Unity Bootcamp

UCF CAP6121
Spring 2023
Mykola Maslych, Ryan Ghamandi
(some slides borrowed from Dr. Kevin Pfeil)

Lectures Coverage

- Mon, Jan 23
  - Unity Setup with XR Interaction Toolkit (XRIT)
  - Scripting
  - Colliders and Triggers
- Wed, Jan 25
  - Advanced examples with Oculus VR (OVR)

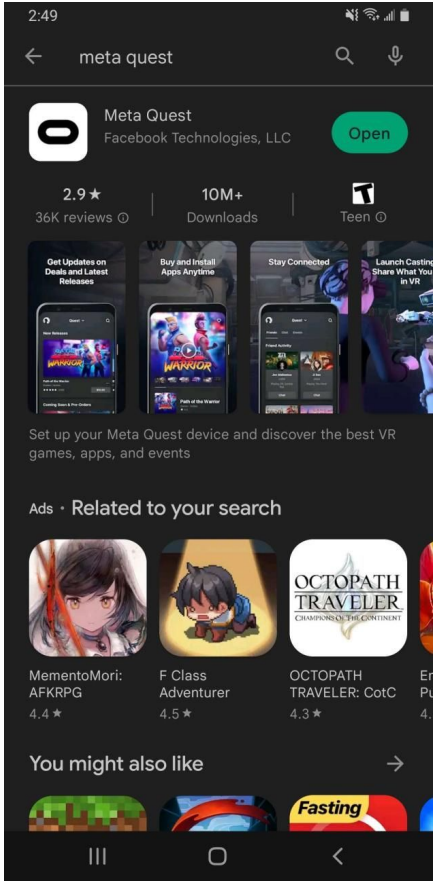
Outline

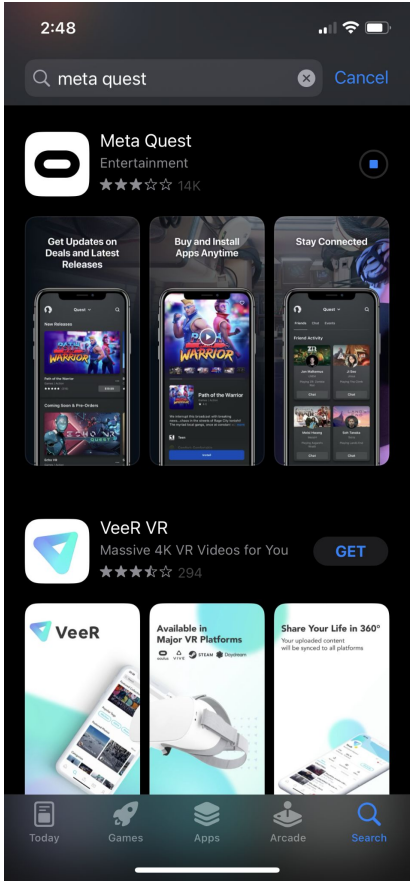
1. (Optional) Pair Meta Quest phone app with Quest 2
2. Install Oculus on PC
3. Install Unity
4. Download Unity Assets
5. Understanding Unity
6. Live Demo

