ATLAS Showcase Presentation

Leo Lee

VR Team

About me

Name	**Leo Lee**
Year / Major	**Sophomore / MATH & CS** (& Minor in Physics)
Interests	• **Solving math problems & coding challenges** • **Sports:** basketball, cycling, hiking, etc. • **Music:** J-pop, Western, hip hop, soundtracks, etc. • ~~Appreciating Internet memes~~
Fun fact	Without being a huge anime fan, I enjoy listening to J-pop! *Racing Into the Night* by YOASOBI, definitely my favorite song →

“Rage Room” Video Demo

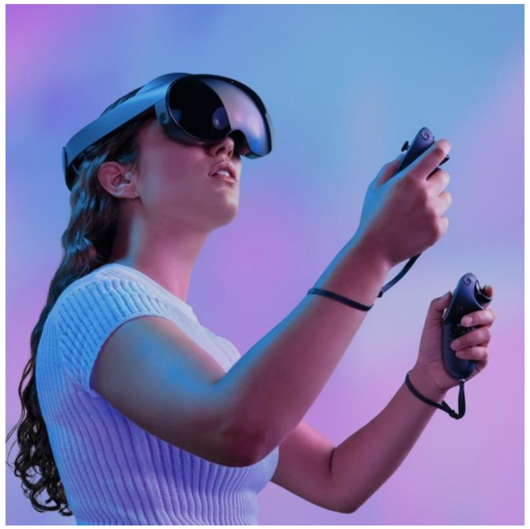
