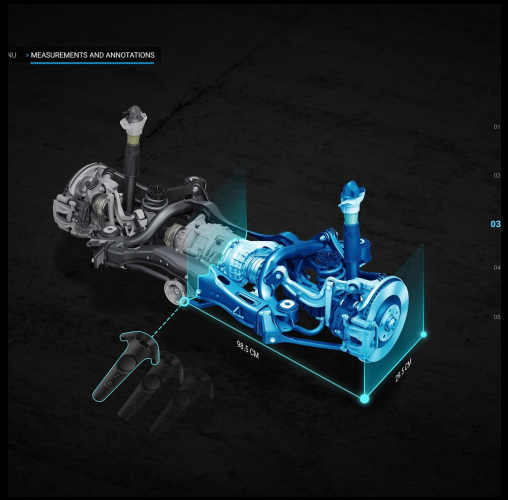
Session 8

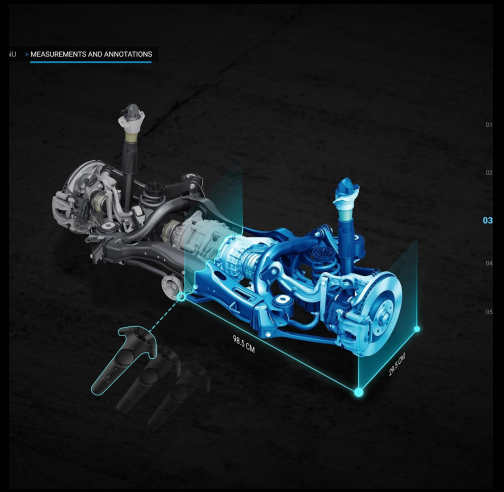
Creating Networked VR Applications

What kind of interactions are possible between two users in VR? Do you foresee any challenges in developing and designing multi-user applications?

Instructor(s)



Instructor Name
Instructor Title
Instructor Company



TA Name
TA Title
TA Company

Session Goals

```
public class NetworkingManagerExample : MonoBehaviourPunCallbacks
{
    [SerializeField] GameObject XRRigPrefab;
    [SerializeField] Vector3 XRRigSpawnPosition;

    void Start()
    {
        PhotonNetwork.ConnectUsingSettings();
    }

    public override void OnConnectedToMaster()
    {
            PhotonNetwork.JoinRoom("defaultRoom");
    }

    public override void OnJoinedRoom()
    {
        Debug.Log("joined a room!");
        PhotonNetwork.Instantiate(XRRigPrefab.name, XRRigSpawnPosition, Quaternion.identity);
    }
```

In this session, you'll:

- Set up shared experiences in Virtual Reality Applications

- Identify the key features of Photon Networking that make it ideal for multi-user experiences in Unity

Session outline

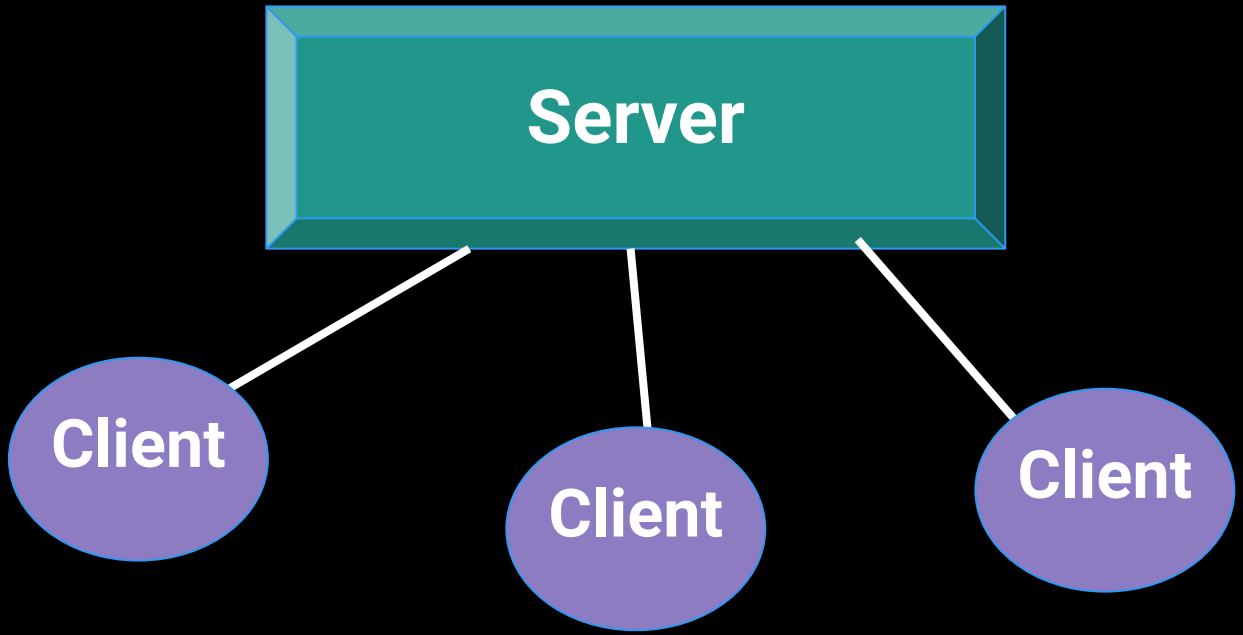
1. Overview of Photon Unity Networking
    - Establish a basic network connection in Unity

