronization : PhotonNetwork
- precisionForVectorSynchronization : PhotonNetwork
- PrefabCache : PhotonNetwork

• PreferredRegion : ServerSettings

• prefixBackup : PhotonView

• PrivateChannels : ExitGames.Client.Photon.Chat.ChatClient

• PrivateMessage:

ExitGames.Client.Photon.Chat.ChatEventCode

• Properties : ParameterCode

• PropertiesChanged : EventCode

• PropsListedInLobby : GamePropertyKey

• Protocol : ServerSettings

• PublicChannels : ExitGames.Client.Photon.Chat.ChatClient

• Publish : ExitGames.Client.Photon.Chat.ChatOperationCode

• PublishUserId : ParameterCode

• PunLogging : ServerSettings

Online Documentation - Dashboard - Support Forum



v1.81

ı	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	5	M	odı	ıles	S	(	Cla	SS	es		File	es
C	Class	s Lis	st						C	lass	Hie	rarc	hy		Clas	s M	leml	oers	•			
F	All	1	Func	Class Index actions Varial					es		Enu	ımeı	ratio	ns		Enι	ıme	rato	r			
F	Prop	ertie	s																			
а	b	С	d	е	f	g	h	i	j	1	m	n	o	р	q	r	s	t	u	v	w	

- q -

• QueueState : EventCode

Online Documentation - Dashboard - Support Forum



v1.81

I	Mai	n P	age	Э	F	Rel	ate	d F	Pag	jes	5	M	odı	ıles	S	(	Cla	SS	es		File	es
C	Class	s Lis	List Class Index						С	lass	Hie	rarc	hy		Clas	s M	lemk	ers	•			
F	AII		Fund	t Class Index Functions Varia					es		Enu	ıme	ratio	ns		Enι	ımeı	rato	r			
F	Prop	ertie	es																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	0	р	q	r	s	t	u	v	w	

#### - r -

- RaiseEvent : OperationCode
- ReceiverGroup : ParameterCode
- Receivers : RaiseEventOptions
- Region : ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode
- Remove : ParameterCode
- Removed : GamePropertyKey
- RemoveFriends :
  - ExitGames.Client.Photon.Chat.ChatOperationCode
- RoomCount : TypedLobbyInfo
- RoomName : ParameterCode
- RpcList : ServerSettings
- RunInBackground : ServerSettings



v1.81

I	Mai	n P	age	9	F	Rel	ate	d F	Paç	jes	6	M	odı	ıles	5	(	Cla	SS	es		File	es
C	Class	s Lis	st	C	Class Index					lass	s Hie	rarc	hy		Clas	s M	eml	oers	•			
F	AII	ı	Func	ction	Class Index tions Variab				es		Enu	ıme	ratio	ns		Eηι	ıme	rato	r			
F	Prop	ertie	s																			
а	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	s	t	u	v	w	

#### - S -

- Secret : ExitGames.Client.Photon.Chat.ChatParameterCode , ExitGames.Client.Photon.Chat.ParameterCode , ParameterCode
- Secret1: EncryptionDataParameters
- Secret2 : EncryptionDataParameters
- Sender : ExitGames.Client.Photon.Chat.ChatParameterCode
- sender : PhotonMessageInfo
- Senders: ExitGames.Client.Photon.Chat.ChatChannel, ExitGames.Client.Photon.Chat.ChatParameterCode
- SendMonoMessageTargets : PhotonNetwork
- SendMonoMessageTargetType : PhotonNetwork
- SendPrivate :
  - ExitGames.Client.Photon.Chat.ChatOperationCode
- SequenceChannel : RaiseEventOptions
- ServerAddress : ServerSettings
- ServerFull : ErrorCode , ExitGames.Client.Photon.Chat.ErrorCode
- ServerPort : ServerSettings
- ServerSettings : OperationCode
- SetProperties : EventCode , OperationCode
- SkipMessage:

### ExitGames.Client.Photon.Chat.ChatParameterCode

- SlotError : ErrorCode
- StartRpcsAsCoroutine : PhotonNetwork
- statsOn : PhotonStatsGuistatsRect : PhotonStatsGui
- statsWindowOn : PhotonStatsGui
- Status: ExitGames.Client.Photon.Chat.ChatParameterCode
- StatusUpdate: ExitGames.Client.Photon.Chat.ChatEventCode
- Subscribe: ExitGames.Client.Photon.Chat.ChatEventCode, ExitGames.Client.Photon.Chat.ChatOperationCode
- SubscribeResults:

   SubscribeResults:
  - ExitGames.Client.Photon.Chat.ChatParameterCode
- SuppressRoomEvents : ParameterCode
- synchronization : PhotonView
- SynchronizeEnabled : PhotonTransformViewPositionModel , PhotonTransformViewRotationModel , PhotonTransformViewScaleModel
- SynchronizeType: PhotonAnimatorView.SynchronizedLayer, PhotonAnimatorView.SynchronizedParameter

Online Documentation - Dashboard - Support Forum



v1.81

ľ	Mai	n P	age	Э	F	Rel	ate	d F	Paç	ges	5	M	odı	ıles	S	(	Cla	SS	es		File	es
C	Class	s Lis							C	lass	Hie	rarc	hy		Clas	s M	leml	oers	•			
F	AII		Fund	unctions Varia					es		Enu	ıme	ratio	ns		Enι	ıme	rato	r			
F	Prop	ertie	es																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	0	р	q	r	s	t	u	v	w	

### - t -

- TagObject : PhotonPlayer
- TargetActorNr : ParameterCode
- TargetActors : RaiseEventOptions
- TeleportEnabled : PhotonTransformViewPositionModel
- TeleportIfDistanceGreaterThan : PhotonTransformViewPositionModel
- trafficStatsOn : PhotonStatsGui
- Type: PhotonAnimatorView.SynchronizedParameter, TypedLobby

Online Documentation - Dashboard - Support Forum



v1.81

I	Mai	n P	Page Related					d F	Pages Modules						S	Classes					File	es
C	Class	Lis	t	C	lass	s Inc	dex		С	lass	s Hie	rarc	hy		Clas	s M	eml	oers	3			
F	All	F	=unc	tior	ıs		Vari	abl	es		En	ıme	ratio	ns		Enu	ıme	rato	r			
F	rop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	S	t	u	v	w	

#### - u -

- Unsubscribe: ExitGames.Client.Photon.Chat.ChatEventCode, ExitGames.Client.Photon.Chat.ChatOperationCode
- UpdateStatus :
  - ExitGames.Client.Photon.Chat.ChatOperationCode
- UriPath : ParameterCode
- UseNative : PhotonPingManager
- UsePrefabCache: PhotonNetwork
- UserBlocked : **ErrorCode** ,
  - ExitGames.Client.Photon.Chat.ErrorCode
- UserId : ActorProperties ,
  - ExitGames.Client.Photon.Chat.ChatParameterCode,
  - ExitGames.Client.Photon.Chat.ParameterCode,
  - **ParameterCode**
- UseRpcMonoBehaviourCache : PhotonNetwork
- Users : ExitGames.Client.Photon.Chat.ChatEventCode



v1.81

ı	Mai	n Page Related				d F	Pages Modules						Classes						File	es		
C	Class	s Lis	st	C	Class	s In	dex		С	lass	Hie	rarc	hy		Clas	s M	eml	ers	3			
F	AII	ı	Func	ction	าร		Vari	ablo	es		Enu	ıme	ratio	ns		Enι	ımeı	rato	r			
F	Prop	ertie	s																			
а	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	s	t	u	v	w	

#### - V -

• versionPUN : PhotonNetwork

• Visible : PhotonLagSimulationGui

• visibleField : RoomInfo

• VoiceAppID : ServerSettings

• VoiceServerPort : ServerSettings

Online Documentation - Dashboard - Support Forum



v1.81

ı	Mai	n Page Related				d F	Paç	ges	6	M	odı	ıles	5	Classes					File	es		
C	Class	s Lis	st	C	Class	s In	dex		С	lass	Hie	rarc	hy		Clas	s M	emb	ers	3			
F	All	ı	Func	ction	าร		Varia	ablo	es		Enu	ıme	ratio	ns		Enι	ımeı	rato	r			
F	Prop	ertie	s																			
a	b	С	d	е	f	g	h	i	j	ı	m	n	o	р	q	r	s	t	u	v	w	

#### - W -

• WebRpc : OperationCode

WebRpcParameters : ParameterCodeWebRpcReturnCode : ParameterCode

• WebRpcReturnMessage : ParameterCode

• Windowld : PhotonLagSimulationGui , PhotonStatsGui

• WindowRect : PhotonLagSimulationGui

Online Documentation - Dashboard - Support Forum



v1.81

Mair	Main Page			Related Pages				Classes	Files
Class	List	Class	Index	Clas	ss Hie	erarchy	Cla	ass Members	
All	Fund	ctions	Variab	oles	En	umeration	ıs	Enumerator	
Prope	erties								

- ExtrapolateOptions : PhotonTransformViewPositionModel
- HostingOption : ServerSettings
- InterpolateOptions: PhotonTransformViewPositionModel, PhotonTransformViewRotationModel, PhotonTransformViewScaleModel
- ParameterType : **PhotonAnimatorView**
- SynchronizeType : PhotonAnimatorView

Online Documentation - Dashboard - Support Forum



v1.81

Mair	Main Page			Page	s Mod	ules		Classes	Files
Class	List	Class	Index	Clas	ss Hierarchy	С	lass	Members	
All	Fund	ctions	Variab	oles	Enumeration	ns	Eı	numerator	
Prope	erties								

- BestRegion : ServerSettings
- Bool: PhotonAnimatorView
- Continuous : PhotonAnimatorView
- Disabled: PhotonAnimatorView.
  - PhotonTransformViewPositionModel .
  - PhotonTransformViewRotationModel .
  - **PhotonTransformViewScaleModel**
- Discrete: PhotonAnimatorView
- EstimatedSpeed : PhotonTransformViewPositionModel
- EstimateSpeedAndTurn : PhotonTransformViewPositionModel
- FixedSpeed : PhotonTransformViewPositionModel
- Float : PhotonAnimatorView
- Int : PhotonAnimatorView
- Lerp: PhotonTransformViewPositionModel, PhotonTransformViewRotationModel, PhotonTransformViewScaleModel
- MoveTowards : PhotonTransformViewScaleModel
- NotSet : ServerSettings
- OfflineMode : ServerSettings
- PhotonCloud : ServerSettings
- RotateTowards : PhotonTransformViewRotationModel
- SelfHosted : ServerSettings
- SynchronizeValues : PhotonTransformViewPositionModel
- Trigger: PhotonAnimatorView



v1.81

ı	Main Page Related				ed	Pag	es		M	odu	les	6	(	Cla	sse	es	File	S			
C	Class	s Lis	st	C	Clas	s In	dex		CI	ass	Hier	arch	ny		Clas	s M	eml	oers			
F	AII		Fund	ction	าร		Variables Enumerations Enumerator									r					
F	Properties																				
a	b	С	d	е	f	g	i	1	m	n	o	р	q	r	s	t	u	v			

#### - a -

- ActiveSceneBuildIndex : SceneManagerHelper
- ActiveSceneName : SceneManagerHelper
- allProperties : PhotonPlayer
- AllProperties : PhotonPlayer
- Appld : ExitGames.Client.Photon.Chat.ChatClient
- AppVersion : ExitGames.Client.Photon.Chat.ChatClient
- AuthGetParameters: AuthenticationValues,
   ExitGames.Client.Photon.Chat.AuthenticationValues
- AuthPostData: AuthenticationValues, ExitGames.Client.Photon.Chat.AuthenticationValues
- AuthType: AuthenticationValues,
   ExitGames.Client.Photon.Chat.AuthenticationValues
- AuthValues: ExitGames.Client.Photon.Chat.ChatClient, PhotonNetwork
- AutoCleanUp : RoomautoCleanUp : Room
- autoCleanUpPlayerObjects : PhotonNetwork
- autoJoinLobby : PhotonNetwork
- automaticallySyncScene : **PhotonNetwork**

### - b -

- BestRegion : PhotonPingManager
- BestRegionCodeCurrently: ServerSettings
- BestRegionCodeInPreferences : ServerSettings

#### - C -

- CanChat: ExitGames.Client.Photon.Chat.ChatClient
- ChatRegion : ExitGames.Client.Photon.Chat.ChatClient
- cleanupCacheOnLeave : RoomOptions
- CleanupCacheOnLeave : RoomOptions
- connected : PhotonNetwork
- connectedAndReady: PhotonNetwork
- connecting : PhotonNetwork
- connectionState : PhotonNetwork
- connectionStateDetailed : PhotonNetwork
- Count : PhotonStream
- countOfPlayers : PhotonNetwork
- countOfPlayersInRooms : PhotonNetwork
- countOfPlayersOnMaster : PhotonNetwork
- countOfRooms : PhotonNetwork
- CrcCheckEnabled : PhotonNetwork
- CreatorActorNr : PhotonView
- customProperties : PhotonPlayer
- CustomProperties: PhotonPlayer, RoomInfo
- customProperties : RoomInfo
- customRoomProperties : RoomOptions
- customRoomPropertiesForLobby : RoomOptions

### - d -

- DebugMessage : WebRpcResponse
- DebugOut : ExitGames.Client.Photon.Chat.ChatClient
- DisconnectedCause : ExitGames.Client.Photon.Chat.ChatClient
- Done: PhotonPingManager

- EnableLobbyStatistics : PhotonNetwork
- ExpectedUsers : RoomexpectedUsers : Room

#### - f -

- Friends : PhotonNetwork
- FriendsListAge : PhotonNetwork
- FrontendAddress: ExitGames.Client.Photon.Chat.ChatClient

### - g -

• gameVersion : PhotonNetwork

### - i -

- ID : PhotonPlayer
- inRoom : PhotonNetwork
- insideLobby : PhotonNetwork
- instantiationData : PhotonView
- IsDefault : TypedLobby
- isInactive : PhotonPlayer
- IsInactive : PhotonPlayer
- IsInRoom : FriendInfo
- isLocal : PhotonPlayer
- IsLocalClientInside : RoomInfo
- isLocalClientInside : RoomInfo
- isMasterClient : PhotonNetwork , PhotonPlayer
- IsMasterClient : PhotonPlayer
- isMessageQueueRunning : PhotonNetwork
- isMine : PhotonView
- isNonMasterClientInRoom : PhotonNetwork
- IsOnline : FriendInfo
- IsOpen : Room , RoomInfo
- isOpen : RoomOptions
- IsOpen: RoomOptions
- isOwnerActive : PhotonView
- IsPrivate: ExitGames.Client.Photon.Chat.ChatChannel

• isReading : PhotonStream

• isSceneView : PhotonView

• IsVisible : Room , RoomInfo

• isVisible : RoomOptions

• IsVisible : RoomOptions

• isWriting : PhotonStream

### -1-

lobby : PhotonNetwork

• LobbyStatistics : PhotonNetwork

#### - m -

masterClient : PhotonNetwork

• MaxPlayers : Room

• maxPlayers : Room , RoomInfo

• MaxPlayers : RoomInfo

• maxPlayers : RoomOptions

• MaxResendsBeforeDisconnect : PhotonNetwork

• MessageCount : ExitGames.Client.Photon.Chat.ChatChannel

#### - n -

Name : FriendInfo

• name : PhotonPlayer , Room

Name : Roomname : RoomInfo

• Name : RoomInfo , WebRpcResponse

• NameServerAddress :

ExitGames.Client.Photon.Chat.ChatClient, ExitGames.Client.Photon.Chat.ChatPeer

NetworkStatisticsEnabled : PhotonNetwork

networkView : Photon.MonoBehaviour

• NickName : PhotonPlayer

- offlineMode : PhotonNetwork
- open: Room, RoomInfo
- otherPlayers : PhotonNetwork
- owner : PhotonView
- OwnerActorNr : PhotonView

### - p -

- PacketLossByCrcCheck : PhotonNetwork
- Parameters : WebRpcResponse
- Peer: PhotonLagSimulationGui
- photonView : Photon.MonoBehaviour
- player : PhotonNetwork
- PlayerCount : Room
- playerCount : Room
- PlayerCount : RoomInfo
- playerCount : RoomInfo
- playerList : PhotonNetwork
- playerName : PhotonNetwork
- plugins : RoomOptions
- PrefabPool : PhotonNetwork
- prefix : PhotonView
- propertiesListedInLobby : Room
- PropertiesListedInLobby : Room
- publishUserId : RoomOptions
- PublishUserId : RoomOptions

### - q -

• QuickResends : PhotonNetwork

#### - r -

- removedFromList : RoomInfo
- ResentReliableCommands : PhotonNetwork
- ReturnCode : WebRpcResponse
- Room : FriendInfo
- room : PhotonNetwork

#### - S -

• sendRate : PhotonNetwork

sendRateOnSerialize : PhotonNetwork

• Server : PhotonNetwork

ServerAddress : PhotonNetwork

• ServerTimestamp : PhotonNetwork

• State: ExitGames.Client.Photon.Chat.ChatClient

• suppressRoomEvents : RoomOptions

• SuppressRoomEvents : RoomOptions

#### - t -

time : PhotonNetwork

• timestamp : PhotonMessageInfo

Token: AuthenticationValues,
 ExitGames.Client.Photon.Chat.AuthenticationValues

#### - u -

- unreliableCommandsLimit : PhotonNetwork
- Userld: AuthenticationValues,
   ExitGames.Client.Photon.Chat.AuthenticationValues,
   ExitGames.Client.Photon.Chat.ChatClient, PhotonPlayer
- userId : PhotonPlayer

#### - V -

viewID : PhotonView

• visible : Room , RoomInfo



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files
File List	File	e Members				
File Lie	st					

Here is a list of all files with brief descriptions:	
[d·	etail level 12345]
▼ Photon Unity Networking	
<b>▼</b> ■ Plugins	
▼ PhotonNetwork	
▼	
PhotonAnimatorView.cs	
PhotonRigidbody2DView.cs	
PhotonRigidbodyView.cs	
PhotonTransformView.cs	
PhotonTransformViewPositionControl.cs	
PhotonTransformViewPositionModel.cs	
PhotonTransformViewRotationControl.cs	
PhotonTransformViewRotationModel.cs	
PhotonTransformViewScaleControl.cs	
PhotonTransformViewScaleModel.cs	
<b>□</b> CustomTypes.cs	Sets up suppo Unity-specific t Can be a bluer to register your Custom Types sending.