- Player moves continuously and turns left or right instantaneously by 45 degrees
- A ball spawns on player’s hand when VR controller’s “select trigger” is pressed down and is released when the trigger is released
- Ball is thrown out when detached
- Objects like block towers or lamp will be destroyed when a ball hits them with speed greater than a certain amount

Video Unavailable

Internship Goals

Proficiency in…

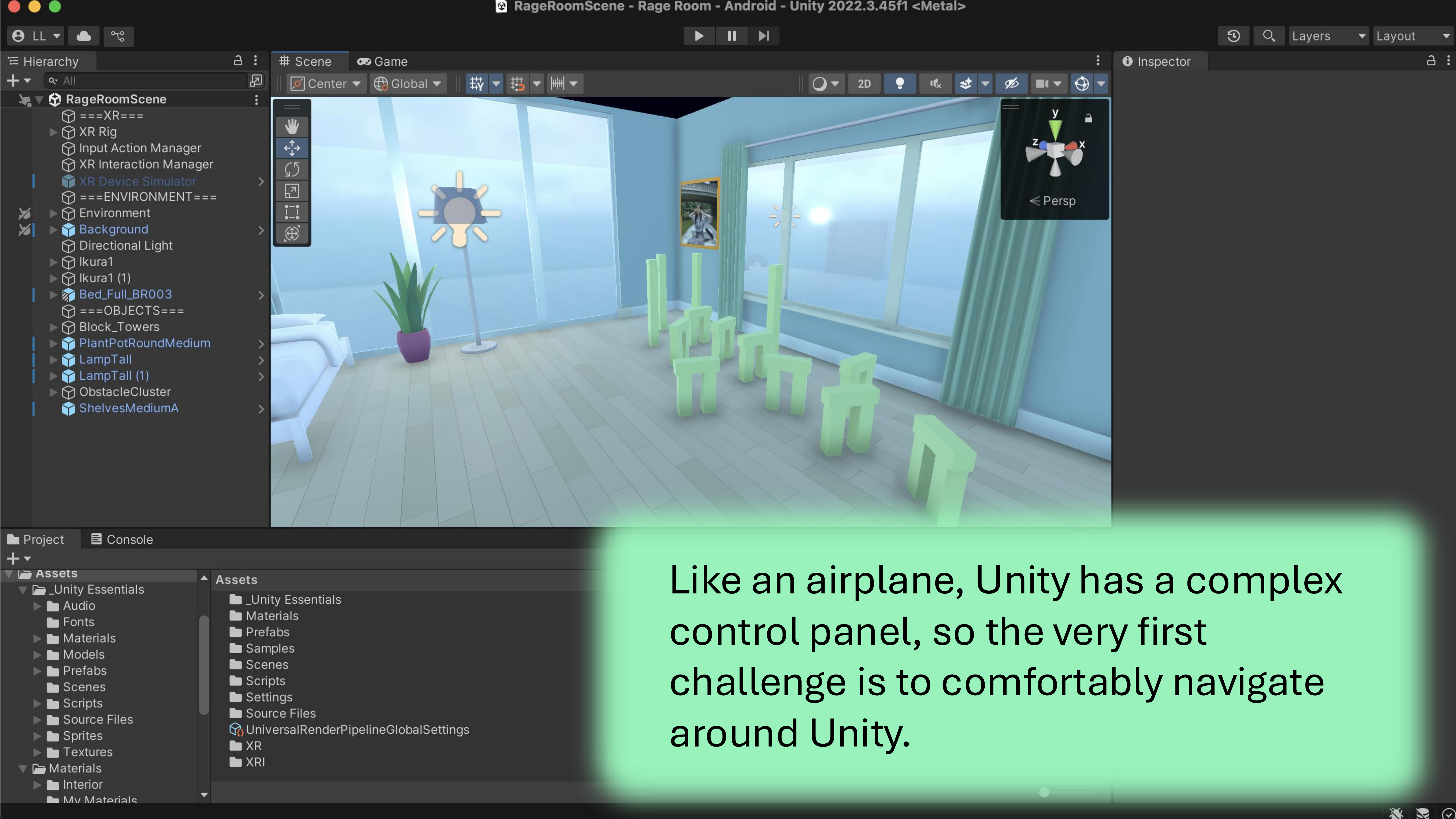
1. **Unity for VR Game Development**
2. **Programming in C# for Unity**