1. Syncing GameObjects across a network
    - Add Photon Views to interactable objects

1. Sharing XR Rigs Across Networks
    - Create and Instantiate a Multi-user XR Rig Prefab

Activity 8: Test networking with a partner

Principles of Networking

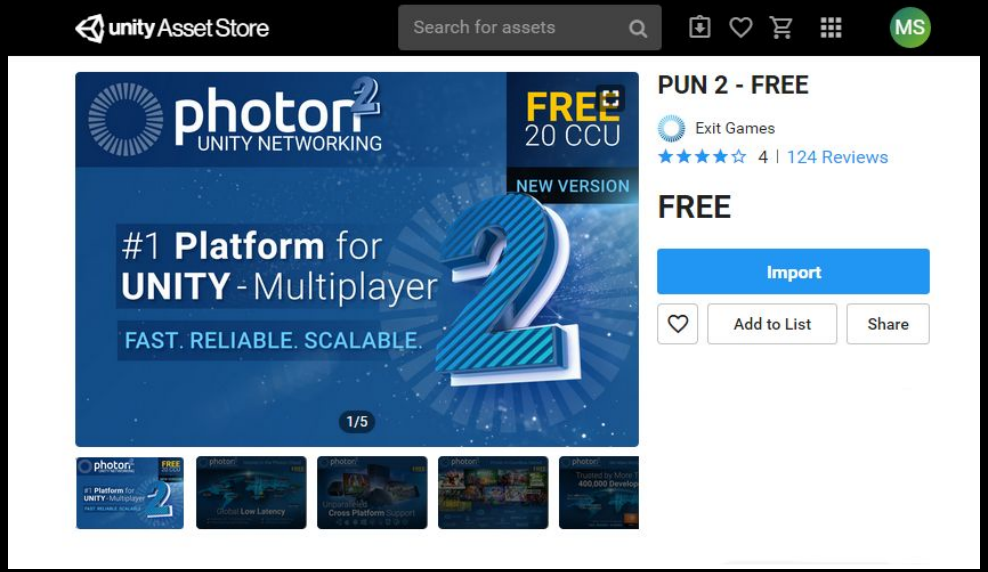


Photon Unity Networking

Most notable features:

