Photon

Outline

```
1. PhotonEngine
2. MonoBehaviourPUN
3. PhotonView
5. IPunCallback
4. IPunObserbable
6. RaiseEvent
7. NetworkManager
8. RoomProperties
```

> **[RPC]** (**R**emote **P**rocedure **C**alls) **[Command]** Server command

static class PhotonEngine {}

> Main **static** - *toolkit for interaction with Photon Network*

```
- string NickName{get;}
- bool IsMasterClient{get;}
- RoomOptions GetRoomList{get;}

- PhotonEngine.Connect();
- PhotonEngine.JoinLobby();

- PhotonEngine.JoinOrCreate();

- PhotonEngine.JoinRoom();
- PhotonEngine.JoinRandomRoomRoom();
- PhotonEngine.LeaveRoom();

- PhotonEngine.RPC("name",BufferType.AllBuffer);
- PhotonEngine.Instantiate(GameObject go);
        //Must be in "Resource" Folder

- PhotonEngine.LoadLevel();
- PhotonEngine.OfflineMode();

- PhotonEngine.GetPing();
```

Extra

- Player list
- Player.custom.properties

class MonoBehaviourPun

> MonoBehaviour from **PUN** - *easy ref to photonView*

```
private PhotonView photonView;
```

class PhotonView : MonoBehaviour

> PhotonView- *core networing asset*

```
- public bool IsMine{get;} // Is local instance
- public bool IsSceneView{get;} // Is local instance
- public int  Id{get;} // Is local instance
- public PhotonView Owner// Is local instance

- public IPunObservable[] watched;

- [RPC] public void Some(); // new command that will be executed;
```

class IPunCallBack

> IPCallBack - *Interface hooked to network events*

```
- public void OnConnectedToMaster;
- public void OnPlayerJoinedRoom;
- public void OnPlayLeavingRoom;
- public void OnRoomUpdate(RoomInfo room);
```

class IPunObsorvable

> MonoBehaviour from **PUN** - *easy ref to photonView*

```
void Stream OnPhotonSerialization(NetworkStream stream)
    {
        int health =5;

        if(stream.isReading)
            health = (int)stream.ReceiveNext();

        if(stream.isWriting)
        stream.SendNext(helath);
    }
```

SubTypes

- class **PhotonPostionSynchronization** : IPunObsorvabl
- class **PhotonPhysicsSynchronization** : IPunObsorbabl
- class **PhotonAnimationSynchronization** : IPunObsorvabl

Rise Event

> Complicated system of easier rising the events.

```
static PhotonEngine.OnRiseEvent.Event += MonoBehaviouPunCallback;
class PhotonView.ViewID;
static PhtonEngipne.RaiseEvent(byte EventCode, Event event,Options options )
```

Network Manager

NetworkReability + check connections

static class PhotonServerSettings

- Serialization

class RoomProperties

```
public int MaxPlayers;
```