(Optional) Pair Meta Quest App with Quest Device

This step is required if you have not done the initial setup of your Quest 2

1. The app is called "Meta Quest"
2. Follow instructions to pair the headset with the phone

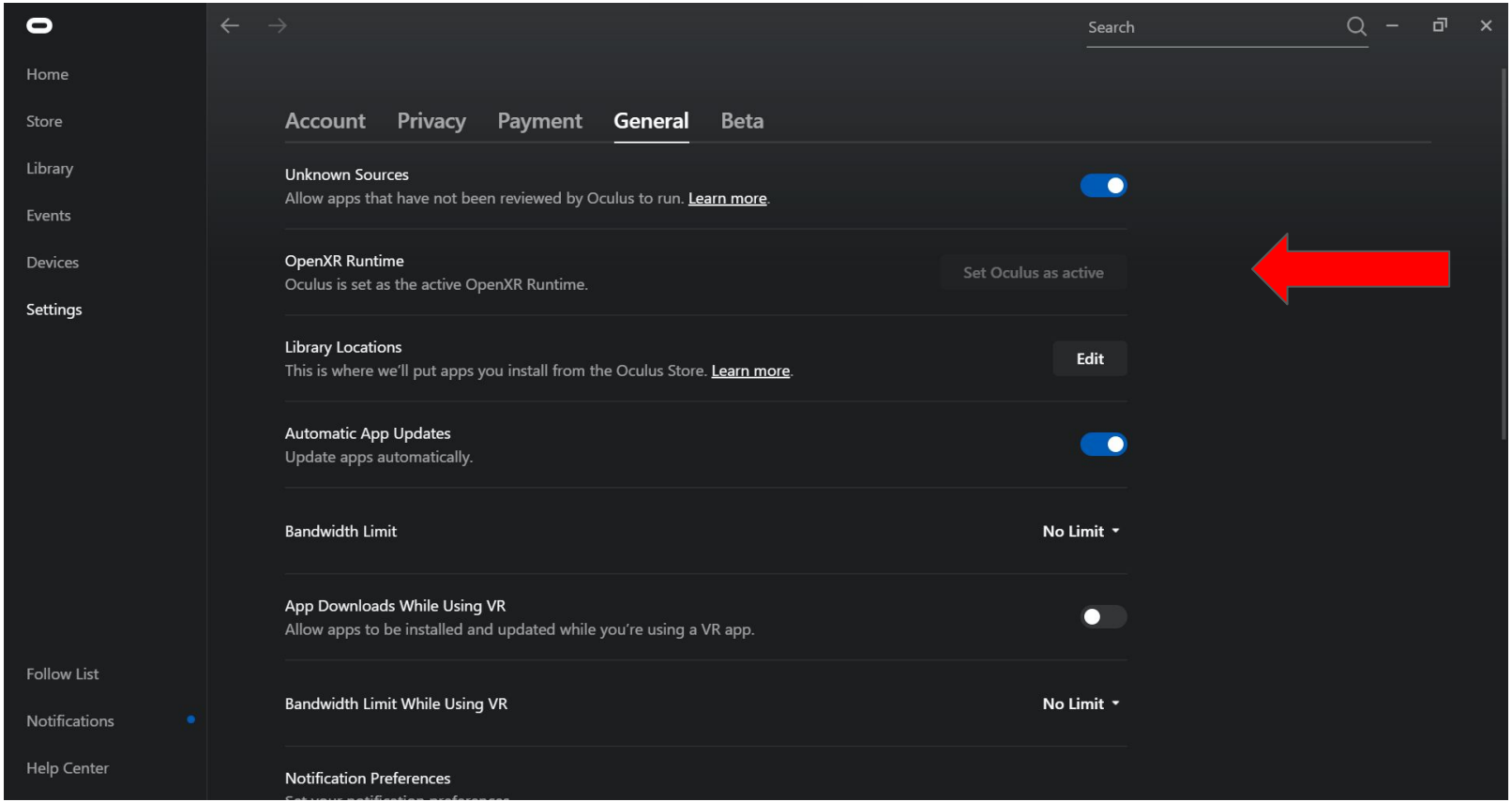




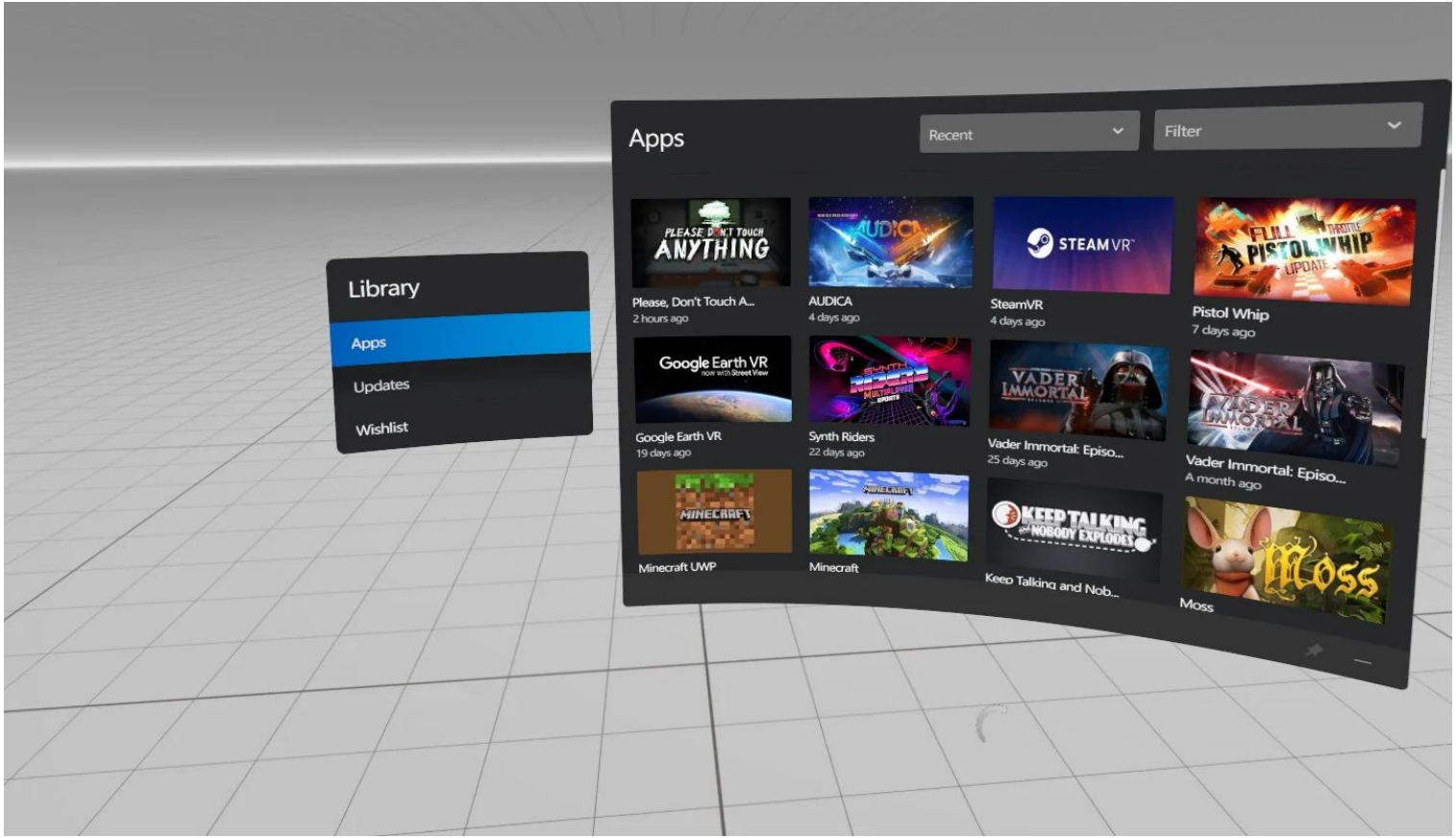
Install Oculus Software on your PC

Download from: https://www.meta.com/quest/setup/
Install **OculusSetup.exe** and login using a Meta account
Connect Quest 2 using a Link Cable