Enums.cs	Wraps up seve commonly use enumerations.
Extensions.cs	
FriendInfo.cs	
EloadbalancingPeer.cs	
<sup>™</sup> NetworkingPeer.cs	
<b>■</b> PhotonClasses.cs	Wraps up sma classes that do their own file.
PhotonHandler.cs	
PhotonLagSimulationGui.cs	Part of the <b>Op1 GUI</b> .
PhotonNetwork.cs	
PhotonPlayer.cs	
PhotonStatsGui.cs	Part of the <b>Opt GUI</b> .
PhotonStreamQueue.cs	
PhotonView.cs	
PingCloudRegions.cs	
Room.cs	
RoomInfo.cs	
RPC.cs	Reimplements Attribute, as it's longer in all ve the <b>UnityEngi</b> assembly.
	Outdated. Here overwrite older import.
ServerSettings.cs	ScriptableObje defining a serv An instance is as

	PhotonServer
SocketWebTcp.cs	
<b>▼</b> ■ PhotonChatApi	
ChatChannel.cs	
<sup>□</sup> ChatClient.cs	
ChatDisconnectCause.cs	
ChatEventCode.cs	
ChatOperationCode.cs	
ChatParameterCode.cs	
ChatPeer.cs	
ChatState.cs	
ChatUserStatus.cs	
lChatClientListener.cs	

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related Pages Modules Classes Files

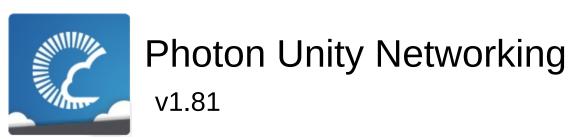
Photon Unity Networking

**Photon Unity Networking Directory Reference** 

### Directories

directory Plugins

Online Documentation - Dashboard - Support Forum



Main Page	Related Pages	Modules	Classes	Files
Photon Unity Netw	vorking Plugins			
Plugins D	irectory Refe	rence		

### Directories

### directory PhotonNetwork

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related Pages Modules Classes Files

Photon Unity Networking Plugins PhotonNetwork

PhotonNetwork Directory Reference

# Directories

### directory Views

#### **Files**

CustomTypes.cs file Sets up support for Unity-specific types. Can be a blueprint how to register your own Custom Types for sending. file Enums.cs Wraps up several of the commonly used enumerations. file Extensions.cs file FriendInfo.cs **GizmoType.cs** file LoadbalancingPeer.cs file file **NetworkingPeer.cs** file PhotonClasses.cs Wraps up smaller classes that don't need their own file. file PhotonHandler.cs **PhotonLagSimulationGui.cs** file Part of the Optional GUI. file PhotonNetwork.cs file PhotonPlayer.cs file PhotonStatsGui.cs

Part of the Optional GUI.

PhotonStreamQueue.cs

file

file PhotonView.cs

file PingCloudRegions.cs

file Room.cs

file RoomInfo.cs

file RPC.cs

Reimplements a RPC Attribute, as it's no longer in all versions of the **UnityEngine** assembly.

file RpcIndexComponent.cs

Outdated. Here to overwrite older files on import.

file ServerSettings.cs

ScriptableObject defining a server setup. An instance is created as **PhotonServerSettings**.

file SocketWebTcp.cs

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related Pages Modules Classes Files

Photon Unity Networking Plugins PhotonNetwork Views

Views Directory Reference

# Files

file	PhotonAnimatorView.cs
file	PhotonRigidbody2DView.cs
file	PhotonRigidbodyView.cs
file	PhotonTransformView.cs
file	PhotonTransformViewPositionControl.cs
file	PhotonTransformViewPositionModel.cs
file	PhotonTransformViewRotationControl.cs
file	PhotonTransformViewRotationModel.cs
file	PhotonTransformViewScaleControl.cs
file	PhotonTransformViewScaleModel.cs

Online Documentation - Dashboard - Support Forum



Main Page Related		Pages	Modules	Classes	Files				
File List	File	e Members							
Photon Unit	Photon Unity Networking Plugins PhotonNetwork Views								
PhotonAnimatorView.cs File Reference									

#### class PhotonAnimatorView

This class helps you to synchronize Mecanim animations Simply add the component to your GameObject and make sure that the **PhotonAnimatorView** is added to the list of observed components <u>More...</u>

class PhotonAnimatorView.SynchronizedParameter

class PhotonAnimatorView.SynchronizedLayer

Online Documentation - Dashboard - Support Forum



Main Page Related			Pages	Modules	CI	asses	Files		
File List	File	e Members							
Photon Unity Networking Plugins PhotonNetwork Views									
PhotonRigidbody2DView.cs File Reference									

#### class PhotonRigidbody2DView

This class helps you to synchronize the velocities of a 2d physics RigidBody. Note that only the velocities are synchronized and because Unitys physics engine is not deterministic (ie. the results aren't always the same on all computers) - the actual positions of the objects may go out of sync. If you want to have the position of this object the same on all clients, you should also add a **PhotonTransformView** to synchronize the position. Simply add the component to your GameObject and make sure that the

**PhotonRigidbody2DView** is added to the list of observed components <u>More...</u>

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related		Pages	Modules	Classes	Files			
File List	File	File Members						
Photon Unity Networking Plugins PhotonNetwork Views								
PhotonRigidbodyView.cs File Reference								

#### class PhotonRigidbodyView

This class helps you to synchronize the velocities of a physics RigidBody. Note that only the velocities are synchronized and because Unitys physics engine is not deterministic (ie. the results aren't always the same on all computers) - the actual positions of the objects may go out of sync. If you want to have the position of this object the same on all clients, you should also add a **PhotonTransformView** to synchronize the position. Simply add the component to your GameObject and make sure that the **PhotonRigidbodyView** is added to the list of observed components <u>More...</u>

Online Documentation - Dashboard - Support Forum



Main Page Related			Pages	Modules	CI	asses	Files	
File List	File	e Members						
Photon Unity Networking Plugins PhotonNetwork Views								
PhotonTransformView.cs File Reference								

#### class PhotonTransformView

This class helps you to synchronize position, rotation and scale of a GameObject. It also gives you many different options to make the synchronized values appear smooth, even when the data is only send a couple of times per second. Simply add the component to your GameObject and make sure that the **PhotonTransformView** is added to the list of observed components <u>More...</u>

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Rela		Related	Pages	Modules	Classes	Files			
File List	File List File Members								
Photon Unity Networking Plugins PhotonNetwork Views									
	Classes								
PhotonTransformViewPositionControl.cs File									
D - f		_							

# Reference

### class PhotonTransformViewPositionControl

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Rel		Rela	ited Pag	jes	Modules	S	Classes	Files	
File List	File List File Members								
Photon Unity Networking Plugins PhotonNetwork Views									
PhotonTransformViewPositionModel.cs File Reference									

### class PhotonTransformViewPositionModel

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related		Pages	Modules	C	lasses	Files			
File List	File	e Members							
Photon Unity Networking Plugins PhotonNetwork Views									
PhotonTransformViewRotationControl.cs File Reference									

### class PhotonTransformViewRotationControl

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related			ted Pag	es	Modules	5	Classes	Files	
File List	File List File Members								
Photon Unity Networking Plugins PhotonNetwork Views									
PhotonTransformViewRotationModel.cs File Reference									

### class PhotonTransformViewRotationModel

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related			Pages	Modules	Classes	Files			
File List File Members									
Photon Unity Networking Plugins PhotonNetwork Views									
PhotonTransformViewScaleControl.cs File Reference									

### class PhotonTransformViewScaleControl

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files				
File List	File	e Members								
Photon Unit	Photon Unity Networking Plugins PhotonNetwork Views									
Photon Unity Networking   Plugins   PhotonNetwork   Views   PhotonTransformViewScaleModel.cs File   Reference										

### class PhotonTransformViewScaleModel

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related			Pages	Modules	Classes	Files		
File List	File	e Members						
Photon Unity Networking Plugins PhotonNetwork								
CustomTypes.cs File Reference								

Sets up support for Unity-specific types. Can be a blueprint how to register your own Custom Types for sending. <u>More...</u>

### class **CustomTypes**

Internally used class, containing de/serialization methods for various Unity-specific classes. Adding those to the **Photon** serialization protocol allows you to send them in events, etc.

# **Detailed Description**

Sets up support for Unity-specific types. Can be a blueprint how to register your own Custom Types for sending.

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files
File List File Members						
Photon Unity Networking Plugins PhotonNetwork						
Enums.cs File Reference					<u>Classes</u>   !	Enumerations

Wraps up several of the commonly used enumerations. More...

### class EncryptionDataParameters

#### **Enumerations**

PhotonNetworkingMessage { PhotonNetworkingMessage.OnConnectedToPhoton. PhotonNetworkingMessage.OnLeftRoom. PhotonNetworkingMessage.OnMasterClientSwitched. PhotonNetworkingMessage.OnPhotonCreateRoomFailed, PhotonNetworkingMessage.OnPhotonJoinRoomFailed, PhotonNetworkingMessage.OnCreatedRoom, PhotonNetworkingMessage.OnJoinedLobby, PhotonNetworkingMessage.OnLeftLobby, PhotonNetworkingMessage.OnDisconnectedFromPhoton. PhotonNetworkingMessage.OnConnectionFail. PhotonNetworkingMessage.OnFailedToConnectToPhoton. PhotonNetworkingMessage.OnReceivedRoomListUpdate, PhotonNetworkingMessage.OnJoinedRoom, PhotonNetworkingMessage.OnPhotonPlayerConnected, PhotonNetworkingMessage.OnPhotonPlayerDisconnected, enum PhotonNetworkingMessage.OnPhotonRandomJoinFailed, PhotonNetworkingMessage.OnConnectedToMaster. PhotonNetworkingMessage.OnPhotonSerializeView. PhotonNetworkingMessage.OnPhotonInstantiate. PhotonNetworkingMessage.OnPhotonMaxCccuReached. PhotonNetworkingMessage.OnPhotonCustomRoomPrope PhotonNetworkingMessage.OnPhotonPlayerPropertiesCha PhotonNetworkingMessage.OnUpdatedFriendList, PhotonNetworkingMessage.OnCustomAuthenticationFailed PhotonNetworkingMessage.OnCustomAuthenticationRes PhotonNetworkingMessage.OnWebRpcResponse. PhotonNetworkingMessage.OnOwnershipReguest. PhotonNetworkingMessage.OnLobbyStatisticsUpdate. PhotonNetworkingMessage.OnPhotonPlayerActivityChan PhotonNetworkingMessage.OnOwnershipTransfered This enum defines the set of MonoMessages **Photon** Unity Net using as callbacks. Implemented by PunBehaviour. More...

enum

PhotonLogLevel { PhotonLogLevel.ErrorsOnly, PhotonLogLevel.Informational, PhotonLogLevel.Full }

Used to define the level of logging output created by the PUN cl log errors, info (some more) or full. <u>More...</u>

```
PhotonTargets {
        PhotonTargets.All, PhotonTargets.Others, PhotonTargets.
       PhotonTargets.AllBuffered.
enum
        PhotonTargets.OthersBuffered, PhotonTargets.AllViaServe
       PhotonTargets.AllBufferedViaServer
       Enum of "target" options for RPCs. These define which remote
       your RPC call. More...
       CloudRegionCode {
        CloudRegionCode.eu = 0, CloudRegionCode.us = 1,
       CloudRegionCode.asia = 2, CloudRegionCode.jp = 3,
        CloudRegionCode.au = 5, CloudRegionCode.usw = 6,
enum
       CloudRegionCode.sa = 7, CloudRegionCode.cae = 8,
        CloudRegionCode.kr = 9, CloudRegionCode.in = 10,
       CloudRegionCode.none = 4
       Currently available Photon Cloud regions as enum. More...
       CloudRegionFlag {
        CloudRegionFlag.eu = 1 << 0, CloudRegionFlag.us = 1 <<
       CloudRegionFlag.asia = 1 << 2, CloudRegionFlag.jp = 1 << 3
        CloudRegionFlag.au = 1 << 4, CloudRegionFlag.usw = 1 <<
enum
       CloudRegionFlag.sa = 1 << 6, CloudRegionFlag.cae = 1 << 1
        CloudRegionFlag.kr = 1 << 8, CloudRegionFlag.in = 1 << 9
       Available regions as enum of flags. To be used as "enabled" flag
       Region pinging. More...
       ConnectionState {
        ConnectionState.Disconnected, ConnectionState.Connect
       ConnectionState.Connected, ConnectionState.Disconnection
enum
        ConnectionState.InitializingApplication
       High level connection state of the client. Better use the more de
       ClientState. More...
```

enum

EncryptionMode { EncryptionMode.PayloadEncryption, EncryptionMode.DatagramEncryption = 10 }
Defines how the communication gets encrypted. More...

# **Detailed Description**

Wraps up several of the commonly used enumerations.

# **Enumeration Type Documentation**

#### enum CloudRegionCode

Currently available Photon Cloud regions as enum.

This is used in **PhotonNetwork.ConnectToRegion**.

Enumerator			
eu	European servers in Amsterdam.		
us	US servers (East Coast).		
asia	Asian servers in Singapore.		
jp	Japanese servers in Tokyo.		
au	Australian servers in Melbourne.		
	summary>USA West, San José, usw		
usw	summary>South America , Sao Paulo, sa		
sa	summary>Canada East, Montreal, cae		
cae	summary>South Korea, Seoul, kr		
kr	summary>India, Chennai, in		
in			
none	No region selected.		

#### enum CloudRegionFlag

Available regions as enum of flags. To be used as "enabled" flags for Best **Region** pinging.

Note that these enum values skip CloudRegionCode.none and their values are in strict order (power of 2).

Enumerator		
eu		
us		
asia		
jp		
au		
usw		
sa		
cae		
kr		
in		

#### enum ConnectionState

High level connection state of the client. Better use the more detailed **ClientState**.

Enumerator		
Disconnected		
Connecting		
Connected		
Disconnecting		
InitializingApplication		

### enum EncryptionMode

Defines how the communication gets encrypted.

Enumerator				
PayloadEncryption	This is the default encryption mode: Messages get encrypted only on demand (when you send operations with the "encrypt" parameter set to true).			

DatagramEncryption	With this encryption mode for UDP, the connection gets setup and all further datagrams get encrypted almost entirely. On-demand message encryption (like in PayloadEncryption) is skipped.
	This mode requires AuthOnce or AuthOnceWss as AuthMode!

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files
File List File Members						
Photon Unit	Photon Unity Networking Plugins PhotonNetwork					
Extensions.cs File Reference			Class	es   Typedefs		

#### Classes

#### class Extensions

This static class defines some useful extension methods for several existing classes (e.g. Vector3, float and others). More...

#### class **GameObjectExtensions**

Small number of extension methods that make it easier for PUN to work cross-Unity-versions. <u>More...</u>

# Typedefs

using **Hashtable** = ExitGames.Client.Photon.Hashtable

using **SupportClassPun** = ExitGames.Client.Photon.SupportClass

## **Typedef Documentation**

using Hashtable = ExitGames.Client.Photon.Hashtable

using SupportClassPun = ExitGames.Client.Photon.SupportClass

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files
File List File Members						
Photon Unit	Photon Unity Networking Plugins PhotonNetwork					
FriendInfo.cs File Reference						

#### Classes

#### class FriendInfo

Used to store info about a friend's online state and in which room he/she is. More...

