How To Use

Adding the TinyBirdNet folder under Assets to your project will give you everything needed to start networking on your game.
Inside the Examples folder you will find a simple working demo of a networked game.

You will need a TinyNetGameManager or derived instance always enabled on your game, so please add one to your first scene and mark it as Don't Destroy on Load. An unique "ConnectKey" is required for your game to be able to connect, preferably include the current version of your application on the key string.

The recommended workflow is to create a class derived from TinyNetPlayerController and implement your player control logic there, for each new player on the game a new TinyNetPlayerController is spawned. You are able to send messages between the client Player Controllers and the server one by using an TinyNetInputMessage.

By creating classes derived from TinyNetBehaviour you are able to enjoy most of the automatic serialization and deserialization, though manual syncing by use of ITinyNetMessage is possible and might be necessary in many applications.

You are able to sync up to 32 Properties per TinyNetBehaviour by using the [TinyNetSyncVar] Attribute. You may at any time bypass this limit by implementing your own TinySerialize and TinyDeserialize Methods.

You are able to spawn and destroy objects in the network by merely calling TinyNetServer.instance.SpawnObject and TinyNetServer.instance.DestroyObject on a server, given a valid GameObject that contains a TinyNetIdentity.

Remember to register all prefabs that have TinyNetIdentity on your TinyNetGameManager, manually or by clicking the Register all TinyNetIdentity prefabs button on your TinyNetGameManager inspector.

*Please refer to the **Script_Reference.pdf** for more articles and an API Guide.*