Install Oculus Software on your PC

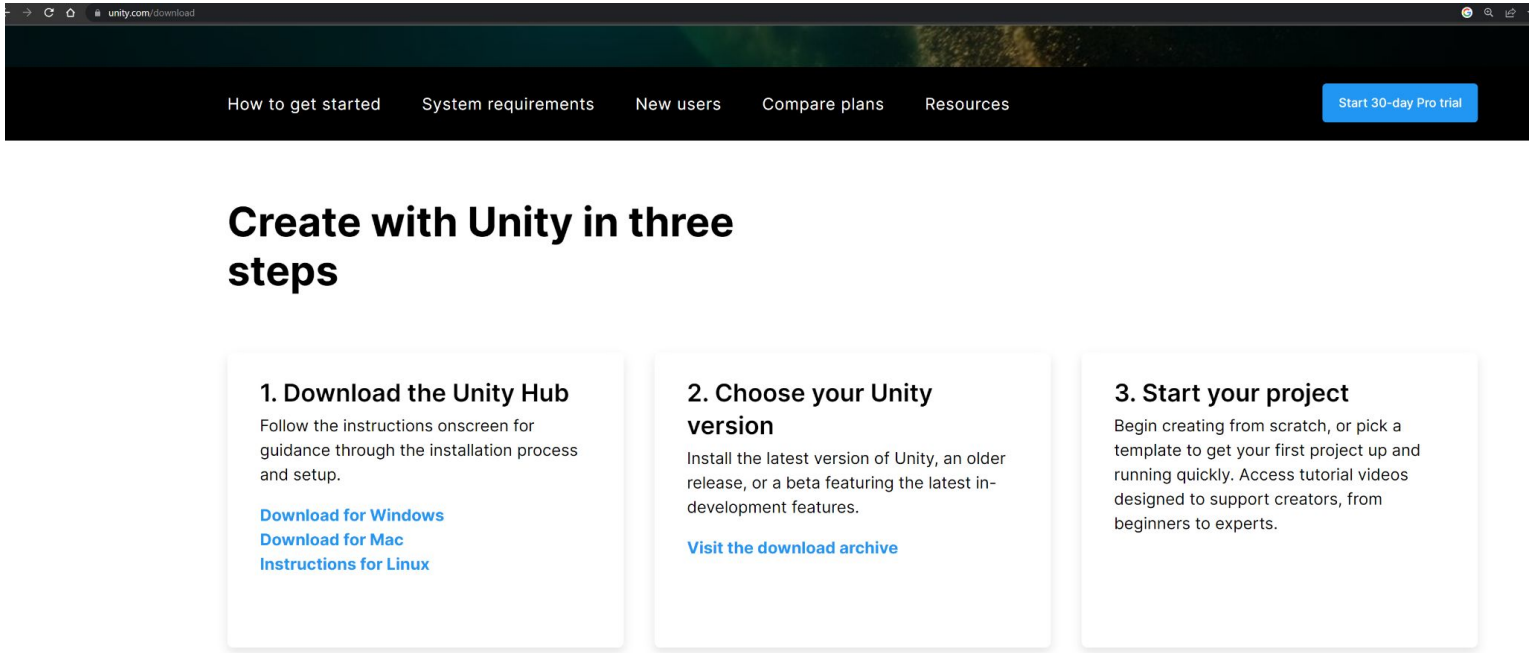


How Oculus Link Menu Looks like



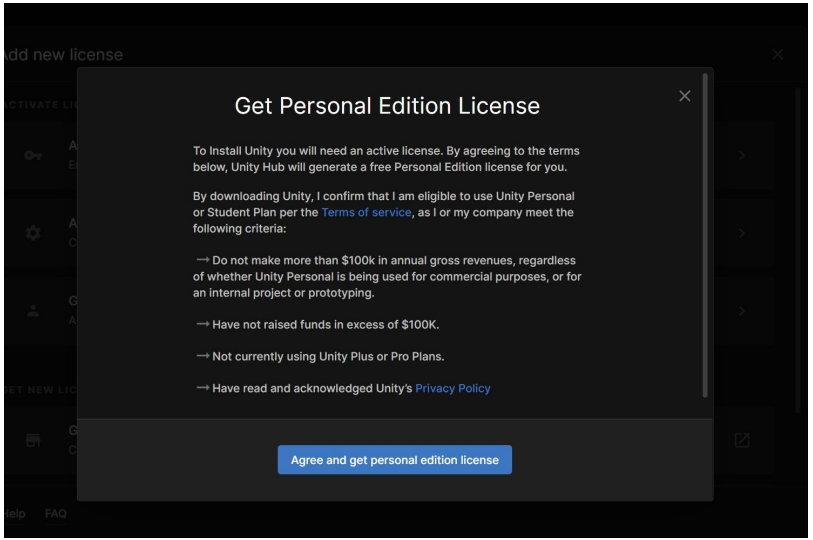
Download Unity Hub

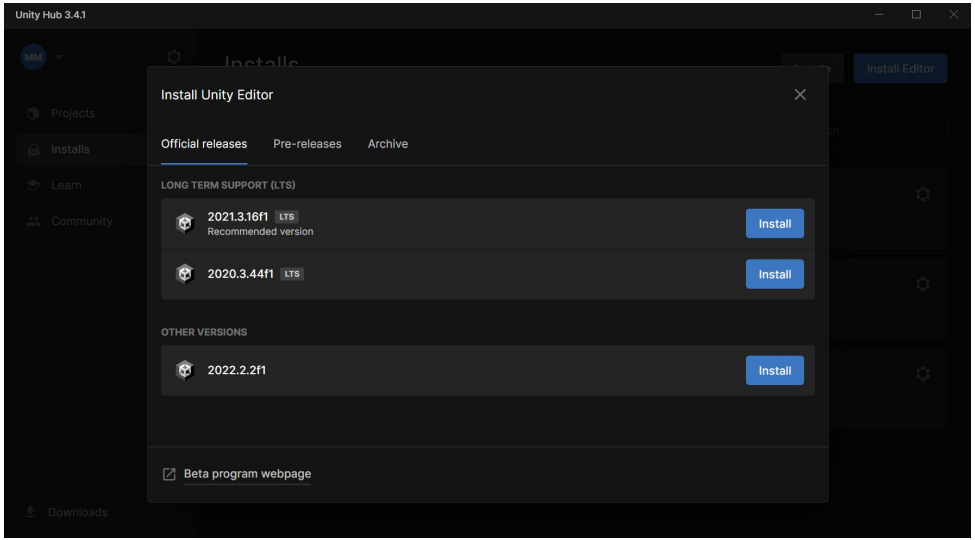
https://unity.com/download



Download Unity Hub

- Activate Personal License
- Download Latest **Long Term Support** version of unity