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files
File List File Members						
Photon Unit	Photon Unity Networking Plugins PhotonNetwork					
GizmoType.cs File Reference		le	<u>Classes</u>	s   <u>Namespaces</u>   !	Enumerations	

## Classes

class ExitGames.Client.GUI.GizmoTypeDrawer

# Namespaces

package ExitGames.Client.GUI

#### **Enumerations**

ExitGames.Client.GUI.GizmoType {
 ExitGames.Client.GUI.GizmoType.WireSphere,
 enum ExitGames.Client.GUI.GizmoType.Sphere,
 ExitGames.Client.GUI.GizmoType.WireCube,
 ExitGames.Client.GUI.GizmoType.Cube }

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files
File List	File	e Members				
Photon Unit	Photon Unity Networking Plugins PhotonNetwork					
LoadbalancingPeer.cs File Reference			<u>Classes</u>   <u>!</u>	Enumerations		

#### Classes

#### class LoadBalancingPeer

Internally used by PUN. A LoadbalancingPeer provides the operations and enum definitions needed to use the loadbalancing server application which is also used in **Photon** Cloud.

#### class OpJoinRandomRoomParams

#### class EnterRoomParams

#### class **ErrorCode**

**ErrorCode** defines the default codes associated with **Photon** client/server communication. <u>More...</u>

#### class ActorProperties

Class for constants. These (byte) values define "well known" properties for an Actor / Player. Pun uses these constants internally. More...

#### class GamePropertyKey

Class for constants. These (byte) values are for "well known" room/game properties used in **Photon** Loadbalancing. Pun uses these constants internally. <u>More...</u>

#### class EventCode

Class for constants. These values are for events defined by **Photon** Loadbalancing. Pun uses these constants internally. More...

#### class ParameterCode

Class for constants. Codes for parameters of Operations and Events. Pun uses these constants internally. More...

#### class OperationCode

Class for constants. Contains operation codes. Pun uses these constants internally. <u>More...</u>

#### class RoomOptions

Wraps up common room properties needed when you create rooms. Read the individual entries for more details. More...

#### class RaiseEventOptions

Aggregates several less-often used options for operation RaiseEvent. See field descriptions for usage details. <u>More...</u>

#### class TypedLobby

Refers to a specific lobby (and type) on the server. More...

#### class TypedLobbyInfo

#### class AuthenticationValues

Container for user authentication in **Photon**. Set AuthValues before you connect - all else is handled. <u>More...</u>

#### **Enumerations**

JoinMode: byte { JoinMode.Default = 0, enum JoinMode.CreatelfNotExists = 1, JoinMode.JoinOrRejoin = 2, JoinMode.RejoinOnly = 3 } Defines possible values for OpJoinRoom and OpJoinOrCreate. It tells the server if the room can be only be

OpJoinOrCreate. It tells the server if the room can be only be joined normally, created implicitly or found on a web-service for Turnbased games. <u>More...</u>

enum O, MatchmakingMode.SerialMatching = 1,
MatchmakingMode.RandomMatching = 2 }
Options for matchmaking rules for OpJoinRandom. More...

enum ReceiverGroup: byte { ReceiverGroup.Others = 0, ReceiverGroup.All = 1, ReceiverGroup.MasterClient = 2 } Lite - OpRaiseEvent lets you chose which actors in the room should receive events. By default, events are sent to "Others" but you can overrule this. More...

```
EventCaching : byte {
       EventCaching.DoNotCache = 0.
      EventCaching.MergeCache = 1,
      EventCaching.ReplaceCache = 2,
      EventCaching.RemoveCache = 3,
       EventCaching.AddToRoomCache = 4,
      EventCaching.AddToRoomCacheGlobal = 5.
      EventCaching.RemoveFromRoomCache = 6.
enum
      EventCaching.RemoveFromRoomCacheForActorsLeft =
      7.
       EventCaching.SliceIncreaseIndex = 10,
      EventCaching.SliceSetIndex = 11,
      EventCaching.SlicePurgeIndex = 12,
      EventCaching.SlicePurgeUpToIndex = 13
      Lite - OpRaiseEvent allows you to cache events and
      automatically send them to joining players in a room. Events
```

are cached per event code and player: Event 100 (example!) can be stored once per player. Cached events can be modified, replaced and removed. More...

#### enum

**PropertyTypeFlag**: byte { **PropertyTypeFlag.None** = 0x00, PropertyTypeFlag.Game = 0x01, PropertyTypeFlag.Actor = 0x02, **PropertyTypeFlag.GameAndActor** = Game | Actor } Flags for "types of properties", being used as filter in OpGetProperties. More...

#### enum

**LobbyType**: byte { **LobbyType.Default** = 0, LobbyType.SqlLobby = 2, LobbyType.AsyncRandomLobby = 3 }

Options of lobby types available. Lobby types might be implemented in certain **Photon** versions and won't be available on older servers. More...

AuthModeOption { AuthModeOption.Auth, enum AuthModeOption.AuthOnce, AuthModeOption.AuthOnceWss }

> Options for authentication modes. From "classic" auth on each server to AuthOnce (on NameServer). More...

CustomAuthenticationType : byte { CustomAuthenticationType.Custom = 0, CustomAuthenticationType.Steam = 1, CustomAuthenticationType.Facebook = 2, CustomAuthenticationType.Oculus = 3, enum CustomAuthenticationType.PlayStation = 4, CustomAuthenticationType.Xbox = 5. **CustomAuthenticationType.None** = byte.MaxValue Options for optional "Custom Authentication" services used with **Photon**. Used by OpAuthenticate after connecting to Photon. More...

## **Enumeration Type Documentation**

#### enum AuthModeOption

Options for authentication modes. From "classic" auth on each server to AuthOnce (on NameServer).

Enumerator	
Auth	
AuthOnce	
AuthOnceWss	

#### enum CustomAuthenticationType : byte

Options for optional "Custom Authentication" services used with **Photon**. Used by OpAuthenticate after connecting to **Photon**.

Enumerator	
Custom	Use a custom authentification service. Currently the only implemented option.
Steam	Authenticates users by their Steam Account. Set auth values accordingly!
Facebook	Authenticates users by their Facebook Account. Set auth values accordingly!
Oculus	Authenticates users by their Oculus Account and token.
PlayStation	Authenticates users by their PSN Account and token.
Xbox	Authenticates users by their Xbox Account and XSTS token.
None	Disables custom authentification. Same as not providing any <b>AuthenticationValues</b> for connect

(more precisely for: OpAuthenticate).

#### enum EventCaching: byte

Lite - OpRaiseEvent allows you to cache events and automatically sent joining players in a room. Events are cached per event code and player 100 (example!) can be stored once per player. Cached events can be n replaced and removed.

Caching works only combination with ReceiverGroup options Others ar

Enumerator	
DoNotCache	Default value (not sent).
MergeCache	Will merge this event's keys those already cached.
ReplaceCache	Replaces the event cache for eventCode with this event's
RemoveCache	Removes this event (by eve from the cache.
AddToRoomCache	Adds an event to the room's
AddToRoomCacheGlobal	Adds this event to the cache actor 0 (becoming a "globall owned" event in the cache).
RemoveFromRoomCache	Remove fitting event from the cache.
RemoveFromRoomCacheForActorsLeft	Removes events of players already left the room (cleani
SliceIncreaseIndex	Increase the index of the slic cache.
SliceSetIndex	Set the index of the sliced carbon you must set RaiseEventOptions.CacheS for this.
SlicePurgeIndex	Purge cache slice with index Exactly one slice is removed

	cache. You must set RaiseEventOptions.CacheS for this.
SlicePurgeUpToIndex	Purge cache slices with spe index and anything lower the You must set RaiseEventOptions.CacheS for this.

#### enum JoinMode : byte

Defines possible values for OpJoinRoom and OpJoinOrCreate. It tells the server if the room can be only be joined normally, created implicitly or found on a web-service for Turnbased games.

These values are not directly used by a game but implicitly set.

Enumerator	
Default	Regular join. The room must exist.
CreateIfNotExists	Join or create the room if it's not existing. Used for OpJoinOrCreate for example.
JoinOrRejoin	The room might be out of memory and should be loaded (if possible) from a Turnbased webservice.
RejoinOnly	Only re-join will be allowed. If the user is not yet in the room, this will fail.

#### enum LobbyType : byte

Options of lobby types available. Lobby types might be implemented in certain **Photon** versions and won't be available on older servers.

Enumerator	
Default	This lobby is used unless another is defined by game or JoinRandom. Room-

	lists will be sent and JoinRandomRoom can filter by matching properties.
SqlLobby	This lobby type lists rooms like Default but JoinRandom has a parameter for SQL-like "where" clauses for filtering. This allows bigger, less, or and and combinations.
AsyncRandomLobby	This lobby does not send lists of games. It is only used for OpJoinRandomRoom. It keeps rooms available for a while when there are only inactive users left.

#### enum MatchmakingMode: byte

Options for matchmaking rules for OpJoinRandom.

Enumerator				
FillRoom	Fills up rooms (oldest first) to get players together as fast as possible. Default.			
	Makes most sense with MaxPlayers > 0 and games that can only start with more players.			
SerialMatching	Distributes players across available rooms sequentially but takes filter into account. Without filter, rooms get players evenly distributed.			
RandomMatching	Joins a (fully) random room. Expected properties must match but aside from this, any available room might be selected.			

#### enum PropertyTypeFlag : byte

Flags for "types of properties", being used as filter in OpGetProperties.

Enumerator	

None	(0x00) Flag type for no property type.
Game	(0x01) Flag type for game-attached properties.
Actor	(0x02) Flag type for actor related propeties.
GameAndActor	(0x01) Flag type for game AND actor properties. Equal to 'Game'

#### enum ReceiverGroup : byte

Lite - OpRaiseEvent lets you chose which actors in the room should receive events. By default, events are sent to "Others" but you can overrule this.

Enumerator				
Others Default value (not sent). Anyone else gets my event.				
All	Everyone in the current room (including this peer) will get this event.			
MasterClient	The server sends this event only to the actor with the lowest actorNumber.			
	The "master client" does not have special rights but is the one who is in this room the longest time.			

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related		Pages	Modules	Classes	Files	
File List	File	e Members				
Photon Unity Networking Plugins PhotonNetwork						
NetworkingPeer.cs File Reference						

### Classes

#### class NetworkingPeer

Implements **Photon** LoadBalancing used in PUN. This class is used internally by **PhotonNetwork** and not intended as public API.

# Typedefs

using **Hashtable** = ExitGames.Client.Photon.Hashtable

using **SupportClassPun** = ExitGames.Client.Photon.SupportClass

#### **Enumerations**

```
ClientState {
        ClientState.Uninitialized, ClientState.PeerCreated.
       ClientState.Oueued, ClientState.Authenticated,
        ClientState.JoinedLobby.
       ClientState.DisconnectingFromMasterserver,
       ClientState.ConnectingToGameserver,
       ClientState.ConnectedToGameserver,
        ClientState.Joining, ClientState.Joined,
       ClientState.Leaving,
       ClientState.DisconnectingFromGameserver,
enum
        ClientState.ConnectingToMasterserver.
       ClientState.OueuedComingFromGameserver.
       ClientState.Disconnecting, ClientState.Disconnected,
        ClientState.ConnectedToMaster,
       ClientState.ConnectingToNameServer,
       ClientState.ConnectedToNameServer,
       ClientState.DisconnectingFromNameServer,
        ClientState.Authenticating
       Detailed connection / networking peer state. PUN
       implements a loadbalancing and authentication workflow
       "behind the scenes", so some states will automatically
       advance to some follow up state. Those states are
       commented with "(will-change)". More...
```

# DisconnectCause.DisconnectByServerUserLimit = StatusCode.DisconnectByServerUserLimit, DisconnectCause.ExceptionOnConnect = StatusCode.ExceptionOnConnect, DisconnectCause.DisconnectByServerTimeout = StatusCode.DisconnectByServer, DisconnectCause.DisconnectByServerLogic = StatusCode.DisconnectByServerLogic, DisconnectCause.Exception = StatusCode.Exception, DisconnectCause.InvalidAuthentication = ErrorCode.InvalidAuthentication.

DisconnectCause.MaxCcuReached,
DisconnectCause.InvalidRegion = ErrorCode.InvalidRegion,
DisconnectCause.SecurityExceptionOnConnect =
StatusCode.SecurityExceptionOnConnect,
DisconnectCause.DisconnectByClientTimeout =
StatusCode.TimeoutDisconnect,
DisconnectCause.InternalReceiveException =
StatusCode.ExceptionOnReceive,
DisconnectCause.AuthenticationTicketExpired = 32753
}
Summarizes the cause for a disconnect. Used in:
OnConnectionFail and OnFailedToConnectToPhoton. More...

ServerConnection { ServerConnection.MasterServer, enum ServerConnection.GameServer, ServerConnection.NameServer }

Available server (types) for internally used field: server. More...

## **Typedef Documentation**

using Hashtable = ExitGames.Client.Photon.Hashtable

using SupportClassPun = ExitGames.Client.Photon.SupportClass

# **Enumeration Type Documentation**

#### enum ServerConnection

Available server (types) for internally used field: server.

**Photon** uses 3 different roles of servers: Name Server, Master Server and Game Server.

Enumerator				
MasterServer	This server is where matchmaking gets done and where clients can get lists of rooms in lobbies.			
GameServer	This server handles a number of rooms to execute and relay the messages between players (in a room).			
NameServer	This server is used initially to get the address (IP) of a Master Server for a specific region. Not used for <b>Photon</b> OnPremise (self hosted).			

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related		Pages	Modules	Classes	Files		
File List	File	e Members					
Photon Unit	Photon Unity Networking Plugins PhotonNetwork						
PhotonClasses.cs File Reference							

Wraps up smaller classes that don't need their own file. More...

#### Classes

#### interface IPunObservable

Defines the OnPhotonSerializeView method to make it easy to implement correctly for observable scripts. More...

#### interface IPunCallbacks

This interface is used as definition of all callback methods of PUN, except OnPhotonSerializeView. Preferably, implement them individually. <u>More...</u>

#### interface IPunPrefabPool

Defines all the methods that a Object Pool must implement, so that PUN can use it. More...

#### class Photon.MonoBehaviour

This class adds the property photonView, while logging a warning when your game still uses the networkView. More...

#### class Photon.PunBehaviour

This class provides a .photonView and all callbacks/events that PUN can call. Override the events/methods you want to use. More...

#### struct PhotonMessageInfo

Container class for info about a particular message, RPC or update. More...

#### class **PunEvent**

Defines **Photon** event-codes as used by PUN.

#### class PhotonStream

This container is used in **OnPhotonSerializeView()** to either provide incoming data of a **PhotonView** or for you to provide it. <u>More...</u>

class	HelpURL Empty implementation of the upcoming HelpURL of Unity 5.1. This one is only for compatibility of attributes. More
class	UnityEngine.SceneManagement.SceneManager Minimal implementation of the SceneManager for older Unity, up to v5.2. More
class	SceneManagerHelper
class	WebRpcResponse Reads an operation response of a WebRpc and provides convenient access to most common values. More

## Namespaces

package	Photon
package	UnityEngine.SceneManagement

## Typedefs

using **Hashtable** = ExitGames.Client.Photon.Hashtable

using **SupportClassPun** = ExitGames.Client.Photon.SupportClass

using **Photon.Hashtable** = ExitGames.Client.Photon.Hashtable

# **Detailed Description**

Wraps up smaller classes that don't need their own file.

## **Typedef Documentation**

using Hashtable = ExitGames.Client.Photon.Hashtable

using SupportClassPun = ExitGames.Client.Photon.SupportClass

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	Main Page Related		Pages	Modules	Classes	Files
File List	File	e Members				
Photon Unity Networking Plugins PhotonNetwork						
PhotonHandler.cs File Reference						

### class **PhotonHandler**

Internal Monobehaviour that allows **Photon** to run an Update loop.

# Typedefs

using **Debug** = UnityEngine.Debug

using **Hashtable** = ExitGames.Client.Photon.Hashtable

using **SupportClassPun** = ExitGames.Client.Photon.SupportClass

## **Typedef Documentation**

using Debug = UnityEngine.Debug

using Hashtable = ExitGames.Client.Photon.Hashtable

using SupportClassPun =
ExitGames.Client.Photon.SupportClass

Online Documentation - Dashboard - Support Forum



Main Page Related		Pages	Modules	Classes	Files			
File List	File	Members						
Photon Unit	Photon Unity Networking Plugins PhotonNetwork							
<u>Classes</u>								
PhotonLagSimulationGui.cs File Reference								

Part of the **Optional GUI**. <u>More...</u>

### class PhotonLagSimulationGui

This MonoBehaviour is a basic GUI for the **Photon** client's network-simulation feature. It can modify lag (fixed delay), jitter (random lag) and packet loss. <u>More...</u>

# **Detailed Description**

Part of the Optional GUI.