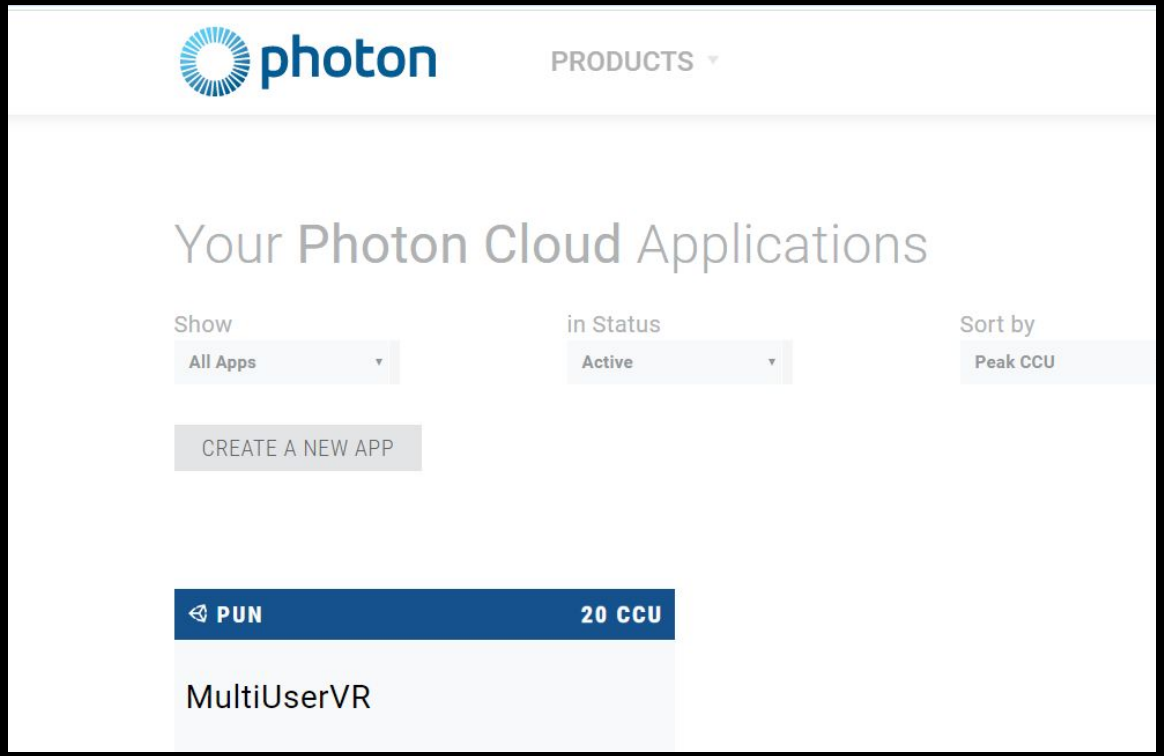
- Dead-easy API
- Lots of demos and an extensive PUN Basics Tutorial
- Server available as hosted service (free for development) or as "On Premise"
- Load-balanced! Scales across servers (with no extra effort)
- Outstanding performance of the Photon Server
- Dedicated servers. No NAT punch-through needed
- Offline mode: re-use your multiplayer code in singleplayer game modes

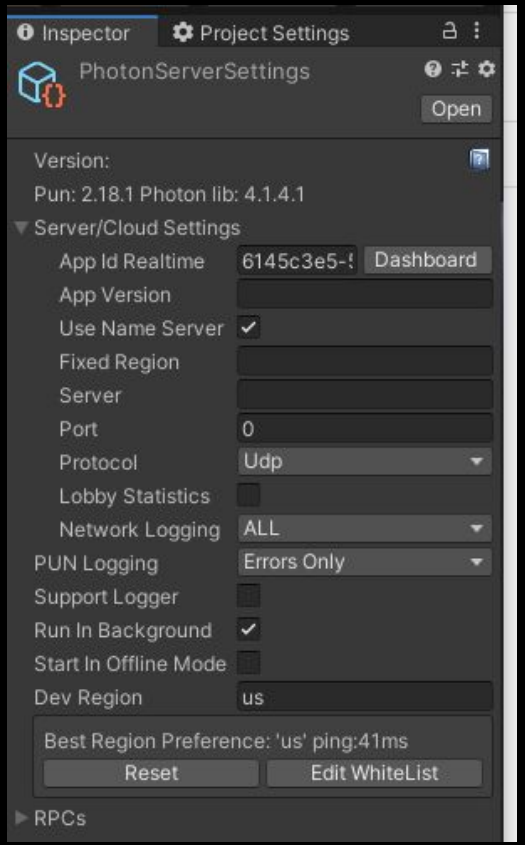
Initializing Photon in Unity



unity

Initializing Photon in Unity





unity

Initializing Photon in Unity

PhotonNetwork.ConnectUsingSettings

PhotonNetwork.JoinRoom

PhotonNetwork.CreateRoom

OnConnectedToMaster

OnJoinRoom

OnJoinRoomFailed

```
using UnityEngine;
using Photon.Pun;
using Photon.Realtime;


public class NetworkingManagerExample : MonoBehaviourPunCallbacks
{
    void Start()
    {
        PhotonNetwork.ConnectUsingSettings();
    }

    public override void OnConnectedToMaster()
    {
            PhotonNetwork.JoinRoom("defaultRoom");
    }

    public override void OnJoinedRoom()
    {
        Debug.Log("joined a room!");
    }

    public override void OnJoinRoomFailed(short returnCode, string message)
    {
        PhotonNetwork.CreateRoom("defaultRoom", new RoomOptions { MaxPlayers = 4 });
    }
```