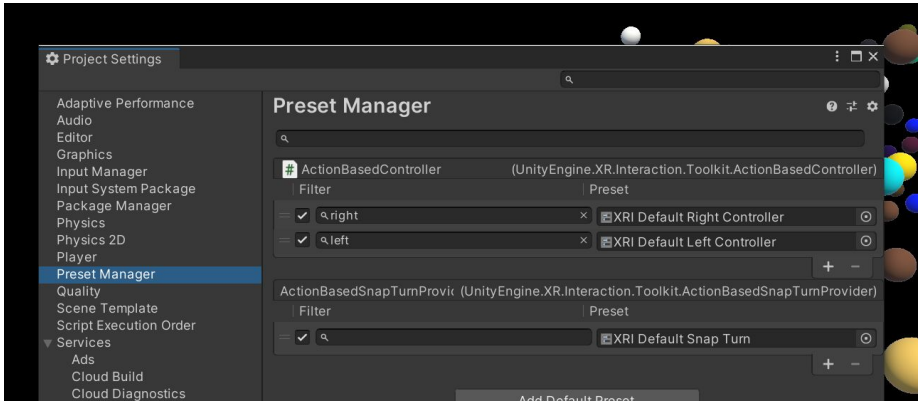


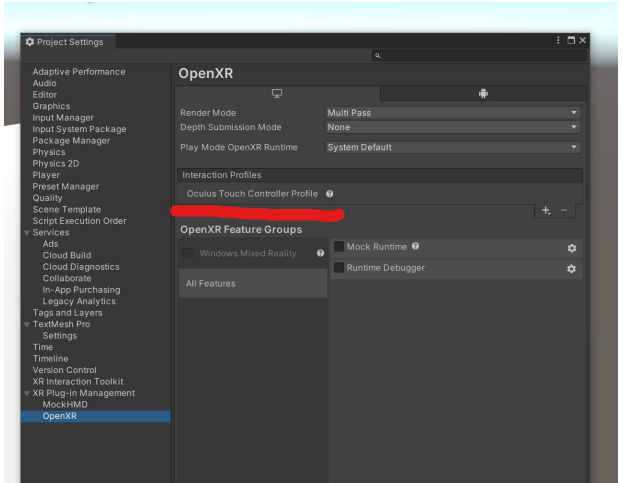
XRIT Setup

- Create a new empty 3D Project
- Then, in: Window -> Package Manager
  - XR Plugin Management (https://docs.unity3d.com/Packages/com.unity.xr.openxr@1.6/manual/index.html)
  - XR Interaction Toolkit https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit@2.3/)
    - Click "Yes" when prompted about new Input System
    - Also install "Starter Assets" from XR Interaction Toolkit
    - Add all Action Set components to the Action Set