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files	
File List	File	e Members					
Photon Unit	Photon Unity Networking Plugins PhotonNetwork						
PhotonNetwork.cs File Reference					es   Typedefs		

### class **PhotonNetwork**

The main class to use the  ${\bf PhotonNetwork}$  plugin. This class is static.  $\underline{{\bf More...}}$ 

# Typedefs

using **Debug** = UnityEngine.Debug

using **Hashtable** = ExitGames.Client.Photon.Hashtable

# **Typedef Documentation**

using Debug = UnityEngine.Debug

using Hashtable = ExitGames.Client.Photon.Hashtable

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files
File List	File	e Members				
Photon Unit	y Netv	vorking Plu	ugins Pho	otonNetwork		
PhotonPlayer.cs File Reference						es   Typedefs

class PhotonPlayer
Summarizes a "player" within a room, identified (in that room) by actorID. More...

# Typedefs

using **Hashtable** = ExitGames.Client.Photon.Hashtable

# **Typedef Documentation**

### using Hashtable = ExitGames.Client.Photon.Hashtable

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files
File List	File	e Members				
Photon Unit	y Netv	vorking Plu	ugins Ph	otonNetwork >		
Photo: Refere		atsGui. e	cs File			Classes

Part of the  ${\bf Optional\ GUI.\ }\underline{{\bf More...}}$ 

### class PhotonStatsGui

Basic GUI to show traffic and health statistics of the connection to **Photon**, toggled by shift+tab. <u>More...</u>

# **Detailed Description**

Part of the Optional GUI.

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files
File List	File	e Members				
Photon Unit						
PhotonStreamQueue.cs File Reference						Classes

### class PhotonStreamQueue

The PhotonStreamQueue helps you poll object states at higher frequencies then what PhotonNetwork.sendRate dictates and then sends all those states at once when Serialize() is called. On the receiving end you can call Deserialize() and then the stream will roll out the received object states in the same order and timeStep they were recorded in. More...

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	Main Page Related			Modules	Classes	Files
File List	File	e Members				
Photon Unit	y Netv	vorking Pl	ugins Pho	otonNetwork		
PhotonView.cs File Reference					<u>Classes</u>   <u>I</u>	Enumerations

### class **PhotonView**

PUN's NetworkView replacement class for networking. Use it like a NetworkView. <u>More...</u>

### **Enumerations**

ViewSynchronization { ViewSynchronization.Off, ViewSynchronization.ReliableDeltaCompressed, enum ViewSynchronization.Unreliable, ViewSynchronization.UnreliableOnChange } OnSerializeTransform { OnSerializeTransform.OnlyPosition. OnSerializeTransform.OnlyRotation, enum OnSerializeTransform.OnlyScale, OnSerializeTransform.PositionAndRotation, OnSerializeTransform.All OnSerializeRigidBody { OnSerializeRigidBody.OnlyVelocity, enum OnSerializeRigidBody.OnlyAngularVelocity, OnSerializeRigidBody.All } OwnershipOption { OwnershipOption.Fixed, enum OwnershipOption.Takeover, OwnershipOption.Request } Options to define how Ownership Transfer is handled per PhotonView. More...

## **Enumeration Type Documentation**

### enum OnSerializeRigidBody

Enumerator	
OnlyVelocity	
OnlyAngularVelocity	
All	

#### enum OnSerializeTransform

Enumerator	
OnlyPosition	
OnlyRotation	
OnlyScale	
PositionAndRotation	
All	

### enum OwnershipOption

Options to define how Ownership Transfer is handled per **PhotonView**.

This setting affects how RequestOwnership and TransferOwnership work at runtime.

Enumerator	Enumerator							
Fixed	Ownership is fixed. Instantiated objects stick with their creator, scene objects always belong to the Master Client.							
Takeover	Ownership can be taken away from the current owner							

	who can't object.
Request	Ownership can be requested with <b>PhotonView.RequestOwnership</b> but the current owner has to agree to give up ownership.
	The current owner has to implement IPunCallbacks.OnOwnershipRequest to react to the ownership request.

enum ViewSynchronizatior	า
Enumerator	
Off	
ReliableDeltaCompressed	
Unreliable	
UnreliableOnChange	

Online Documentation - Dashboard - Support Forum Exit Games GmbH



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files
File List	File	e Members				
Photon Unit	y Netv	vorking Plu	ugins Pho	otonNetwork		
PingCloudRegions.cs File Reference						es   Typedefs

class

PingMonoEditor
Uses C# Socket class from System.Net.Sockets (as Unity usually does). More...

class **PhotonPingManager** 

# Typedefs

using **Debug** = UnityEngine.Debug

using **SupportClassPun** = ExitGames.Client.Photon.SupportClass

# **Typedef Documentation**

using Debug = UnityEngine.Debug

using SupportClassPun = ExitGames.Client.Photon.SupportClass

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related		Pages	Modules	Classes	Files		
File List	File	e Members					
Photon Unity Networking Plugins PhotonNetwork							
Room.cs File Reference						Classes	

### class Room

This class resembles a room that PUN joins (or joined). The properties are settable as opposed to those of a **RoomInfo** and you can close or hide "your" room. More...

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related		Pages	Modules	Classes	Files		
File List	File	e Members					
Photon Unity Networking Plugins PhotonNetwork							
RoomInfo.cs File Reference						Classes	

#### class RoomInfo

A simplified room with just the info required to list and join, used for the room listing in the lobby. The properties are not settable (open, MaxPlayers, etc). More...

Online Documentation - Dashboard - Support Forum



Main Pa	age Related		Pages	Modules	Classes	Files		
File List	File	e Members						
Photon Unity Networking Plugins PhotonNetwork								
RPC.cs File Reference						Classes		

Reimplements a RPC Attribute, as it's no longer in all versions of the **UnityEngine** assembly. <u>More...</u>

### class PunRPC

Replacement for RPC attribute with different name. Used to flag methods as remote-callable. <u>More...</u>

# **Detailed Description**

Reimplements a RPC Attribute, as it's no longer in all versions of the **UnityEngine** assembly.

Online Documentation - Dashboard - Support Forum



Main Page Related		Pages	Modules	Classes	Files		
File List	File	e Members					
Photon Unity Networking Plugins PhotonNetwork							
RpcIndexComponent.cs File Reference							

Outdated. Here to overwrite older files on import. More...

### **Detailed Description**

Outdated. Here to overwrite older files on import.

Online Documentation - Dashboard - Support Forum



v1.81

Main Pa	ge	Related	Pages	Modules	Classes	Files			
File List	File	e Members							
Photon Unit	Photon Unity Networking Plugins PhotonNetwork								
Server Refere		tings.c	s File			Classes			

ScriptableObject defining a server setup. An instance is created as  ${\bf PhotonServerSettings}.$ 

class	Region
class	ServerSettings Collection of connection-relevant settings, used internally by PhotonNetwork.ConnectUsingSettings. More

### **Detailed Description**

ScriptableObject defining a server setup. An instance is created as **PhotonServerSettings**.

Online Documentation - Dashboard - Support Forum



Main Pa	ge	Related	Pages	Modules	Classes	Files			
File List	File	Members	Members						
Photon Unit	y Netv	orking Plu	ugins Pho	otonNetwork					
Socke	SocketWebTcp.cs File Reference								

Online Documentation - Dashboard - Support Forum



v1.81

Main Page Related Pages Modules Classes Files

PhotonChatApi

### PhotonChatApi Directory Reference

### Files

file	ChatChannel.cs
file	ChatClient.cs
file	ChatDisconnectCause.cs
file	ChatEventCode.cs
file	ChatOperationCode.cs
file	ChatParameterCode.cs
file	ChatPeer.cs
file	ChatState.cs
file	ChatUserStatus.cs
file	IChatClientListener.cs

Online Documentation - Dashboard - Support Forum



Main Page Related		Pages	Modules	Classes	Files			
File List	File	e Members						
PhotonChat	PhotonChatApi >							
ChatC Refere		nel.cs	File		<u>Classes</u>	<u>Namespaces</u>		

class ExitGames.Client.Photon.Chat.ChatChannel

A channel of communication in **Photon Chat**, updated by **ChatClient** and provided as READ ONLY. <u>More...</u>

#### package ExitGames.Client.Photon.Chat

Online Documentation - Dashboard - Support Forum



Main Page Related		Pages	Modules	Classes	Files				
File List	File	e Members							
PhotonChat	PhotonChatApi >								
ChatC Refere		it.cs Fil	е		<u>Classes</u>	<u>Namespaces</u>			

class ExitGames.Client.Photon.Chat.ChatClient

Central class of the **Photon Chat** API to connect, handle channels and messages. <u>More...</u>

#### package ExitGames.Client.Photon.Chat

Online Documentation - Dashboard - Support Forum



Main Page		Related	Pages	Modules	Classes	Files
File List	File	e Members				
PhotonChat	Api $ angle$					
					Namesnaces I F	Enumerations

ChatDisconnectCause.cs File Reference

package ExitGames.Client.Photon.Chat

#### **Enumerations**

ExitGames.Client.Photon.Chat.ChatDisconnectCause.Non
ExitGames.Client.Photon.Chat.ChatDisconnectCause.Disco
ExitGames.Client.Photon.Chat.ChatDisconnectCause.Exce
ExitGames.Client.Photon.Chat.ChatDisconnectCause.Disco
ExitGames.Client.Photon.Chat.ChatDisconnectCause.Time
enum
ExitGames.Client.Photon.Chat.ChatDisconnectCause.Exce
ExitGames.Client.Photon.Chat.ChatDisconnectCause.Invali
ExitGames.Client.Photon.Chat.ChatDisconnectCause.MaxC
ExitGames.Client.Photon.Chat.ChatDisconnectCause.Invali
ExitGames.Client.Photon.Chat.ChatDisconnectCause.Invali
ExitGames.Client.Photon.Chat.ChatDisconnectCause.Opera
ExitGames.Client.Photon.Chat.ChatDisconnectCause.Custo
}
Enumaration of causes for Disconnects (used in LoadBalancing
More...

Online Documentation - Dashboard - Support Forum



Main Page Related		Pages	Modules	Classes	Files			
File List	File	e Members						
PhotonChat	PhotonChatApi >							
ChatE <sup>s</sup> Refere		tCode.	cs File		<u>Classes</u>	<u>Namespaces</u>		

class ExitGames.Client.Photon.Chat.ChatEventCode

Wraps up internally used constants in **Photon Chat** events. You don't have to use them directly usually. <u>More...</u>

#### package ExitGames.Client.Photon.Chat

Online Documentation - Dashboard - Support Forum



Main Page Related		Pages	Modules	Classes	Files	
File List	File	e Members				
PhotonChat	Api					
ChatO File Re	•	ationCo ence	ode.cs		<u>Classes</u>	<u>Namespaces</u>

class

ExitGames.Client.Photon.Chat.ChatOperationCode
Wraps up codes for operations used internally in Photon
Chat. You don't have to use them directly usually. More...

#### package ExitGames.Client.Photon.Chat

Online Documentation - Dashboard - Support Forum



Main Page Related		Pages	Modules	Classes	Files			
File List	File	e Members						
PhotonChat	PhotonChatApi >							
ChatPa File Re		meterC ence	ode.cs		<u>Classes</u>	<u>Namespaces</u>		

#### class ExitGames.Client.Photon.Chat.ChatParameterCode

Wraps up codes for parameters (in operations and events) used internally in **Photon Chat**. You don't have to use them directly usually. <u>More...</u>

#### package ExitGames.Client.Photon.Chat

Online Documentation - Dashboard - Support Forum



Main Page Related		Pages	Modules	Classes	Files				
File List	st File Members								
PhotonChat	PhotonChatApi >								
ChatPeer.cs File Reference				<u>Classes</u>	s   <u>Namespaces</u>   !	Enumerations			

#### class ExitGames.Client.Photon.Chat.ChatPeer

Provides basic operations of the **Photon Chat** server. This internal class is used by public **ChatClient**. More...

#### class ExitGames.Client.Photon.Chat.AuthenticationValues

Container for user authentication in **Photon**. Set AuthValues before you connect - all else is handled. <u>More...</u>

#### class ExitGames.Client.Photon.Chat.ParameterCode

#### class ExitGames.Client.Photon.Chat.ErrorCode

**ErrorCode** defines the default codes associated with **Photon** client/server communication. More...

package ExitGames.Client.Photon.Chat

#### **Enumerations**

```
ExitGames.Client.Photon.Chat.CustomAuthenticationType
O, ExitGames.Client.Photon.Chat.CustomAuthenticationType
1, ExitGames.Client.Photon.Chat.CustomAuthenticationType
2, ExitGames.Client.Photon.Chat.CustomAuthenticationType
3,
ExitGames.Client.Photon.Chat.CustomAuthenticationType
4, ExitGames.Client.Photon.Chat.CustomAuthenticationType
5, ExitGames.Client.Photon.Chat.CustomAuthenticationType
byte.MaxValue
}
Options for optional "Custom Authentication" services used with Used by OpAuthenticate after connecting to Photon. More...
```

Online Documentation - Dashboard - Support Forum



Main Page Related		Pages	Modules	Classes	Files				
File List	File Members								
PhotonChat	PhotonChatApi >								
ChatSi Refere		cs File			Namespaces   !	Enumerations			

package ExitGames.Client.Photon.Chat

#### **Enumerations**

enum

ExitGames.Client.Photon.Chat.ChatState.Uninitialized,
ExitGames.Client.Photon.Chat.ChatState.ConnectingToNan
ExitGames.Client.Photon.Chat.ChatState.ConnectedToNam
ExitGames.Client.Photon.Chat.ChatState.Authenticating,
ExitGames.Client.Photon.Chat.ChatState.Authenticated,
ExitGames.Client.Photon.Chat.ChatState.DisconnectingFro
ExitGames.Client.Photon.Chat.ChatState.ConnectingToFro
ExitGames.Client.Photon.Chat.ChatState.DisconnectingFro
ExitGames.Client.Photon.Chat.Chat

Online Documentation - Dashboard - Support Forum



Main Page		Related	Pages	Modules	Classes	Files				
File List	File	e Members								
PhotonChatApi >										
ChatU Refere		Status.	cs File		<u>Classes</u>	<u>Namespaces</u>				

#### class ExitGames.Client.Photon.Chat.ChatUserStatus

Contains commonly used status values for SetOnlineStatus. You can define your own. More...

#### package ExitGames.Client.Photon.Chat

Online Documentation - Dashboard - Support Forum



Main Page		Related Pages		Modules	Classes	Files				
File List	File	Members								
PhotonChatApi >										
IChatClientListener.cs File Reference				<u>Classes</u>	Namespaces					

## Classes

## interface ExitGames.Client.Photon.Chat.IChatClientListener

Callback interface for **Chat** client side. Contains callback methods to notify your app about updates. Must be provided to new **ChatClient** in constructor <u>More...</u>

# Namespaces

## package ExitGames.Client.Photon.Chat

Online Documentation - Dashboard - Support Forum



v1.81

Main Page			ı	Rel	ate	d	Paç	ges	3	M	od	ule	S	(	Cla	SS	es		Fil	es		
File List File Membe				ers																		
All Ty		Тур	ede	s	E	Enu	mei	ratio	ns		Enu	mer	ator									
a	С	d	е	f	g	h	i	j	k	ı	m	n	o	р	q	r	s	t	u	v	х	

Here is a list of all file members with links to the files they belong to:

#### - a -

- Actor : LoadbalancingPeer.cs
- AddToRoomCache : LoadbalancingPeer.cs
- AddToRoomCacheGlobal : LoadbalancingPeer.cs
- All : Enums.cs , LoadbalancingPeer.cs , PhotonView.cs
- AllBuffered : Enums.cs
- AllBufferedViaServer : Enums.cs
- AllViaServer : Enums.cs
- asia : Enums.cs
- AsyncRandomLobby : LoadbalancingPeer.cs
- au : Enums.cs
- Auth: LoadbalancingPeer.cs
- Authenticated : NetworkingPeer.cs
- Authenticating : NetworkingPeer.cs
- AuthenticationTicketExpired : NetworkingPeer.cs
- AuthModeOption : LoadbalancingPeer.cs
- AuthOnce : LoadbalancingPeer.cs
- AuthOnceWss : LoadbalancingPeer.cs

#### - C -

• cae: Enums.cs

• ClientState : NetworkingPeer.cs

- CloudRegionCode : Enums.cs
- CloudRegionFlag : Enums.cs
- Connected : Enums.cs
- ConnectedToGameserver : NetworkingPeer.cs
- ConnectedToMaster : NetworkingPeer.cs
- ConnectedToNameServer : NetworkingPeer.cs
- Connecting : Enums.cs
- ConnectingToGameserver : NetworkingPeer.cs
- ConnectingToMasterserver : NetworkingPeer.cs
- ConnectingToNameServer : NetworkingPeer.cs
- ConnectionState : Enums.cs
- CreatelfNotExists : LoadbalancingPeer.cs
- CreateRoom : NetworkingPeer.cs
- Custom : LoadbalancingPeer.cs
- CustomAuthenticationType : LoadbalancingPeer.cs