Establish a Basic Network Connection

1. Use the Photon Cloud to create and assign an AppID
2. Use the PUN Wizard to setup a networked application
3. Create a new scene for your networked Application
4. Create a basic networked connection

```
public override void OnConnectedToMaster()
{
        PhotonNetwork.JoinRoom("defaultRoom");
}

public override void OnJoinedRoom()
{
    Debug.Log("joined a room!");
}
```

unity

Establish a Basic Network Connection

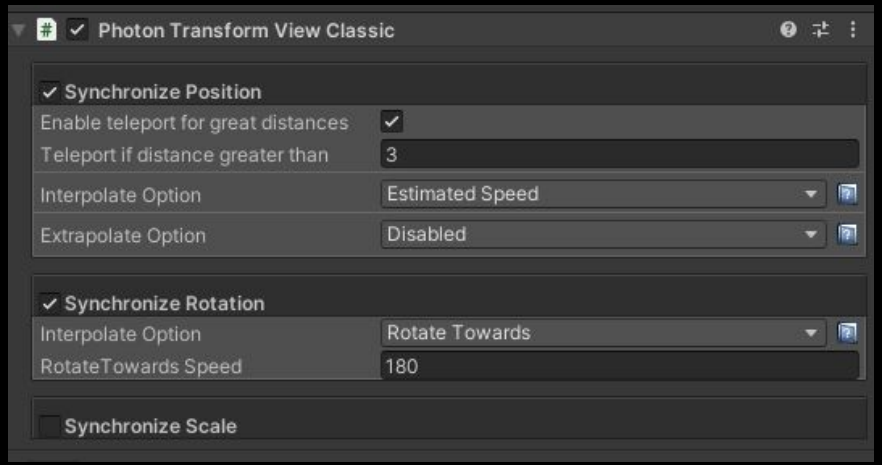
1. Use the Photon Cloud to create and assign an AppID
2. Use the PUN Wizard to setup a networked application
3. Create a new scene for your networked Application
4. Create a basic networked connection

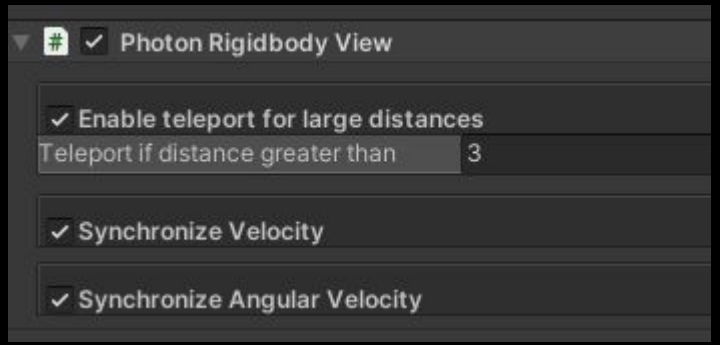
```
public override void OnConnectedToMaster()
{
        PhotonNetwork.JoinRoom("defaultRoom");
}

public override void OnJoinedRoom()
{
    Debug.Log("joined a room!");
}
```