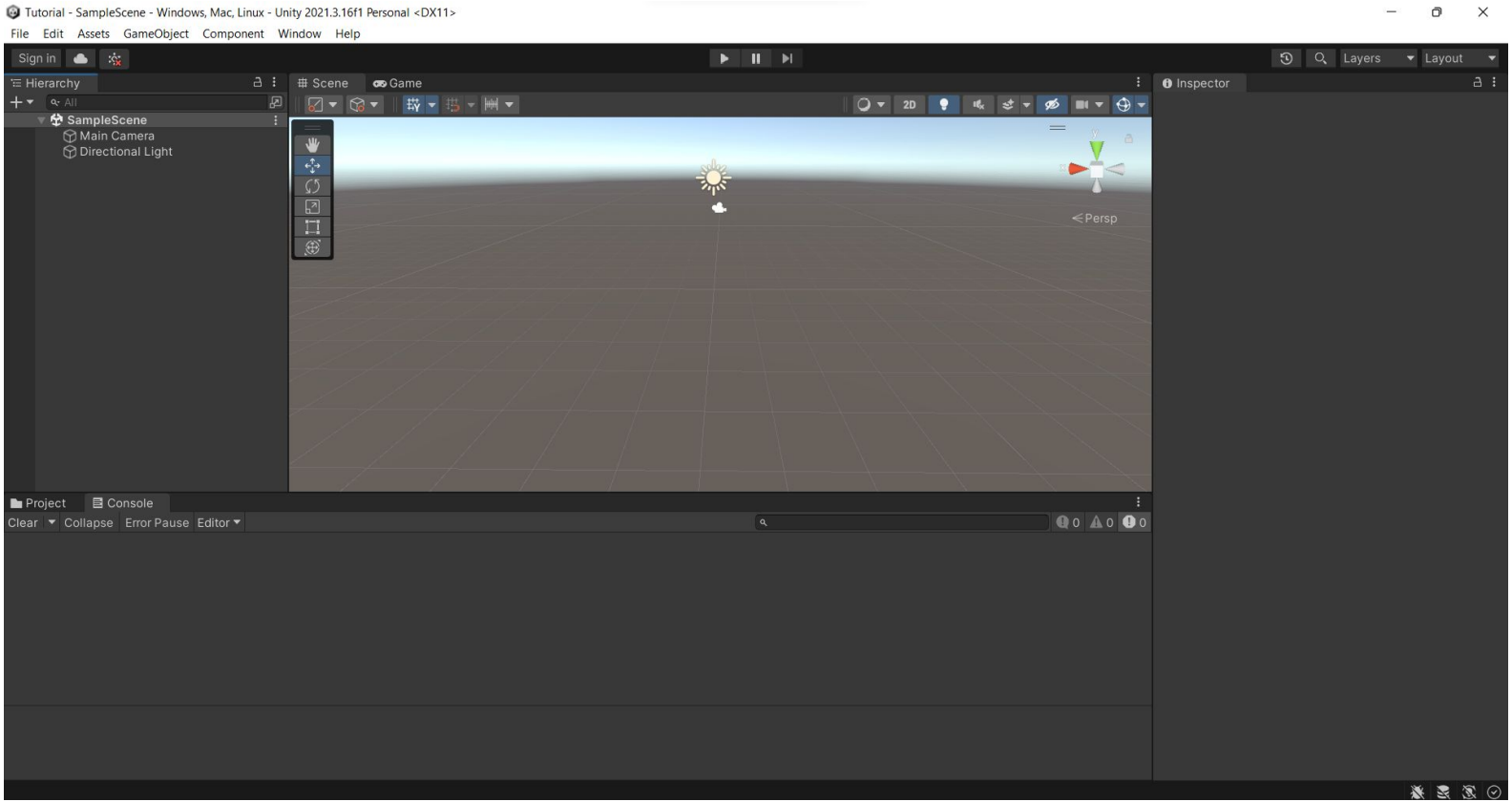
Assets for OpenXR project

- Edit -> Project Settings -> XR Plug-in Management -> Check OpenXR
- Edit -> Project Settings -> Preset Manager -> add "right", "left" in ActionBasedController
- Edit -> Project Settings -> XR Plug-in Management -> add "Oculus Touch Controller Profile" to interaction Profiles