### - d -

- DatagramEncryption : Enums.cs
- Debug: PhotonHandler.cs, PhotonNetwork.cs, PingCloudRegions.cs
- Default : LoadbalancingPeer.cs
- DisconnectByClientTimeout : NetworkingPeer.cs
- DisconnectByServerLogic : NetworkingPeer.cs
- DisconnectByServerTimeout : NetworkingPeer.cs
- DisconnectByServerUserLimit : NetworkingPeer.cs
- DisconnectCause : NetworkingPeer.cs
- Disconnected : Enums.cs , NetworkingPeer.cs
- Disconnecting: Enums.cs, NetworkingPeer.cs
- DisconnectingFromGameserver : NetworkingPeer.cs
- DisconnectingFromMasterserver : **NetworkingPeer.cs**
- DisconnectingFromNameServer : NetworkingPeer.cs
- DoNotCache : LoadbalancingPeer.cs

#### - e -

• EncryptionMode: Enums.cs

• ErrorsOnly : Enums.cs

• eu : Enums.cs

- EventCaching : LoadbalancingPeer.cs
- Exception : NetworkingPeer.cs
- ExceptionOnConnect : NetworkingPeer.cs

### - f -

- Facebook : LoadbalancingPeer.cs
   FillRoom : LoadbalancingPeer.cs
- Fixed : PhotonView.cs
- Full: Enums.cs

### - g -

- Game : LoadbalancingPeer.cs
- GameAndActor : LoadbalancingPeer.cs
- GameServer : NetworkingPeer.cs

### - h -

Hashtable: Extensions.cs, NetworkingPeer.cs,
 PhotonClasses.cs, PhotonHandler.cs, PhotonNetwork.cs,
 PhotonPlayer.cs

### - i -

- in : Enums.cs
- Informational : Enums.cs
- InitializingApplication : Enums.cs
- InternalReceiveException : NetworkingPeer.cs
- InvalidAuthentication : NetworkingPeer.cs
- InvalidRegion : NetworkingPeer.cs

## - j -

- Joined : NetworkingPeer.cs
- JoinedLobby : NetworkingPeer.cs

- Joining : NetworkingPeer.cs
- JoinMode : LoadbalancingPeer.cs
- JoinOrCreateRoom : NetworkingPeer.cs
- JoinOrRejoin : LoadbalancingPeer.cs
- JoinRandomRoom : NetworkingPeer.cs
- JoinRoom : NetworkingPeer.cs
- jp : Enums.cs
- k -
  - kr : Enums.cs
- | -
  - Leaving : NetworkingPeer.cs
  - LobbyType : LoadbalancingPeer.cs
- m -
  - MasterClient : Enums.cs , LoadbalancingPeer.cs
  - MasterServer : NetworkingPeer.cs
  - MatchmakingMode : LoadbalancingPeer.cs
  - MaxCcuReached : NetworkingPeer.cs
  - MergeCache : LoadbalancingPeer.cs
- n -
  - NameServer : NetworkingPeer.cs
  - none : Enums.cs
  - None : LoadbalancingPeer.cs
- 0 -
  - Oculus : LoadbalancingPeer.cs
  - Off : PhotonView.cs
  - OnConnectedToMaster : Enums.cs

- OnConnectedToPhoton: Enums.cs
- OnConnectionFail: Enums.cs
- OnCreatedRoom : Enums.cs
- OnCustomAuthenticationFailed : Enums.cs
- OnCustomAuthenticationResponse : Enums.cs
- OnDisconnectedFromPhoton: Enums.cs
- OnFailedToConnectToPhoton: Enums.cs
- OnJoinedLobby : Enums.cs
- OnJoinedRoom : Enums.cs
- OnLeftLobby : Enums.cs
- OnLeftRoom : Enums.cs
- OnLobbyStatisticsUpdate : Enums.cs
- OnlyAngularVelocity : PhotonView.cs
- OnlyPosition : PhotonView.cs
- OnlyRotation : PhotonView.cs
- OnlyScale : PhotonView.cs
- OnlyVelocity : PhotonView.cs
- OnMasterClientSwitched: Enums.cs
- OnOwnershipRequest : Enums.cs
- OnOwnershipTransfered : Enums.cs
- OnPhotonCreateRoomFailed : Enums.cs
- OnPhotonCustomRoomPropertiesChanged : Enums.cs
- OnPhotonInstantiate : Enums.cs
- OnPhotonJoinRoomFailed : Enums.cs
- OnPhotonMaxCccuReached : Enums.cs
- OnPhotonPlayerActivityChanged : Enums.cs
- OnPhotonPlayerConnected : Enums.cs
- OnPhotonPlayerDisconnected : Enums.cs
- OnPhotonPlayerPropertiesChanged : **Enums.cs**
- OnPhotonRandomJoinFailed: Enums.cs
- OnPhotonSerializeView : Enums.cs
- OnReceivedRoomListUpdate : Enums.cs
- OnSerializeRigidBody: PhotonView.cs
- OnSerializeTransform : PhotonView.cs
- OnUpdatedFriendList : Enums.cs
- OnWebRpcResponse : Enums.cs
- Others : Enums.cs , LoadbalancingPeer.cs
- OthersBuffered : Enums.cs
- OwnershipOption : PhotonView.cs

### - p -

- PayloadEncryption : **Enums.cs**
- PeerCreated : NetworkingPeer.cs
- PhotonLogLevel: Enums.cs
- PhotonNetworkingMessage : Enums.cs
- PhotonTargets : Enums.cs
- PlayStation : LoadbalancingPeer.cs
- PositionAndRotation : PhotonView.cs
- PropertyTypeFlag : LoadbalancingPeer.cs

### - q -

- Queued : NetworkingPeer.cs
- QueuedComingFromGameserver : NetworkingPeer.cs

#### - r -

- RandomMatching : LoadbalancingPeer.cs
- ReceiverGroup : LoadbalancingPeer.cs
- RejoinOnly: LoadbalancingPeer.cs
- ReliableDeltaCompressed : PhotonView.cs
- RemoveCache : LoadbalancingPeer.cs
- RemoveFromRoomCache : LoadbalancingPeer.cs
- RemoveFromRoomCacheForActorsLeft : LoadbalancingPeer.cs
- ReplaceCache : LoadbalancingPeer.cs
- Request : PhotonView.cs

#### - S -

- sa: Enums.cs
- SecurityExceptionOnConnect : NetworkingPeer.cs
- SerialMatching : LoadbalancingPeer.cs
- ServerConnection : NetworkingPeer.cs
- SliceIncreaseIndex : LoadbalancingPeer.cs
- SlicePurgeIndex : LoadbalancingPeer.cs
- SlicePurgeUpToIndex : LoadbalancingPeer.cs
- SliceSetIndex : LoadbalancingPeer.cs

- SqlLobby : LoadbalancingPeer.cs
- Steam : LoadbalancingPeer.cs
- SupportClassPun: Extensions.cs, NetworkingPeer.cs, PhotonClasses.cs, PhotonHandler.cs, PingCloudRegions.cs

#### - t -

Takeover : PhotonView.cs

#### - u -

• Uninitialized : NetworkingPeer.cs

• Unreliable : PhotonView.cs

• UnreliableOnChange : PhotonView.cs

us : Enums.csusw : Enums.cs

#### - V -

• ViewSynchronization : PhotonView.cs

#### - X -

• Xbox : LoadbalancingPeer.cs

Online Documentation - Dashboard - Support Forum



v1.81

Main Page		ge	Related	Pages	Modules	Classes	Files
File List		File M	embers				
All	All Typede		Enume	erations	Enumerator		

- Debug: PhotonHandler.cs, PingCloudRegions.cs, PhotonNetwork.cs
- Hashtable: Extensions.cs, PhotonPlayer.cs,
   PhotonNetwork.cs, PhotonHandler.cs, PhotonClasses.cs,
   NetworkingPeer.cs
- SupportClassPun: Extensions.cs, PingCloudRegions.cs, PhotonHandler.cs, PhotonClasses.cs, NetworkingPeer.cs

Online Documentation - Dashboard - Support Forum



v1.81

Main Page		ge I	Related	Pages	Modules	Classes	Files
File L	File List		e Members				
All	All Typedefs		Enume	erations	Enumerator		

AuthModeOption : LoadbalancingPeer.cs

ClientState : NetworkingPeer.csCloudRegionCode : Enums.cs

CloudRegionFlag : Enums.csConnectionState : Enums.cs

• CustomAuthenticationType : LoadbalancingPeer.cs

• DisconnectCause : NetworkingPeer.cs

• EncryptionMode : Enums.cs

• EventCaching : LoadbalancingPeer.cs

• JoinMode : LoadbalancingPeer.cs

• LobbyType : LoadbalancingPeer.cs

• MatchmakingMode : LoadbalancingPeer.cs

• OnSerializeRigidBody : PhotonView.cs

• OnSerializeTransform : PhotonView.cs

• OwnershipOption : PhotonView.cs

• PhotonLogLevel: Enums.cs

• PhotonNetworkingMessage : Enums.cs

• PhotonTargets : Enums.cs

• PropertyTypeFlag : LoadbalancingPeer.cs

• ReceiverGroup : LoadbalancingPeer.cs

• ServerConnection : NetworkingPeer.cs

• ViewSynchronization : PhotonView.cs



v1.81

I	Main Page			ı	Re	late	ed	Pa	ges		M	odı	ule	S		Cla	SS	es	Files	
File List			Fil	е Ме	emb	ers														
P	All Ty		Тур	edet	s	ı	Enu	ıme	ratio	ons		Enu	mer	ator						
a	a c		е	f	g	i	j	k	ı	m	n	0	р	q	r	s	t	u	х	

#### - a -

• Actor : LoadbalancingPeer.cs

• AddToRoomCache : LoadbalancingPeer.cs

• AddToRoomCacheGlobal : LoadbalancingPeer.cs

• All : Enums.cs , LoadbalancingPeer.cs , PhotonView.cs

• AllBuffered : Enums.cs

• AllBufferedViaServer : Enums.cs

• AllViaServer : Enums.cs

• asia : Enums.cs

• AsyncRandomLobby : LoadbalancingPeer.cs

• au : Enums.cs

• Auth: LoadbalancingPeer.cs

Authenticated : NetworkingPeer.csAuthenticating : NetworkingPeer.cs

• AuthenticationTicketExpired : NetworkingPeer.cs

• AuthOnce : LoadbalancingPeer.cs

• AuthOnceWss: LoadbalancingPeer.cs

#### - C -

• cae : Enums.cs

• Connected : Enums.cs

• ConnectedToGameserver : **NetworkingPeer.cs** 

- ConnectedToMaster: NetworkingPeer.cs
- ConnectedToNameServer : NetworkingPeer.cs
- Connecting : Enums.cs
- ConnectingToGameserver : NetworkingPeer.cs
- ConnectingToMasterserver : NetworkingPeer.cs
- ConnectingToNameServer : NetworkingPeer.cs
- CreatelfNotExists : LoadbalancingPeer.cs
- CreateRoom : NetworkingPeer.cs
- Custom : LoadbalancingPeer.cs

#### - d -

- DatagramEncryption : **Enums.cs**
- Default : LoadbalancingPeer.cs
- DisconnectByClientTimeout : NetworkingPeer.cs
- DisconnectByServerLogic : NetworkingPeer.cs
- DisconnectByServerTimeout : NetworkingPeer.cs
- DisconnectByServerUserLimit : NetworkingPeer.cs
- Disconnected : Enums.cs , NetworkingPeer.cs
- Disconnecting : Enums.cs , NetworkingPeer.cs
- DisconnectingFromGameserver : NetworkingPeer.cs
- DisconnectingFromMasterserver : NetworkingPeer.cs
- DisconnectingFromNameServer : NetworkingPeer.cs
- DoNotCache : LoadbalancingPeer.cs

#### - e -

- ErrorsOnly : Enums.cs
- eu : Enums.cs
- Exception : NetworkingPeer.cs
- ExceptionOnConnect : NetworkingPeer.cs

### - f -

- Facebook : LoadbalancingPeer.csFillRoom : LoadbalancingPeer.cs
- Fixed : PhotonView.cs
- Full: Enums.cs

### - g -

- Game : LoadbalancingPeer.cs
- GameAndActor : LoadbalancingPeer.cs
- GameServer : NetworkingPeer.cs

### - i -

- in : Enums.cs
- Informational : Enums.cs
- InitializingApplication : Enums.cs
- InternalReceiveException : NetworkingPeer.cs
- InvalidAuthentication : NetworkingPeer.cs
- InvalidRegion : NetworkingPeer.cs

## - j -

- Joined : NetworkingPeer.cs
- JoinedLobby : NetworkingPeer.cs
- Joining : NetworkingPeer.cs
- JoinOrCreateRoom : NetworkingPeer.cs
- JoinOrRejoin : LoadbalancingPeer.cs
- JoinRandomRoom : NetworkingPeer.cs
- JoinRoom : NetworkingPeer.cs
- jp : Enums.cs

### - k -

- kr : Enums.cs
- | -
  - Leaving : NetworkingPeer.cs
- m -

- MasterClient : Enums.cs , LoadbalancingPeer.cs
- MasterServer : NetworkingPeer.cs
- MaxCcuReached : NetworkingPeer.cs
- MergeCache : LoadbalancingPeer.cs

#### - n -

- NameServer : NetworkingPeer.cs
- none : Enums.cs
- None : LoadbalancingPeer.cs

#### - 0 -

- Oculus : LoadbalancingPeer.cs
- Off: PhotonView.cs
- OnConnectedToMaster: Enums.cs
- OnConnectedToPhoton: Enums.cs
- OnConnectionFail: Enums.cs
- OnCreatedRoom : Enums.cs
- OnCustomAuthenticationFailed : Enums.cs
- OnCustomAuthenticationResponse : Enums.cs
- OnDisconnectedFromPhoton: Enums.cs
- OnFailedToConnectToPhoton: Enums.cs
- OnJoinedLobby: Enums.cs
- OnJoinedRoom : Enums.cs
- OnLeftLobby: Enums.cs
- OnLeftRoom : Enums.cs
- OnLobbyStatisticsUpdate : Enums.cs
- OnlyAngularVelocity : PhotonView.cs
- OnlyPosition : PhotonView.cs
- OnlyRotation : PhotonView.cs
- OnlyScale : PhotonView.cs
- OnlyVelocity: PhotonView.cs
- OnMasterClientSwitched : Enums.cs
- OnOwnershipRequest : Enums.cs
- OnOwnershipTransfered : Enums.cs
- OnPhotonCreateRoomFailed : Enums.cs
- OnPhotonCustomRoomPropertiesChanged : **Enums.cs**
- OnPhotonInstantiate : Enums.cs

- OnPhotonJoinRoomFailed: Enums.cs
- OnPhotonMaxCccuReached : Enums.cs
- OnPhotonPlayerActivityChanged : Enums.cs
- OnPhotonPlayerConnected : Enums.cs
- OnPhotonPlayerDisconnected : **Enums.cs**
- OnPhotonPlayerPropertiesChanged : **Enums.cs**
- OnPhotonRandomJoinFailed: Enums.cs
- OnPhotonSerializeView : Enums.cs
- OnReceivedRoomListUpdate : Enums.cs
- OnUpdatedFriendList: Enums.cs
- OnWebRpcResponse : Enums.cs
- Others : Enums.cs , LoadbalancingPeer.cs
- OthersBuffered : Enums.cs

### - p -

- PayloadEncryption : Enums.cs
- PeerCreated : NetworkingPeer.cs
- PlayStation : LoadbalancingPeer.cs
- PositionAndRotation : PhotonView.cs

### - q -

- Queued : NetworkingPeer.cs
- QueuedComingFromGameserver : NetworkingPeer.cs

#### - r -

- RandomMatching : LoadbalancingPeer.cs
- RejoinOnly : LoadbalancingPeer.cs
- ReliableDeltaCompressed : PhotonView.cs
- RemoveCache : LoadbalancingPeer.cs
- RemoveFromRoomCache : LoadbalancingPeer.cs
- RemoveFromRoomCacheForActorsLeft : LoadbalancingPeer.cs
- ReplaceCache : LoadbalancingPeer.cs
- Request : PhotonView.cs

- sa : Enums.cs
- SecurityExceptionOnConnect : NetworkingPeer.cs
- SerialMatching : LoadbalancingPeer.cs
- SliceIncreaseIndex : LoadbalancingPeer.cs
- SlicePurgeIndex : LoadbalancingPeer.cs
- SlicePurgeUpToIndex : LoadbalancingPeer.cs
- SliceSetIndex : LoadbalancingPeer.cs
- SqlLobby : LoadbalancingPeer.cs
- Steam : LoadbalancingPeer.cs

#### - t -

Takeover : PhotonView.cs

#### - u -

• Uninitialized : NetworkingPeer.cs

• Unreliable : PhotonView.cs

• UnreliableOnChange : PhotonView.cs

us : Enums.csusw : Enums.cs

#### - X -

• Xbox: LoadbalancingPeer.cs



v1.81

Main Page	<b>Related Pages</b>	Modules	Classes	Files
	_			

# **Related Pages**

Here is a list of all related documentation pages:

General Documentation	Brief overview of Photon, subscriptions, hosting options and how to start
<b>Network Simulation GUI</b>	Simple GUI element to control the built-in network condition simulation
Network Statistics GUI	The PhotonStatsGui is a simple GUI component to track and show networkmetrics at runtime
Public API Module	The Public API module rounds up the most commonly used classes of PUN

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

# **IPunObservable Member List**

This is the complete list of members for **IPunObservable**, including all inherited members.