unity

Syncing GameObjects Across the Network

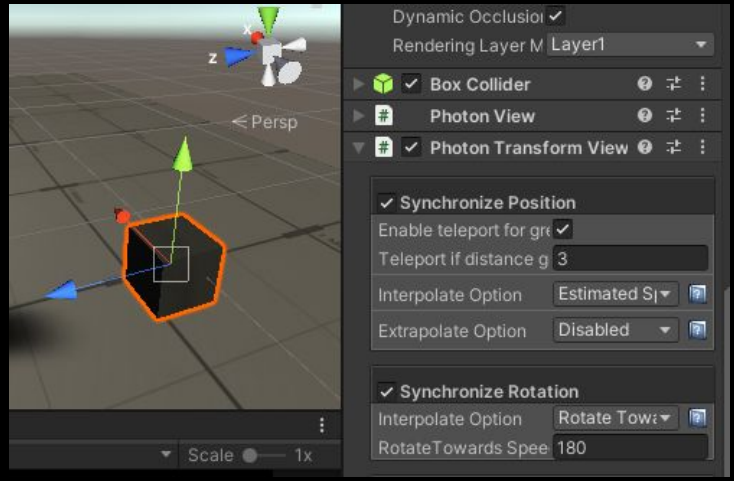




unity

Syncing GameObjects with Photon

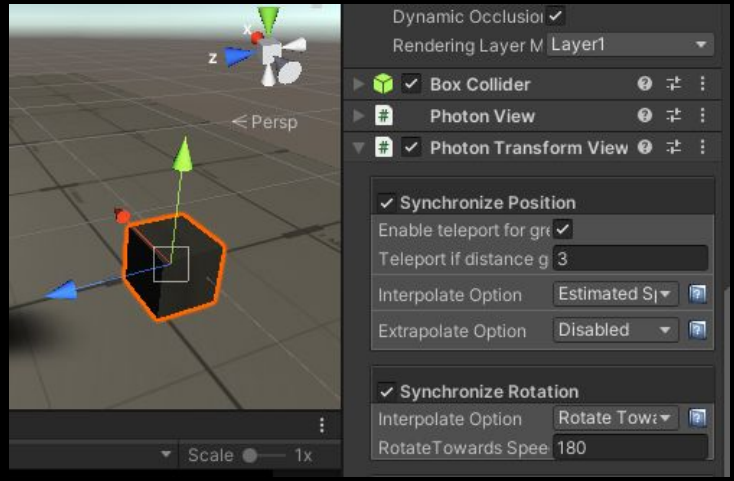
1. Create a few primitive interactable components that use Photon Transform and Rigidbody Views
2. Test your Photon Views by Building your game and running alongside in the editor



You may need to unplug your VR Headset in order to have two instances of the application running

Syncing GameObjects with Photon

1. Create a few primitive interactable components that use Photon Transform and Rigidbody Views
2. Test your Photon Views by Building your game and running alongside in the editor



You may need to unplug your VR Headset in order to have two instances of the application running

Sharing XR Rigs Across Networks


```
public override void OnJoinedRoom()
{
    Debug.Log("joined a room!");
    PhotonNetwork.Instantiate(XRRigPrefab.name, XRRigSpawnPosition, Quaternion.identity);
}
```

Instantiating Multi-user XR Rig

1. Create a Multiuser-XRRig prefab and store it in a folder named Resources
2. Instantiate an XR Rig into your Multi-user scene with PhotonNetwork.Instantiate

```
public override void OnJoinedRoom()
{
    Debug.Log("joined a room!");
    PhotonNetwork.Instantiate(XRRigPrefab.name, XRRigSpawnPosition, Quaternion.identity);
}
```

Instantiating Multi-user XR Rig

1. Create a Multiuser-XRRig prefab and store it in a folder named Resources
2. Instantiate an XR Rig into your Multi-user scene with PhotonNetwork.Instantiate

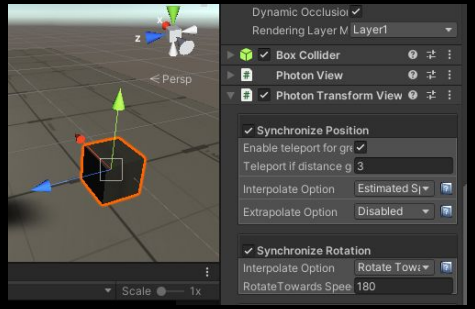
```
public override void OnJoinedRoom()
{
    Debug.Log("joined a room!");
    PhotonNetwork.Instantiate(XRRigPrefab.name, XRRigSpawnPosition, Quaternion.identity);
}
```