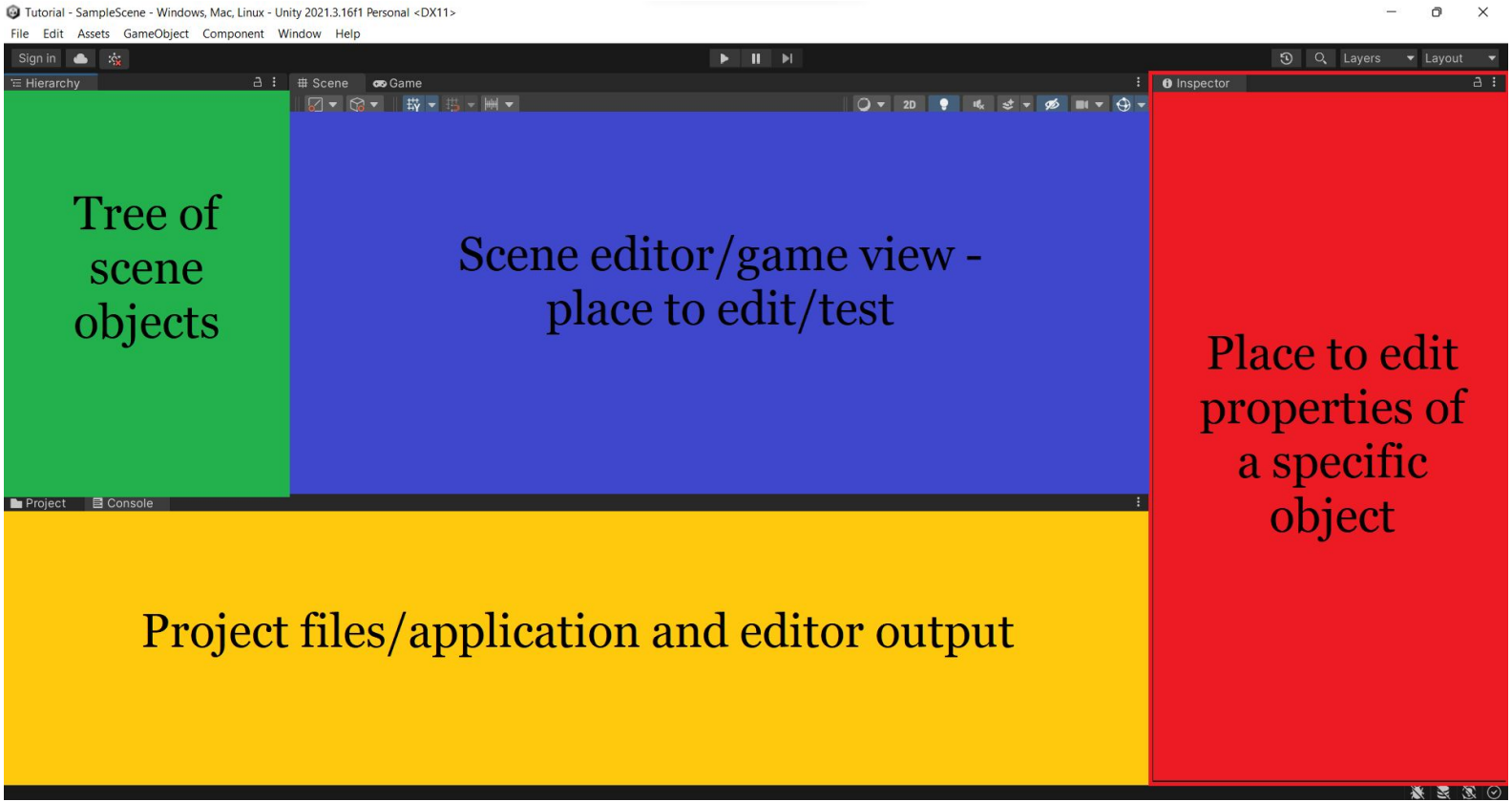




Unity Intro


Tutorial - SampleScene - Windows, Mac, Linux - Unity 2021.3.16f1 Personal <DX11>
File Edit Assets GameObject Component Window Help
Sign in
Layers
Layout
Hierarchy
SampleScene
Main Camera
Directional Light
Scene
Game
2D
Persp
Inspector
Project
Console
Clear
Collapse
Error Pause
Editor