OnPhotonSerializeView(PhotonStream stream, PhotonMessageInfo in

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

## **IPunCallbacks Member List**

This is the complete list of members for **IPunCallbacks**, including all inherited members.

OnConnectedToMaster()

OnConnectedToPhoton()

OnConnectionFail(DisconnectCause cause)

OnCreatedRoom()

OnCustomAuthenticationFailed(string debugMessage)

**OnCustomAuthenticationResponse**(Dictionary< string, object > data)

OnDisconnectedFromPhoton()

**OnFailedToConnectToPhoton**(DisconnectCause cause)

OnJoinedLobby()

OnJoinedRoom()

OnLeftLobby()

OnLeftRoom()

OnLobbyStatisticsUpdate()

**OnMasterClientSwitched**(PhotonPlayer newMasterClient)

OnOwnershipRequest(object[] viewAndPlayer)

**OnOwnershipTransfered**(object[] viewAndPlayers)

OnPhotonCreateRoomFailed(object[] codeAndMsg)

OnPhotonCustomRoomPropertiesChanged(Hashtable propertiesTha

**OnPhotonInstantiate**(PhotonMessageInfo info)

OnPhotonJoinRoomFailed(object[] codeAndMsg)

OnPhotonMaxCccuReached()

OnPhotonPlayerActivityChanged(PhotonPlayer otherPlayer)

OnPhotonPlayerConnected(PhotonPlayer newPlayer)

**OnPhotonPlayerDisconnected**(PhotonPlayer otherPlayer)

OnPhotonPlayerPropertiesChanged(object[] playerAndUpdatedProps

OnPhotonRandomJoinFailed(object[] codeAndMsg)

OnReceivedRoomListUpdate()

OnUpdatedFriendList()

**OnWebRpcResponse**(OperationResponse response)

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Related	Related Pages			Classes	Files
Class List	Class Index	Class Index Class Hi		Clas	ss Members	
Photon Pu	nBehaviour )					

## Photon.PunBehaviour Member List

This is the complete list of members for **Photon.PunBehaviour**, including all inherited members.

networkView

OnConnectedToMaster()

OnConnectedToPhoton()

**OnConnectionFail**(DisconnectCause cause)

OnCreatedRoom()

**OnCustomAuthenticationFailed**(string debugMessage)

**OnCustomAuthenticationResponse**(Dictionary< string, object > data)

OnDisconnectedFromPhoton()

**OnFailedToConnectToPhoton**(DisconnectCause cause)

OnJoinedLobby()

OnJoinedRoom()

OnLeftLobby()

OnLeftRoom()

OnLobbyStatisticsUpdate()

**OnMasterClientSwitched**(PhotonPlayer newMasterClient)

OnOwnershipRequest(object[] viewAndPlayer)

OnOwnershipTransfered(object[] viewAndPlayers)

OnPhotonCreateRoomFailed(object[] codeAndMsg)

OnPhotonCustomRoomPropertiesChanged(Hashtable propertiesTha

**OnPhotonInstantiate**(PhotonMessageInfo info)

OnPhotonJoinRoomFailed(object[] codeAndMsg)

OnPhotonMaxCccuReached()

**OnPhotonPlayerActivityChanged**(PhotonPlayer otherPlayer)

**OnPhotonPlayerConnected**(PhotonPlayer newPlayer)

OnPhotonPlayerDisconnected(PhotonPlayer otherPlayer)

OnPhotonPlayerPropertiesChanged(object[] playerAndUpdatedProps

OnPhotonRandomJoinFailed(object[] codeAndMsg)

OnReceivedRoomListUpdate()

OnUpdatedFriendList()

OnWebRpcResponse(OperationResponse response)

photonView

Online Documentation - Dashboard - Support Forum



v1.81

Main Page		Related	Pages	Modu	les	Classes	Files
Class List	C	lass Index	Class Hi	erarchy	Clas	ss Members	

# **PhotonMessageInfo Member List**

This is the complete list of members for **PhotonMessageInfo**, including all inherited members.

**PhotonMessageInfo**(PhotonPlayer player, int timestamp, PhotonView v photonView

sender

timestamp

ToString()

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

## **PhotonStream Member List**

This is the complete list of members for **PhotonStream**, including all inherited members.

Count	PhotonStream
isReading	PhotonStream
isWriting	PhotonStream
PeekNext()	PhotonStream
PhotonStream(bool write, object[] incomingData)	PhotonStream
ReceiveNext()	PhotonStream
SendNext(object obj)	PhotonStream
Serialize(ref bool myBool)	PhotonStream
Serialize(ref int myInt)	PhotonStream
Serialize(ref string value)	PhotonStream
Serialize(ref char value)	PhotonStream
Serialize(ref short value)	PhotonStream
Serialize(ref float obj)	PhotonStream
Serialize(ref PhotonPlayer obj)	PhotonStream
Serialize(ref Vector3 obj)	PhotonStream
Serialize(ref Vector2 obj)	PhotonStream
Serialize(ref Quaternion obj)	PhotonStream
SetReadStream(object[] incomingData, byte pos=0)	PhotonStream

ToArray() PhotonStream

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	е	Related	Pages	Modules		Classes	Files
Class List	C	Class Index	Class Hi	erarchy	Clas	ss Members	

## **PhotonNetwork Member List**

This is the complete list of members for **PhotonNetwork**, including all inherited members.

AllocateSceneViewID()

AllocateViewID()

**AuthValues** 

autoCleanUpPlayerObjects

autoJoinLobby

automaticallySyncScene

BackgroundTimeout

**CacheSendMonoMessageTargets**(Type type)

**CloseConnection**(PhotonPlayer kickPlayer)

connected

connectedAndReady

connecting

connectionState

connectionStateDetailed

**ConnectToBestCloudServer**(string gameVersion)

ConnectToMaster(string masterServerAddress, int port, string appID, st

ConnectToRegion(CloudRegionCode region, string gameVersion)

**ConnectUsingSettings**(string gameVersion)

countOfPlay(	ers
--------------	-----

countOfPlayersInRooms

countOfPlayersOnMaster

countOfRooms

CrcCheckEnabled

CreateRoom(string roomName)

**CreateRoom**(string roomName, RoomOptions roomOptions, TypedLobk **CreateRoom**(string roomName, RoomOptions roomOptions, TypedLobk

**Destroy**(PhotonView targetView)

**Destroy**(GameObject targetGo)

DestroyAll()

**DestroyPlayerObjects**(PhotonPlayer targetPlayer)

**DestroyPlayerObjects**(int targetPlayerId)

**Disconnect()** 

**EnableLobbyStatistics** 

**EventCallback**(byte eventCode, object content, int senderId)

FetchServerTimestamp()

FindFriends(string[] friendsToFind)

FindGameObjectsWithComponent(Type type)

**Friends** 

FriendsListAge

gameVersion

**GetCustomRoomList**(TypedLobby typedLobby, string sqlLobbyFilter)

**GetPing()** 

GetRoomList()

InitializeSecurity()

inRoom

insideLobby

Instantiate(string prefabName, Vector3 position, Quaternion rotation, int Instantiate(string prefabName, Vector3 position, Quaternion rotation, int InstantiateInRoomOnly

InstantiateSceneObject(string prefabName	, Vector3 position,	Quaterni
isMasterClient		

**isMessageQueueRunning** 

**isNonMasterClientInRoom** 

JoinLobby()

JoinLobby(TypedLobby)

JoinOrCreateRoom(string roomName, RoomOptions roomOptions, Typ JoinOrCreateRoom(string roomName, RoomOptions roomOptions, Typ JoinRandomRoom()

**JoinRandomRoom**(Hashtable expectedCustomRoomProperties, byte є **JoinRandomRoom**(Hashtable expectedCustomRoomProperties, byte є **JoinRoom**(string roomName)

JoinRoom(string roomName, string[] expectedUsers)

LeaveLobby()

LeaveRoom()

LoadLevel(int levelNumber)

LoadLevel(string levelName)

lobby

**LobbyStatistics** 

logLevel

masterClient

MAX\_VIEW\_IDS

maxConnections

MaxResendsBeforeDisconnect

**NetworkStatisticsEnabled** 

NetworkStatisticsReset()

NetworkStatisticsToString()

offlineMode

**OnEventCall** 

otherPlayers

OverrideBestCloudServer(CloudRegionCode region)

PacketLossByCrcCheck
PhotonServerSettings
player
playerList
playerName
precisionForFloatSynchronization
precisionForQuaternionSynchronization
precisionForVectorSynchronization
PrefabCache
PrefabPool
QuickResends
RaiseEvent(byte eventCode, object eventContent, bool sendReliable, F
Reconnect()
ReconnectAndRejoin()
RefreshCloudServerRating()
ReJoinRoom(string roomName)
RemovePlayerCustomProperties(string[] customPropertiesToDelete)
RemoveRPCs(PhotonPlayer targetPlayer)
RemoveRPCs(PhotonView targetPhotonView)
RemoveRPCsInGroup(int targetGroup)
ResentReliableCommands
room
SendMonoMessageTargets
SendMonoMessageTargetType
SendOutgoingCommands()
sendRate
sendRateOnSerialize
Server
ServerAddress
ServerTimestamp
SetLevelPrefix(short prefix)

**SetMasterClient**(PhotonPlayer masterClientPlayer)

**SetPlayerCustomProperties**(Hashtable customProperties)

**SetReceivingEnabled**(int group, bool enabled)

SetReceivingEnabled(int[] enableGroups, int[] disableGroups)

**SetSendingEnabled**(int group, bool enabled)

**SetSendingEnabled**(int[] enableGroups, int[] disableGroups)

**StartRpcsAsCoroutine** 

**SwitchToProtocol**(ConnectionProtocol cp)

time

UnAllocateViewID(int viewID)

unreliableCommandsLimit

**UsePrefabCache** 

UseRpcMonoBehaviourCache

versionPUN

**WebRpc**(string name, object parameters)

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Re	elated	Pages	s Modules		Classes	Files
Class List	Class I	ndex	Class Hi	erarchy	Clas	ss Members	

# **PhotonPlayer Member List**

This is the complete list of members for **PhotonPlayer**, including all inherited members.

**AllProperties** 

allProperties

**CompareTo**(PhotonPlayer other)

**CompareTo**(int other)

**customProperties** 

**CustomProperties** 

Equals(object p)

**Equals**(PhotonPlayer other)

**Equals**(int other)

Find(int ID)

Get(int id)

GetHashCode()

GetNext()

**GetNextFor**(PhotonPlayer currentPlayer)

**GetNextFor**(int currentPlayerId)

ID

isInactive

**IsInactive** 

IsLocal

isLocal

**isMasterClient** 

**IsMasterClient** 

name

**NickName** 

PhotonPlayer(bool isLocal, int actorID, string name)

SetCustomProperties(Hashtable propertiesToSet, Hashtable expected)

**TagObject** 

**ToString()** 

**ToStringFull()** 

UserId

userId

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Re	elated	Pages	s Modules		Classes	Files
Class List	Class I	ndex	Class Hi	erarchy	Clas	ss Members	

## **PhotonView Member List**

This is the complete list of members for **PhotonView**, including all inherited members.

CreatorActorNr

currentMasterID

**DeserializeView**(PhotonStream stream, PhotonMessageInfo info)

Find(int viewID)

**Get**(Component component)

Get(GameObject gameObj)

group

instantiationData

instantiationId

isMine

**isOwnerActive** 

**isSceneView** 

networkView

ObservedComponents

**OnMasterClientSwitched**(PhotonPlayer newMasterClient)

onSerializeRigidBodyOption

onSerializeTransformOption

owner

**OwnerActorNr** 

ownerld

ownershipTransfer

**OwnerShipWasTransfered** 

photonView

prefix

prefixBackup

RefreshRpcMonoBehaviourCache()

RequestOwnership()

RPC(string methodName, PhotonTargets target, params object[] parame RPC(string methodName, PhotonPlayer targetPlayer, params object[] parame RpcSecure(string methodName, PhotonTargets target, bool encrypt, parame RpcSecure(string methodName, PhotonPlayer targetPlayer, bool encrypt SerializeView(PhotonStream stream, PhotonMessageInfo info)

synchronization

ToString()

**TransferOwnership**(PhotonPlayer newOwner)

**TransferOwnership**(int newOwnerld)

viewID

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

## **Room Member List**

This is the complete list of members for **Room**, including all inherited members.

AutoCleanUp
autoCleanUp
autoCleanUpField
ClearExpectedUsers()
CustomProperties
customProperties
Equals(object other)
expectedUsers
ExpectedUsers
expectedUsersField
GetHashCode()
IsLocalClientInside
isLocalClientInside
IsOpen
IsVisible
MaxPlayers
maxPlayers
maxPlayersField

Name

name

nameField

open

openField

playerCount

**PlayerCount** 

propertiesListedInLobby

**PropertiesListedInLobby** 

removedFromList

**SetCustomProperties**(Hashtable propertiesToSet, Hashtable expected **SetPropertiesListedInLobby**(string[] propsListedInLobby)

ToString()

ToStringFull()

visible

visibleField

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### **RoomInfo Member List**

This is the complete list of members for **RoomInfo**, including all inherited members.

autoCleanUpField	RoomInfo	protected
CustomProperties	RoomInfo	
customProperties	RoomInfo	
Equals(object other)	RoomInfo	
expectedUsersField	RoomInfo	protected
GetHashCode()	RoomInfo	
isLocalClientInside	RoomInfo	
<b>IsLocalClientInside</b>	RoomInfo	
IsOpen	RoomInfo	
IsVisible	RoomInfo	
MaxPlayers	RoomInfo	
maxPlayers	RoomInfo	
maxPlayersField	RoomInfo	protected
name	RoomInfo	
Name	RoomInfo	
nameField	RoomInfo	protected
open	RoomInfo	
openField	RoomInfo	protected

playerCount	RoomInfo
PlayerCount	RoomInfo
removedFromList	RoomInfo
ToString()	RoomInfo
ToStringFull()	RoomInfo
visible	RoomInfo
visibleField	RoomInfo protected

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	е	Related Pages		Modules		Classes	Files
Class List	C	lass Index	Class Hi	erarchy	Clas	ss Members	

### **PhotonLagSimulationGui Member List**

This is the complete list of members for **PhotonLagSimulationGui**, including all inherited members.

OnGUI()	PhotonLagSimulationGui
Peer	PhotonLagSimulationGui
Start()	PhotonLagSimulationGui
Visible	PhotonLagSimulationGui
Windowld	PhotonLagSimulationGui
WindowRect	PhotonLagSimulationGui

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Relate	Related Pages		les	Classes	Files
Class List	Class Index	Class H	ierarchy	Clas	ss Members	

#### PhotonStatsGui Member List

This is the complete list of members for **PhotonStatsGui**, including all inherited members.

buttonsOn	<b>PhotonStatsGui</b>
healthStatsVisible	PhotonStatsGui
OnGUI()	PhotonStatsGui
Start()	PhotonStatsGui
statsOn	PhotonStatsGui
statsRect	PhotonStatsGui
statsWindowOn	PhotonStatsGui
trafficStatsOn	PhotonStatsGui
TrafficStatsWindow(int windowID)	PhotonStatsGui
Update()	PhotonStatsGui
Windowld	PhotonStatsGui

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	е	Related Pages		Modu	les	Classes	Files
Class List	С	lass Index	Class Hi	erarchy	Clas	ss Members	
ExitGames Client GUI GizmoTypeDrawer							

# ExitGames.Client.GUI.GizmoTypeDrawer Member List

This is the complete list of members for **ExitGames.Client.GUI.GizmoTypeDrawer**, including all inherited members.

Draw(Vector3 center, GizmoType type, Color color, float size) ExitGame

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	e I	Related	Pages	Modu	les	Classes	Files
Class List	Clas	s Index	Class Hi	erarchy	Clas	ss Members	
ExitGames	Client	Photon	Chat )	Authentica	ationVa	lues	

### ExitGames.Client.Photon.Chat.AuthenticationVa Member List

This is the complete list of members for **ExitGames.Client.Photon.Chat.AuthenticationValues**, including all inherited members.