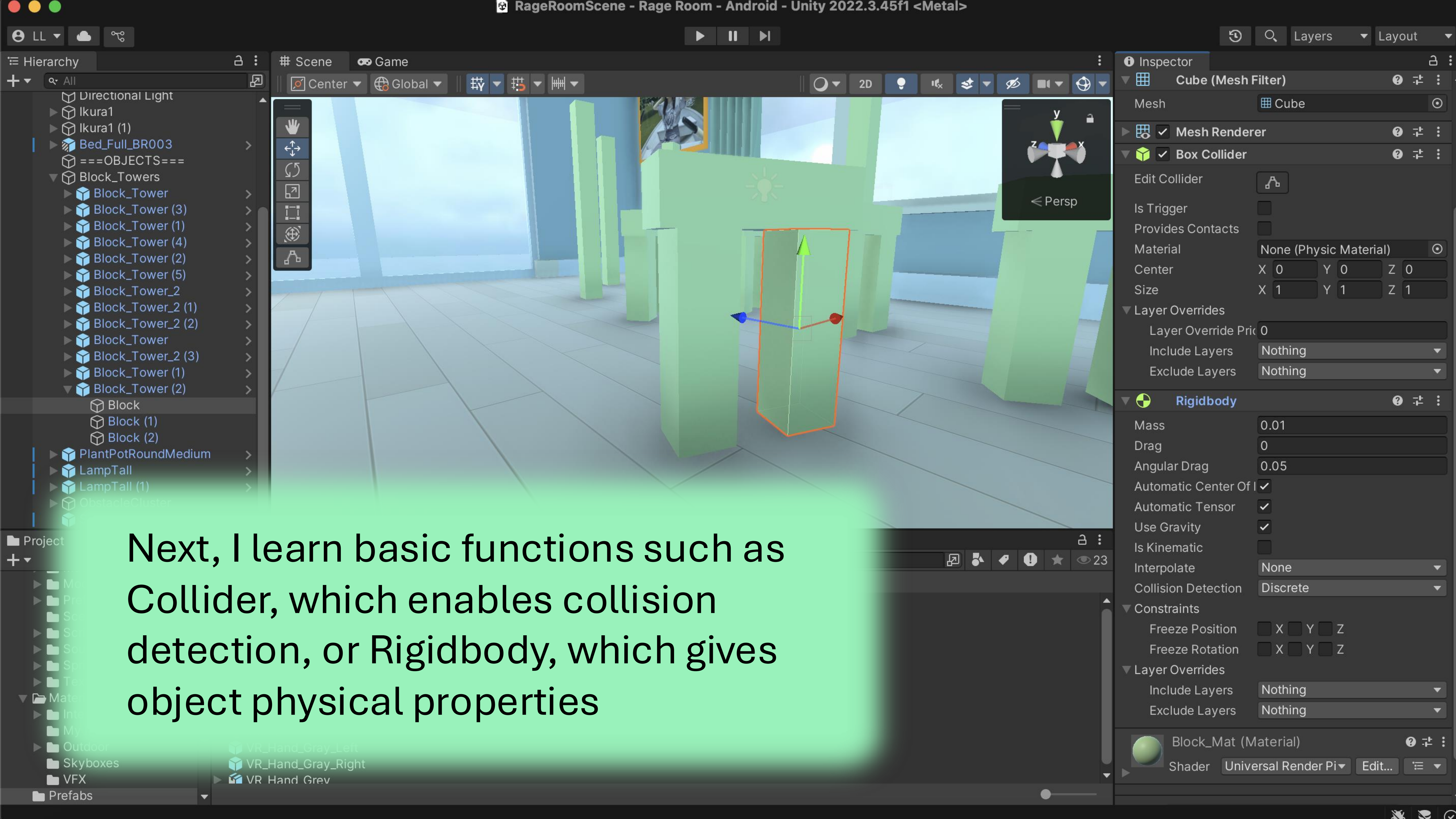
Goal 1 – Unity for VR Game Dev

- **Learning pathway**

Easy	▪ Unity UI & essential functions ▪ Unity VR project setup
Medium	▪ Core built-in VR actions
Hard	▪ Ability to integrate the built-in features of the XR Interaction Toolkit into custom scripts



Like an airplane, Unity has a complex control panel, so the very first challenge is to comfortably navigate around Unity.



Next, I learn basic functions such as Collider, which enables collision detection, or Rigidbody, which gives object physical properties