Unity Intro


Tutorial - SampleScene - Windows, Mac, Linux - Unity 2021.3.16f1 Personal <DX11>
File Edit Assets GameObject Component Window Help
Sign in
Layers
Layout
Hierarchy
Scene
Game
2D
Inspector
Tree of scene objects
Scene editor/game view - place to edit/test
Place to edit properties of a specific object
Project
Console
Project files/application and editor output

Understanding Unity

Game Object Hierarchy

- As you place new things into the world, they are listed in the hierarchy
  - Camera(s)
  - Scenery Objects
  - Player Avatar
  - Etc.
- **Click** on an object to see/edit details in **Inspector**
- **Double** click on object to zoom to it in the **Scene**

Understanding Unity

- Scene Viewer
  - Displays objects in the game world
  - Multiple ways to view your world
  - Game Objects can be manipulated directly
- Game Viewer
  - Displays objects according to the game camera
  - Consider it as a way to “preview” the world without running the whole game

Understanding Unity

- Inspector
  - For selected Game Object, displays components
  - Components take many forms that affect how the game object looks/behaves
    - Scripts, Meshes, Colliders, Audio Sources...
  - Most components can be modified in Inspector
    - Object “Transform” (position, rotation, scale)
    - Mesh properties
    - Script variables
    - Etc.

Understanding Unity

- Console
  - Displays errors, warnings, debugging lines, etc.

- Project Assets
  - Quick way to get to prefabs, scripts, and other resources that are part of the project

Live Demo

- Simple Environment with
  - Ground Plane
  - A couple of boxes
  - XR Origin and navigation

Set up simple XR Origin

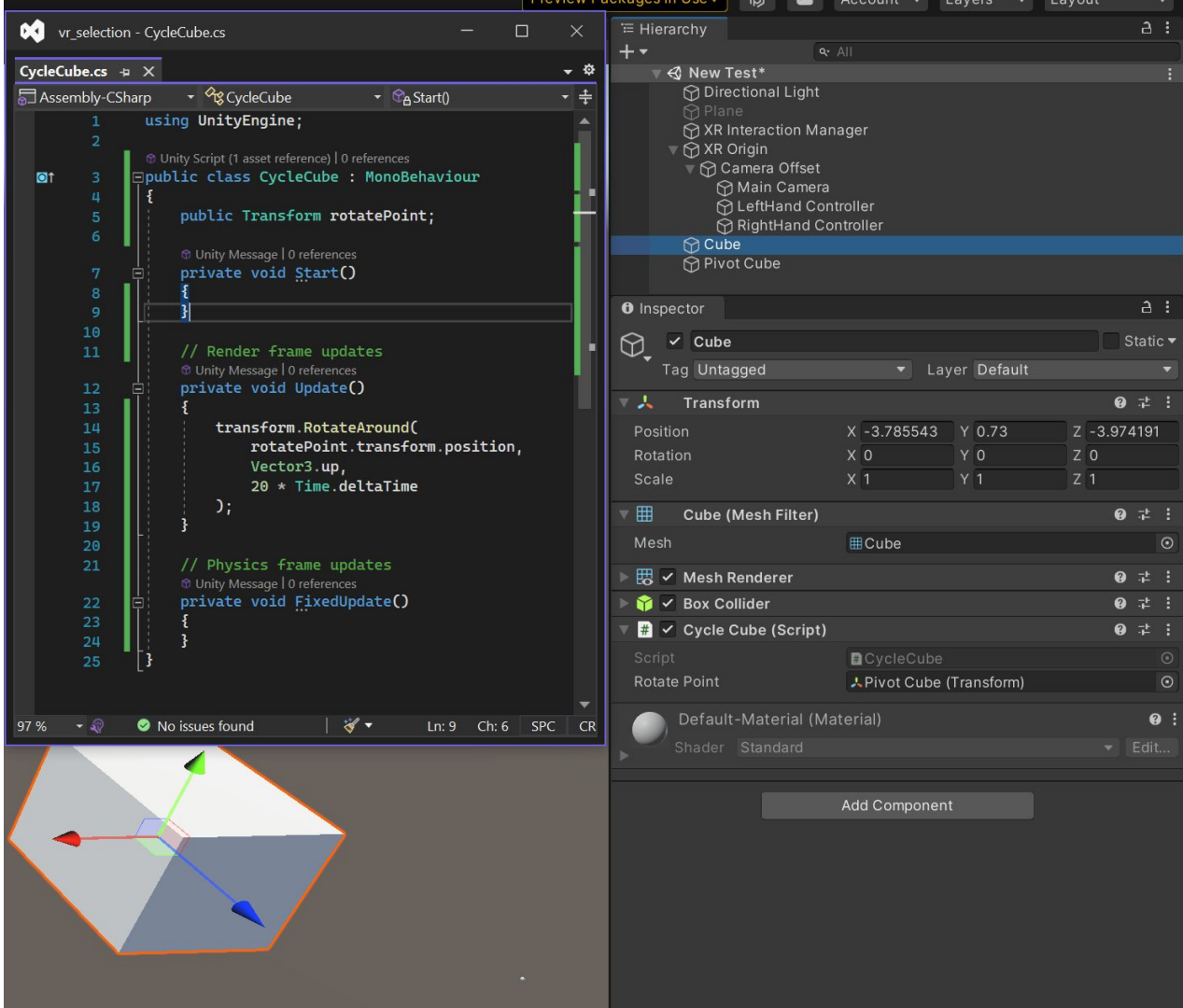
- XR Origin
  - Delete the regular "Main Camera" in the hierarchy
  - Right-click in hierarchy -> XR -> XR Origin (Action-based)
- Locomotion
  - Right click on the XR Origin object in hierarchy -> XR -> Locomotion System (Action-based)
  - Create a plane
  - Click on the plane and "Add Component" with the name "Teleportation Area"
  - You can now teleport by pointing to the plane and hitting the "grab button" (side trigger button)