AddAuthParameter(string key, string value)	<b>ExitGames.Client.Photon</b>
AuthenticationValues()	ExitGames.Client.Photon
AuthenticationValues(string userId)	<b>ExitGames.Client.Photon</b>
AuthGetParameters	ExitGames.Client.Photon
AuthPostData	<b>ExitGames.Client.Photon</b>
AuthType	ExitGames.Client.Photon
SetAuthPostData(string stringData)	<b>ExitGames.Client.Photon</b>
SetAuthPostData(byte[] byteData)	ExitGames.Client.Photon
Token	<b>ExitGames.Client.Photon</b>
ToString()	ExitGames.Client.Photon
UserId	<b>ExitGames.Client.Photon</b>

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	е	Related	Pages	Modu	lles	Classes	Files
Class List	Clas	s Index	Class Hi	erarchy	Clas	ss Members	
ExitGames	Client	Photon	Chat	ChatChan	nel		

# ExitGames.Client.Photon.Chat.ChatChannel Member List

This is the complete list of members for **ExitGames.Client.Photon.Chat.ChatChannel**, including all inherited members.

Add(string sender, object message)	ExitGames.Client.Photon.Ch
Add(string[] senders, object[] message	es) ExitGames.Client.Photon.Ch
ChatChannel(string name)	ExitGames.Client.Photon.Ch
ClearMessages()	ExitGames.Client.Photon.Ch
IsPrivate	ExitGames.Client.Photon.Ch
MessageCount	ExitGames.Client.Photon.Ch
MessageLimit	ExitGames.Client.Photon.Ch
Messages	ExitGames.Client.Photon.Ch
Name	ExitGames.Client.Photon.Ch
Senders	ExitGames.Client.Photon.Ch
ToStringMessages()	ExitGames.Client.Photon.Ch
TruncateMessages()	ExitGames.Client.Photon.Ch

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	e I	Related		Pages Modu		Classes	Files
Class List	Clas	s Index	Class	Hierarchy	Class Members		
ExitGames	Client	Photon	Chat	ChatClient			

# ExitGames.Client.Photon.Chat.ChatClient Member List

This is the complete list of members for **ExitGames.Client.Photon.Chat.ChatClient**, including all inherited members.

AddFriends(string[] friends)

**Appld** 

**AppVersion** 

**AuthValues** 

CanChat

**CanChatInChannel**(string channelName)

ChatClient(IChatClientListener listener, ConnectionProtocol protocol=ContentPeer

ChatRegion

Connect(string appld, string appVersion, AuthenticationValues authValu DebugOut

**Disconnect()** 

**DisconnectedCause** 

**FrontendAddress** 

**GetPrivateChannelNameByUser**(string userName)

MessageLimit

**NameServerAddress** 

**PrivateChannels** 

**PublicChannels** 

PublishMessage(string channelName, object message)

RemoveFriends(string[] friends)

SendAcksOnly()

**SendPrivateMessage**(string target, object message)

**SendPrivateMessage**(string target, object message, bool encrypt)

Service()

**SetOnlineStatus**(int status)

SetOnlineStatus(int status, object message)

**State** 

StopThread()

**Subscribe**(string[] channels)

**Subscribe**(string ☐ channels, int messagesFromHistory)

TryGetChannel(string channelName, bool isPrivate, out ChatChannel cl

**TryGetChannel**(string channelName, out ChatChannel channel)

Unsubscribe(string[] channels)

UserId

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	e I	Related Pages		Modules		Classes	Files
Class List	Clas	s Index	s Index Class Hie		Class Members		
ExitGames	Client	Photon	Chat ChatEventCode				

# ExitGames.Client.Photon.Chat.ChatEventCode Member List

This is the complete list of members for **ExitGames.Client.Photon.Chat.ChatEventCode**, including all inherited members.

ChatMessages	ExitGames.Client.Photon.Chat.ChatEventCode
FriendsList	ExitGames.Client.Photon.Chat.ChatEventCode
PrivateMessage	ExitGames.Client.Photon.Chat.ChatEventCode
StatusUpdate	ExitGames.Client.Photon.Chat.ChatEventCode
Subscribe	ExitGames.Client.Photon.Chat.ChatEventCode
Unsubscribe	ExitGames.Client.Photon.Chat.ChatEventCode
Users	ExitGames.Client.Photon.Chat.ChatEventCode

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	e I	Related	Pages	Modu	les	Classes	Files
Class List	Clas	Class Index Class Hie		erarchy	Clas	s Members	
ExitGames	Client	Photon	Chat ChatOperationCode				

### ExitGames.Client.Photon.Chat.ChatOperationCc Member List

This is the complete list of members for **ExitGames.Client.Photon.Chat.ChatOperationCode**, including all inherited members.

AddFriends	ExitGames.Client.Photon.Chat.ChatOperationCode
Authenticate	ExitGames.Client.Photon.Chat.ChatOperationCode
ChannelHistory	ExitGames.Client.Photon.Chat.ChatOperationCode
Publish	ExitGames.Client.Photon.Chat.ChatOperationCode
RemoveFriends	ExitGames.Client.Photon.Chat.ChatOperationCode
<b>SendPrivate</b>	ExitGames.Client.Photon.Chat.ChatOperationCode
Subscribe	ExitGames.Client.Photon.Chat.ChatOperationCode
Unsubscribe	ExitGames.Client.Photon.Chat.ChatOperationCode
<b>UpdateStatus</b>	ExitGames.Client.Photon.Chat.ChatOperationCode

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	e I	Related Page		Modules		Classes	Files
Class List	Clas	s Index	Class Hi	erarchy	Class Members		
ExitGames	Client	Photon	Chat ChatParameterCode				

### ExitGames.Client.Photon.Chat.ChatParameterCo Member List

This is the complete list of members for **ExitGames.Client.Photon.Chat.ChatParameterCode**, including all inherited members.

Channel	ExitGames.Client.Photon.Chat.ChatParameterC
Channels	ExitGames.Client.Photon.Chat.ChatParameterC
ChannelUserCount	ExitGames.Client.Photon.Chat.ChatParameterC
Friends	ExitGames.Client.Photon.Chat.ChatParameterC
HistoryLength	ExitGames.Client.Photon.Chat.ChatParameterC
Message	ExitGames.Client.Photon.Chat.ChatParameterC
Messages	ExitGames.Client.Photon.Chat.ChatParameterC
MsgId	ExitGames.Client.Photon.Chat.ChatParameterC
Msglds	ExitGames.Client.Photon.Chat.ChatParameterC
Secret	ExitGames.Client.Photon.Chat.ChatParameterC
Sender	ExitGames.Client.Photon.Chat.ChatParameterC
Senders	ExitGames.Client.Photon.Chat.ChatParameterC
SkipMessage	ExitGames.Client.Photon.Chat.ChatParameterC
Status	ExitGames.Client.Photon.Chat.ChatParameterC
SubscribeResults	ExitGames.Client.Photon.Chat.ChatParameterC

Online Documentation - Dashboard - Support Forum Exit Games GmbH



v1.81

Main Pag	e I	Related	Pages	Modu	les	Classes	Files
Class List	Clas	Class Index Class F		ierarchy	erarchy Class Members		
ExitGames	Client	Photon	Chat	ChatPeer			

### ExitGames.Client.Photon.Chat.ChatPeer Member List

This is the complete list of members for **ExitGames.Client.Photon.Chat.ChatPeer**, including all inherited members.

AuthenticateOnNameServer(string appld, string appVersion, string reg ChatPeer(IPhotonPeerListener listener, ConnectionProtocol protocol)
Connect()

**NameServerAddress** 

**NameServerHost** 

**NameServerHttp** 

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	е	Related Pages		Modules		Classes	Files
Class List	Clas	lass Index Class Hie		erarchy	Clas	ss Members	
ExitGames	Client	Photon	Chat ChatUserStatus				

### ExitGames.Client.Photon.Chat.ChatUserStatus Member List

This is the complete list of members for **ExitGames.Client.Photon.Chat.ChatUserStatus**, including all inherited members.

Away	ExitGames.Client.Photon.Chat.ChatUserStatus
DND	ExitGames.Client.Photon.Chat.ChatUserStatus
Invisible	ExitGames.Client.Photon.Chat.ChatUserStatus
LFG	ExitGames.Client.Photon.Chat.ChatUserStatus
Offline	ExitGames.Client.Photon.Chat.ChatUserStatus
Online	ExitGames.Client.Photon.Chat.ChatUserStatus
Playing	ExitGames.Client.Photon.Chat.ChatUserStatus

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	je Related		Pages Modu		les	Classes	Files
Class List	Clas	Class Index Class Hi		lierarchy	erarchy Class Members		
ExitGames	Client	Photon	Chat >	ErrorCode			

# ExitGames.Client.Photon.Chat.ErrorCode Member List

This is the complete list of members for **ExitGames.Client.Photon.Chat.ErrorCode**, including all inherited members.

CustomAuthenticationFailed	ExitGames.Client.Photon.Char
GameClosed	ExitGames.Client.Photon.Cha
GameDoesNotExist	ExitGames.Client.Photon.Char
GameFull	ExitGames.Client.Photon.Cha
GameldAlreadyExists	ExitGames.Client.Photon.Char
InternalServerError	ExitGames.Client.Photon.Cha
InvalidAuthentication	ExitGames.Client.Photon.Char
InvalidOperationCode	ExitGames.Client.Photon.Cha
InvalidRegion	ExitGames.Client.Photon.Cha
MaxCcuReached	ExitGames.Client.Photon.Cha
NoRandomMatchFound	ExitGames.Client.Photon.Cha
Ok	ExitGames.Client.Photon.Cha
OperationNotAllowedInCurrentState	ExitGames.Client.Photon.Cha
ServerFull	ExitGames.Client.Photon.Cha
UserBlocked	ExitGames.Client.Photon.Cha



v1.81

Main Pag	е	Related	Pages	Modu	les	Classes	Files
Class List	Clas	s Index	Class Hierarchy		Class Members		
ExitGames	Client	Photon	Chat	<b>IChatClien</b>	tListen	er	

# ExitGames.Client.Photon.Chat.IChatClientListen Member List

This is the complete list of members for **ExitGames.Client.Photon.Chat.IChatClientListener**, including all inherited members.

**DebugReturn**(DebugLevel level, string message)

**OnChatStateChange**(ChatState state)

OnConnected()

OnDisconnected()

OnGetMessages(string channelName, string[] senders, object[] message OnPrivateMessage(string sender, object message, string channelName OnStatusUpdate(string user, int status, bool gotMessage, object messa OnSubscribed(string[] channels, bool[] results)

OnUnsubscribed(string[] channels)

Online Documentation - Dashboard - Support Forum



v1.81

Main Pag	e I	Related	Pages	Modu	les	Classes	Files
Class List	Clas	s Index	Class Hierarchy		Class Members		
ExitGames	Client	Photon	Chat \	Parameter	Code	)	

# ExitGames.Client.Photon.Chat.ParameterCode Member List

This is the complete list of members for **ExitGames.Client.Photon.Chat.ParameterCode**, including all inherited members.

Address	ExitGames.Client.Photon.Chat.Parame
ApplicationId	${\bf Exit Games. Client. Photon. Chat. Parame}$
AppVersion	ExitGames.Client.Photon.Chat.Parame
ClientAuthenticationData	${\bf Exit Games. Client. Photon. Chat. Parameter {\bf Chat. Parameter} {\bf Chat. Paramete$
ClientAuthenticationParams	ExitGames.Client.Photon.Chat.Parame
ClientAuthenticationType	${\bf Exit Games. Client. Photon. Chat. Parame}$
Region	ExitGames.Client.Photon.Chat.Parame
Secret	ExitGames.Client.Photon.Chat.Parame
UserId	ExitGames.Client.Photon.Chat.Parame

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hierarchy		Class Members		
Photon Mo	noBehaviour					

#### Photon, Mono Behaviour Member List

This is the complete list of members for **Photon.MonoBehaviour**, including all inherited members.

networkView Photon.MonoBehaviour photonView Photon.MonoBehaviour

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hierarchy		Class Members		
UnityEngine	SceneManagen	nent Scer	neManager	$\rightarrow$		

### UnityEngine.SceneManagement.SceneManager Member List

This is the complete list of members for **UnityEngine.SceneManagement.SceneManager**, including all inherited members.

LoadScene(string name) UnityEngine.SceneManagement.SceneMa LoadScene(int buildIndex) UnityEngine.SceneManagement.SceneMa

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

### **ActorProperties Member List**

This is the complete list of members for **ActorProperties**, including all inherited members.

IsInactive ActorProperties
PlayerName ActorProperties
UserId ActorProperties

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	ain Page Related Pa		Modules		Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

### **AuthenticationValues Member List**

This is the complete list of members for **AuthenticationValues**, including all inherited members.

AddAuthParameter(string key, string value)	<b>AuthenticationValues</b>	virt
AuthenticationValues()	AuthenticationValues	
AuthenticationValues(string userId)	AuthenticationValues	
AuthGetParameters	AuthenticationValues	
AuthPostData	AuthenticationValues	
AuthType	AuthenticationValues	
SetAuthPostData(string stringData)	AuthenticationValues	virt
SetAuthPostData(byte[] byteData)	AuthenticationValues	virt
Token	AuthenticationValues	
ToString()	AuthenticationValues	
UserId	AuthenticationValues	

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Re	Related Pages		Modules		Classes	Files
Class List	Class I	ndex	Class Hi	erarchy	Clas	ss Members	

### **EncryptionDataParameters Member List**

This is the complete list of members for **EncryptionDataParameters**, including all inherited members.

Mode EncryptionDataParameters

Secret1 EncryptionDataParameters

**Secret2 EncryptionDataParameters** 

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### **ErrorCode Member List**

This is the complete list of members for **ErrorCode**, including all inherited members.

AlreadyMatched	ErrorCode
AuthenticationTicketExpired	<b>ErrorCode</b>
CustomAuthenticationFailed	ErrorCode
ExternalHttpCallFailed	<b>ErrorCode</b>
GameClosed	ErrorCode
GameDoesNotExist	<b>ErrorCode</b>
GameFull	ErrorCode
GameldAlreadyExists	<b>ErrorCode</b>
HttpLimitReached	ErrorCode
InternalServerError	<b>ErrorCode</b>
InvalidAuthentication	ErrorCode
InvalidEncryptionParameters	<b>ErrorCode</b>
InvalidOperation	ErrorCode
InvalidOperationCode	<b>ErrorCode</b>
InvalidRegion	ErrorCode
JoinFailedFoundActiveJoiner	<b>ErrorCode</b>
<b>JoinFailedFoundExcludedUserId</b>	ErrorCode
JoinFailedFoundInactiveJoiner	<b>ErrorCode</b>

<b>JoinFailedPeerAlreadyJoined</b>	<b>ErrorCode</b>
JoinFailedWithRejoinerNotFound	<b>ErrorCode</b>
MaxCcuReached	ErrorCode
NoRandomMatchFound	<b>ErrorCode</b>
Ok	ErrorCode
OperationNotAllowedInCurrentState	<b>ErrorCode</b>
PluginMismatch	ErrorCode
PluginReportedError	<b>ErrorCode</b>
ServerFull	ErrorCode
SlotError	<b>ErrorCode</b>
UserBlocked	ErrorCode

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### **EventCode Member List**

This is the complete list of members for **EventCode**, including all inherited members.

AppStats	<b>EventCode</b>
AuthEvent	<b>EventCode</b>
AzureNodeInfo	<b>EventCode</b>
CacheSliceChanged	<b>EventCode</b>
ErrorInfo	<b>EventCode</b>
GameList	<b>EventCode</b>
GameListUpdate	<b>EventCode</b>
Join	<b>EventCode</b>
Join Leave	EventCode EventCode
Leave	EventCode
Leave LobbyStats	EventCode EventCode
Leave LobbyStats Match	EventCode EventCode

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	е	Related	Pages	Modu	les	Classes	Files
Class List	Cla	ass Index	Class Hi	erarchy	Clas	ss Members	

#### **Extensions Member List**

This is the complete list of members for **Extensions**, including all inherited members.

AlmostEquals (this Vector3 target, Vector3 second, float sqrMagnitudeP AlmostEquals (this Vector2 target, Vector2 second, float sqrMagnitudeP AlmostEquals (this Quaternion target, Quaternion second, float maxAng AlmostEquals (this float target, float second, float floatDiff)

Contains(this int∏ target, int nr)

**GetCachedParemeters**(this MethodInfo mo)

**GetPhotonView**(this UnityEngine.GameObject go)

GetPhotonViewsInChildren(this UnityEngine.GameObject go)

Merge(this IDictionary target, IDictionary addHash)

MergeStringKeys(this IDictionary target, IDictionary addHash)

**ParametersOfMethods** 

StripKeysWithNullValues(this IDictionary original)

StripToStringKeys(this IDictionary original)

**ToStringFull**(this IDictionary origin)

**ToStringFull**(this object[] data)

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### FriendInfo Member List

This is the complete list of members for **FriendInfo**, including all inherited members.