Activity Session 8

Testing your Networked Application

unity

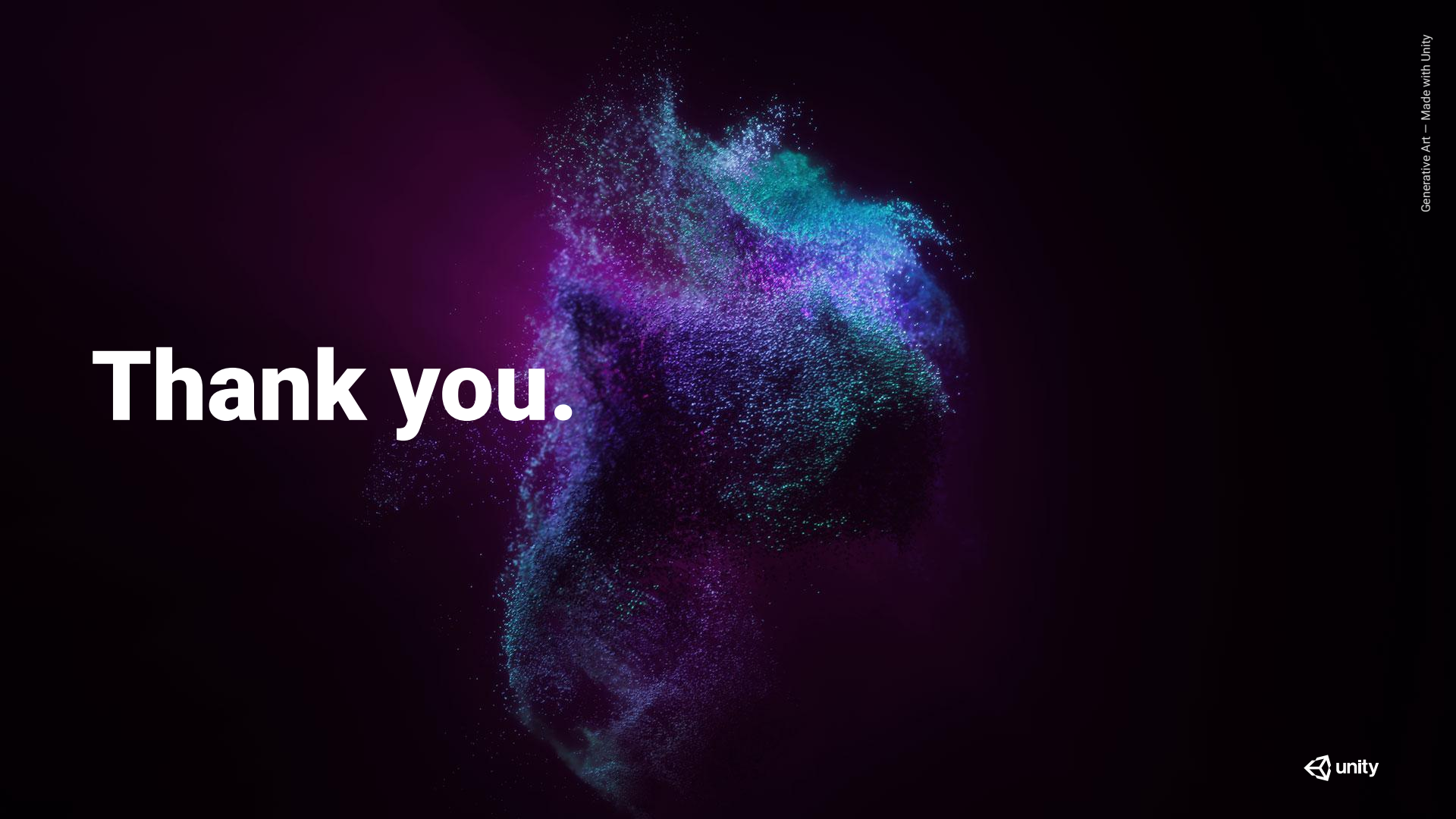
Activity 8 Goals



1. **Build your project and test your networking solution with a partner**
    a. Take note of any issues you encounter and what you think is causing the issues with the XR Rig and interactions in your project (There will be issues!)

1. **Apply Photon Views to all objects in your scene that need to be shared across the network.**
    b. Test the Photon View objects by moving them in the scene view, or with your XR Rig

Feel free to ask questions!


Thank you.
Generative Art — Made with Unity
unity