Scripting

- In addition to physics-based behavior, for custom behavior we use custom scripts
- Create a new Script
- Add Script to game object
- Update Loop https://docs.unity3d.com/Manual/ExecutionOrder.html

Scripting

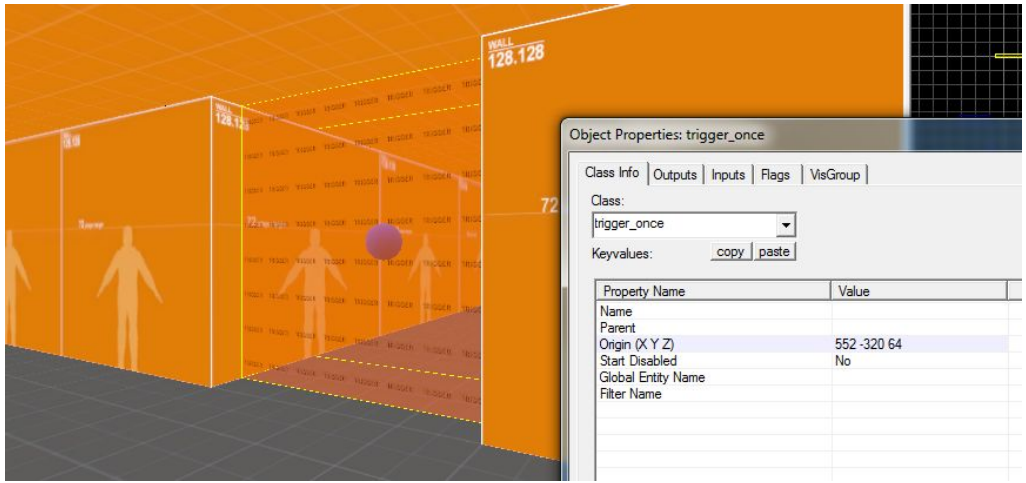
- RotateAround https://docs.unity3d.com/ScriptReference/Transform.RotateAround.html



Colliders and Triggers

Colliders and Triggers

- Colliders are generally for physics
- Triggers are for activating custom behavior
- Triggers also rely on colliders



source: LambdaGeneration

Colliders and Triggers

- Collision Matrix in Unity
- Unity Example:
- https://docs.unity3d.com/ScriptReference/Collider.OnTriggerEnter.html
- Triggers “trigger” specified behavior
- For triggers to work:
  1. Both objects must have colliders
  2. One must have "IsTrigger" and must have a RigidBody
- Let's convert the cubes into destroyable enemies!

Colliders and Triggers

- Check for triggers of different objects with
  - Object names (other.name == "EnemyCube")
  - Object tags (other.tag == "Enemy")
  - Presence of a certain component (TryGetComponent) (arguably the fastest one)

Spawning enemies - Object Instantiation

- Documentation: https://docs.unity3d.com/ScriptReference/Object.Instantiate.html
- In the class demo we made a simple script that spawns an enemy randomly every single frame. In your assignments you will need to be smarter about it :)
- Make enemy spawning depend on conditions. For example, every N seconds, or when a single enemy was killed etc. You may need to do this asynchronously, and you could use Coroutines and IEnumerator.

More useful resources

- **RayCasting** helps to get an object that you aim at, and the exact hit location. RayCasting documentation example.
- **Force** can be used to make objects "jump", and for shooting projectiles. Add force documentation example.
- XRIT Setup full guide.
- **Object Pooling** helps reduce compute load. Documentation example.
- Object **Instantiation** documentation example.
- Which objects will collide with each other? Depends on the Layers. Collision matrix documentation.

Questions