Photon Classes:

* PhotonNetwork
  + Methods
    - .Instantiate(name, position, rotation)
      * looks through the resources folder for the prefabs unique name and spawns it (over the network?)
      * Returns the gameobject being instantiated
    - .Destroy(GameObject)
      * Destroys the game object in question (over the network?)
    - .IsConnected
      * Returns true if connected to...
* MonoBehaviourPunCallbacks
  + Important Note\*\*
    - Photon has classes like “Player” and “GameManager”, so if/when you make a class that inherits from this one, you should strongly consider wrapping it in a namespace.
  + Summary
    - Class to inherit from if you want to override (or call?) methods like PhotonNetwork.CreateRoom(), PhotonNetwork.JoinRoom(), etc.
* MonoBehaviourPun

Photon Components:

* PhotonView
  + Summary
    - Appears to function similarly to Network ID from UNet
  + Features
    - Observed Components
      * Networked / PUN-related components that this component keeps track of?
  + Properties
    - .IsMine
      * Returns boolean (True if isLocalPlayer)
* Photon Rigidbody View
  + Summary
    - Appears to synchronize physics elements (i.e. velocity, angular velocity, etc.) over the network
* Photon Transform View
  + Summary
    - Appears to synchronize position, rotation, and scale over the network (similar to UNet’s Network Transform component)

Photon Interfaces:

* IPunObservable
  + Summary
    - Appears to allow encoding of information into packets to be sent along the network
  + Methods
    - Public void OnPhotonSerializeView(PhotonStream stream, PhotonMessageInfo info)
      * See usage for writing and reading info to/from the stream here: <https://github.com/ditzel/Mobile-Multiplayer-Action-Game-in-Unity/blob/400d9743b6143cfb84435abd2acaa1387e81b2bd/Underdog%20City/Assets/Scripts/Player/Player.cs>