IsInRoom FriendInfo

IsOnline FriendInfo

Name FriendInfo

Room FriendInfo

**ToString() FriendInfo** 

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

### **GameObjectExtensions Member List**

This is the complete list of members for **GameObjectExtensions**, including all inherited members.

GetActive(this GameObject target) GameObjectExtensions Static

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

### **GamePropertyKey Member List**

This is the complete list of members for **GamePropertyKey**, including all inherited members.

CleanupCacheOnLeave	GamePropertyKey
ExpectedUsers	GamePropertyKey
IsOpen	GamePropertyKey
IsVisible	GamePropertyKey
MasterClientId	GamePropertyKey
MaxPlayers	GamePropertyKey
PlayerCount	GamePropertyKey
PropsListedInLobby	GamePropertyKey
Removed	GamePropertyKey

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

### **HelpURL Member List**

This is the complete list of members for **HelpURL**, including all inherited members.

HelpURL(string url) HelpURL

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Relate	d Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	ierarchy	Clas	ss Members	

#### **IPunPrefabPool Member List**

This is the complete list of members for **IPunPrefabPool**, including all inherited members.

Destroy(GameObject gameObject)

**IPunF** 

Instantiate(string prefabld, Vector3 position, Quaternion rotation) IPunF

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

### **OperationCode Member List**

This is the complete list of members for **OperationCode**, including all inherited members.

Authenticate	OperationCode
AuthenticateOnce	OperationCode
ChangeGroups	OperationCode
CreateGame	OperationCode
ExchangeKeysForEncryption	OperationCode
FindFriends	OperationCode
GetGameList	OperationCode
GetLobbyStats	OperationCode
GetProperties	OperationCode
GetRegions	OperationCode
Join	OperationCode
JoinGame	OperationCode
JoinLobby	OperationCode
JoinRandomGame	OperationCode
Leave	OperationCode
LeaveLobby	OperationCode
RaiseEvent	OperationCode
ServerSettings	OperationCode

SetProperties WebRpc	OperationCode OperationCode
<u> </u>	<u> </u>

Online Documentation - Dashboard - Support Forum Exit Games GmbH



v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### **ParameterCode Member List**

This is the complete list of members for **ParameterCode**, including all inherited members.

ActorList	ParameterCode
ActorNr	ParameterCode
Add	ParameterCode
Address	ParameterCode
ApplicationId	ParameterCode
AppVersion	ParameterCode
AzureLocalNodeld	ParameterCode
AzureMasterNodeld	ParameterCode
AzureNodeInfo	ParameterCode
Broadcast	ParameterCode
Cache	ParameterCode
CacheSliceIndex	ParameterCode
CheckUserOnJoin	ParameterCode
CleanupCacheOnLeave	ParameterCode
ClientAuthenticationData	ParameterCode
ClientAuthenticationParams	ParameterCode
ClientAuthenticationType	ParameterCode
Code	ParameterCode

CustomEventContent	ParameterCode
CustomInitData	ParameterCode
Data	ParameterCode
EmptyRoomTTL	ParameterCode
EncryptionData	ParameterCode
EncryptionMode	ParameterCode
EventForward	ParameterCode
ExpectedProtocol	ParameterCode
ExpectedValues	ParameterCode
FindFriendsRequestList	ParameterCode
FindFriendsResponseOnlineList	ParameterCode
FindFriendsResponseRoomldList	ParameterCode
GameCount	ParameterCode
GameList	ParameterCode
GameProperties	ParameterCode
Group	ParameterCode
Info	ParameterCode
IsComingBack	ParameterCode
Islnactive	ParameterCode
JoinMode	ParameterCode
LobbyName	ParameterCode
LobbyStats	ParameterCode
LobbyType	ParameterCode
MasterClientId	ParameterCode
MasterPeerCount	ParameterCode
MatchMakingType	ParameterCode
NickName	ParameterCode
PeerCount	ParameterCode
PlayerProperties	ParameterCode
PlayerTTL	ParameterCode
PluginName	ParameterCode

Plugins	<b>ParameterCode</b>
PluginVersion	<b>ParameterCode</b>
Position	<b>ParameterCode</b>
Properties	<b>ParameterCode</b>
PublishUserId	<b>ParameterCode</b>
ReceiverGroup	<b>ParameterCode</b>
Region	<b>ParameterCode</b>
Remove	<b>ParameterCode</b>
RoomName	<b>ParameterCode</b>
Secret	<b>ParameterCode</b>
SuppressRoomEvents	<b>ParameterCode</b>
TargetActorNr	ParameterCode
UriPath	<b>ParameterCode</b>
UserId	ParameterCode
WebRpcParameters	<b>ParameterCode</b>
WebRpcReturnCode	<b>ParameterCode</b>
WebRpcReturnMessage	<b>ParameterCode</b>

Online Documentation - Dashboard - Support Forum Exit Games GmbH



v1.81

Main Page	е	Related	Pages	Modu	les	Classes	Files
Class List	C	Class Index Class Hie		erarchy	Clas	ss Members	

#### **PhotonAnimatorView Member List**

This is the complete list of members for **PhotonAnimatorView**, including all inherited members.

CacheDiscreteTriggers()

**DoesLayerSynchronizeTypeExist**(int layerIndex)

**DoesParameterSynchronizeTypeExist**(string name)

**GetLayerSynchronizeType**(int layerIndex)

**GetParameterSynchronizeType**(string name)

**GetSynchronizedLayers()** 

**GetSynchronizedParameters()** 

OnPhotonSerializeView(PhotonStream stream, PhotonMessageInfo in ParameterType enum name

**SetLayerSynchronized**(int layerIndex, SynchronizeType synchronizeTy **SetParameterSynchronized**(string name, ParameterType type, SynchronizeType enum name

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	е	Related Pages		Modules		Classes	Files
Class List	C	lass Index	Class Hierarchy		Class Members		
PhotonAnimatorView > SynchronizedLayer >							

#### PhotonAnimatorView.SynchronizedLayer Member List

This is the complete list of members for **PhotonAnimatorView.SynchronizedLayer**, including all inherited members.

LayerIndex PhotonAnimatorView.SynchronizedLayer
SynchronizeType PhotonAnimatorView.SynchronizedLayer

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Related	Related Pages		les	Classes	Files
Class List	Class Index	Class Hierarchy		Clas	ss Members	
PhotonAnimat	torView Synch					

# PhotonAnimatorView.SynchronizedParameter Member List

This is the complete list of members for **PhotonAnimatorView.SynchronizedParameter**, including all inherited members.

Name PhotonAnimatorView.SynchronizedParameter
SynchronizeType PhotonAnimatorView.SynchronizedParameter
Type PhotonAnimatorView.SynchronizedParameter

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

### PhotonPingManager Member List

This is the complete list of members for **PhotonPingManager**, including all inherited members.

Attempts	PhotonPingManager static
BestRegion	PhotonPingManager
Done	PhotonPingManager
IgnoreInitialAttempt	PhotonPingManager static
MaxMilliseconsPerPing	PhotonPingManager static
PingSocket(Region region)	PhotonPingManager
ResolveHost(string hostName)	PhotonPingManager static
UseNative	PhotonPingManager

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### PhotonRigidbody2DView Member List

This is the complete list of members for **PhotonRigidbody2DView**, including all inherited members.

OnPhotonSerializeView(PhotonStream stream, PhotonMessageInfo in

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### **PhotonRigidbodyView Member List**

This is the complete list of members for **PhotonRigidbodyView**, including all inherited members.

OnPhotonSerializeView(PhotonStream stream, PhotonMessageInfo in

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### **PhotonStreamQueue Member List**

This is the complete list of members for **PhotonStreamQueue**, including all inherited members.

Deserialize(PhotonStream stream)	PhotonStreamQueue
HasQueuedObjects()	PhotonStreamQueue
PhotonStreamQueue(int sampleRate)	PhotonStreamQueue
ReceiveNext()	PhotonStreamQueue
Reset()	PhotonStreamQueue
SendNext(object obj)	PhotonStreamQueue
Serialize(PhotonStream stream)	PhotonStreamQueue

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Relate	d Pages	Modu	les	Classes	Files
Class List	Class Index	Class H	ierarchy	Clas	ss Members	

#### **PhotonTransformView Member List**

This is the complete list of members for **PhotonTransformView**, including all inherited members.

**OnPhotonSerializeView**(PhotonStream stream, PhotonMessageInfo in **SetSynchronizedValues**(Vector3 speed, float turnSpeed)

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

# PhotonTransformViewPositionControl Member List

This is the complete list of members for **PhotonTransformViewPositionControl**, including all inherited members.

**GetExtrapolatedPositionOffset()** 

**GetNetworkPosition()** 

OnPhotonSerializeView(Vector3 currentPosition, PhotonStream stream PhotonTransformViewPositionControl(PhotonTransformViewPosition SetSynchronizedValues(Vector3 speed, float turnSpeed)

UpdatePosition(Vector3 currentPosition)

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Relate	d Pages	Modu	les	Classes	Files
Class List	Class Index	Class H	ierarchy	Clas	ss Members	

# PhotonTransformViewPositionModel Member List

This is the complete list of members for **PhotonTransformViewPositionModel**, including all inherited members.

DrawErrorGizmo	<b>PhotonTransformViewPositi</b>
ExtrapolateIncludingRoundTripTime	<b>PhotonTransformViewPositi</b>
ExtrapolateNumberOfStoredPositions	<b>PhotonTransformViewPositi</b>
ExtrapolateOption	${\bf Photon Transform View Position}$
ExtrapolateOptions enum name	<b>PhotonTransformViewPositi</b>
ExtrapolateSpeed	<b>PhotonTransformViewPositi</b>
InterpolateLerpSpeed	<b>PhotonTransformViewPositi</b>
InterpolateMoveTowardsAcceleration	${\bf Photon Transform View Position}$
InterpolateMoveTowardsDeceleration	<b>PhotonTransformViewPositi</b>
InterpolateMoveTowardsSpeed	${\bf Photon Transform View Position}$
InterpolateOption	<b>PhotonTransformViewPositi</b>
InterpolateOptions enum name	<b>PhotonTransformViewPositi</b>
InterpolateSpeedCurve	<b>PhotonTransformViewPositi</b>
SynchronizeEnabled	<b>PhotonTransformViewPositi</b>
TeleportEnabled	<b>PhotonTransformViewPositi</b>
<b>TeleportIfDistanceGreaterThan</b>	${\bf Photon Transform View Position}$



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

# PhotonTransformViewRotationControl Member List

This is the complete list of members for **PhotonTransformViewRotationControl**, including all inherited members.

GetNetworkRotation()

**GetRotation**(Quaternion currentRotation)

**OnPhotonSerializeView**(Quaternion currentRotation, PhotonStream str **PhotonTransformViewRotationControl**(PhotonTransformViewRotation

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	d Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	ierarchy	Clas	ss Members	

# PhotonTransformViewRotationModel Member List

This is the complete list of members for **PhotonTransformViewRotationModel**, including all inherited members.

InterpolateLerpSpeed	PhotonTransformViewRotationMo
InterpolateOption	<b>PhotonTransformViewRotationMo</b>
InterpolateOptions enum name	<b>PhotonTransformViewRotationMo</b>
InterpolateRotateTowardsSpeed	<b>PhotonTransformViewRotationMo</b>
SynchronizeEnabled	<b>PhotonTransformViewRotationMo</b>

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	e Relate	d Pages	Modu	les	Classes	Files
Class List	Class Index	Class H	ierarchy	Clas	ss Members	

# PhotonTransformViewScaleControl Member List

This is the complete list of members for **PhotonTransformViewScaleControl**, including all inherited members.

**GetNetworkScale()** 

GetScale(Vector3 currentScale)

OnPhotonSerializeView(Vector3 currentScale, PhotonStream stream, February PhotonTransformViewScaleControl(PhotonTransformViewScaleMode)

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	d Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	ierarchy	Clas	ss Members	

#### PhotonTransformViewScaleModel Member List

This is the complete list of members for **PhotonTransformViewScaleModel**, including all inherited members.

InterpolateLerpSpeed	<b>PhotonTransformViewScaleModel</b>
InterpolateMoveTowardsSpeed	<b>PhotonTransformViewScaleModel</b>
InterpolateOption	<b>PhotonTransformViewScaleModel</b>
InterpolateOptions enum name	<b>PhotonTransformViewScaleModel</b>
SynchronizeEnabled	<b>PhotonTransformViewScaleModel</b>

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	е	Related	Pages	Modu	les	Classes	Files
Class List	C	lass Index	Class Hi	erarchy	Clas	ss Members	

#### **PingMonoEditor Member List**

This is the complete list of members for **PingMonoEditor**, including all inherited members.

Dispose() PingMonoEditor

Done() PingMonoEditor

StartPing(string ip) PingMonoEditor

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### **RaiseEventOptions Member List**

This is the complete list of members for **RaiseEventOptions**, including all inherited members.

CachingOption	RaiseEventOptions	
Default	RaiseEventOptions	static
Encrypt	RaiseEventOptions	
ForwardToWebhook	RaiseEventOptions	
InterestGroup	RaiseEventOptions	
Receivers	RaiseEventOptions	
SequenceChannel	RaiseEventOptions	
TargetActors	RaiseEventOptions	

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### **Region Member List**

This is the complete list of members for **Region**, including all inherited members.

Code Region

HostAndPort Region

Parse(string codeAsString) Region Static

Ping Region ToString() Region

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	d Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	ierarchy	Clas	ss Members	

#### **RoomOptions Member List**

This is the complete list of members for **RoomOptions**, including all inherited members.

CleanupCacheOnLeave	RoomOptions
cleanupCacheOnLeave	RoomOptions
CustomRoomProperties	RoomOptions
customRoomProperties	RoomOptions
CustomRoomPropertiesForLobby	RoomOptions
$custom {\bf Room Properties For Lobby}$	RoomOptions
EmptyRoomTtl	RoomOptions
isOpen	RoomOptions
IsOpen	RoomOptions
isVisible	RoomOptions
IsVisible	RoomOptions
maxPlayers	RoomOptions
MaxPlayers	RoomOptions
PlayerTtl	RoomOptions
plugins	RoomOptions
Plugins	RoomOptions
publishUserId	RoomOptions
PublishUserId	RoomOptions

SuppressRoomEvents	RoomOptions	
suppressRoomEvents	RoomOptions	
Online Documentation - Dashboard	- Support Forum	Exit Games GmbH



v1.81

Main Page	Related	d Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	ierarchy	Clas	ss Members	

#### SceneManagerHelper Member List

This is the complete list of members for **SceneManagerHelper**, including all inherited members.

ActiveSceneBuildIndex SceneManagerHelper Static SceneManagerHelper Static

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### **ServerSettings Member List**

This is the complete list of members for **ServerSettings**, including all inherited members.

AppID	S
BestRegionCodeCurrently	S
BestRegionCodeInPreferences	S
ChatAppID	S
DisableAutoOpenWizard	S
EnabledRegions	S
<b>EnableLobbyStatistics</b>	S
HostingOption enum name	S
HostType	S
IsAppId(string val)	S
JoinLobby	S
NetworkLogging	S
PreferredRegion	S
Protocol	S
PunLogging	S
ResetBestRegionCodeInPreferences()	S
RpcList	S
RunInBackground	S

ServerAddress	S
ServerPort	S
ToString()	S
UseCloud(string cloudAppid)	S
UseCloud(string cloudAppid, CloudRegionCode code)	S
UseCloudBestRegion(string cloudAppid)	S
UseMyServer(string serverAddress, int serverPort, string application)	S
VoiceAppID	S
VoiceServerPort	S

Online Documentation - Dashboard - Support Forum Exit Games GmbH



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### **TypedLobby Member List**

This is the complete list of members for **TypedLobby**, including all inherited members.

Default	TypedLobby static
IsDefault	TypedLobby
Name	TypedLobby
ToString()	TypedLobby
Туре	TypedLobby
TypedLobby()	TypedLobby
TypedLobby(string name, LobbyType type)	TypedLobby

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	l Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### **TypedLobbyInfo Member List**

This is the complete list of members for **TypedLobbyInfo**, including all inherited members.

Default	TypedLobby	static
IsDefault	TypedLobby	
Name	TypedLobby	
PlayerCount	TypedLobbyInfo	
RoomCount	TypedLobbyInfo	
ToString()	${\bf TypedLobbyInfo}$	
Туре	TypedLobby	
TypedLobby()	TypedLobby	
TypedLobby(string name, LobbyType type)	TypedLobby	

Online Documentation - Dashboard - Support Forum



v1.81

Main Page	Related	Pages	Modu	les	Classes	Files
Class List	Class Index	Class Hi	erarchy	Clas	ss Members	

#### **WebRpcResponse Member List**

This is the complete list of members for **WebRpcResponse**, including all inherited members.

DebugMessage	WebRpcResponse
Name	WebRpcResponse
Parameters	WebRpcResponse
ReturnCode	WebRpcResponse
ToStringFull()	WebRpcResponse
WebRpcResponse(OperationResponse response)	WebRpcResponse

Online Documentation - Dashboard - Support Forum