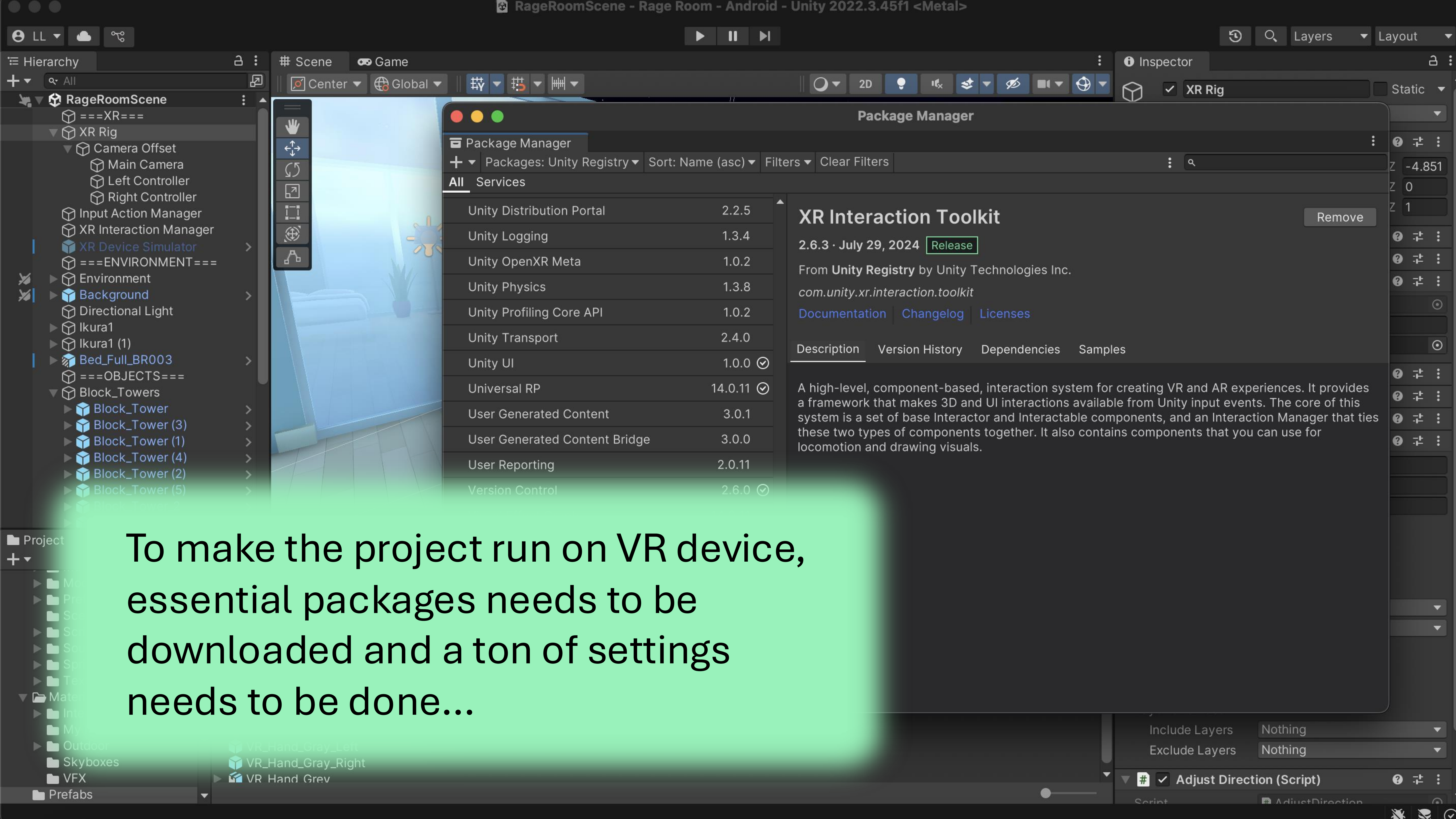
Cube (Mesh Filter)
Mesh: Cube

Mesh Renderer ✓

Box Collider ✓
Edit Collider
Is Trigger: ☐
Provides Contacts: ☐
Material: None (Physic Material)
Center: X 0 Y 0 Z 0
Size: X 1 Y 1 Z 1
Layer Overrides
Layer Override Pri 0
Include Layers: Nothing
Exclude Layers: Nothing

Rigidbody
Mass: 0.01
Drag: 0
Angular Drag: 0.05
Automatic Center Of I ✓
Automatic Tensor ✓
Use Gravity ✓
Is Kinematic ☐
Interpolate: None
Collision Detection: Discrete
Constraints
Freeze Position: ☐ X ☐ Y ☐ Z
Freeze Rotation: ☐ X ☐ Y ☐ Z
Layer Overrides
Include Layers: Nothing
Exclude Layers: Nothing

Block_Mat (Material)
Shader: Universal Render Pi Edit...



To make the project run on VR device, essential packages needs to be downloaded and a ton of settings needs to be done...

Unity Distribution Portal	2.2.5
Unity Logging	1.3.4
Unity OpenXR Meta	1.0.2
Unity Physics	1.3.8
Unity Profiling Core API	1.0.2
Unity Transport	2.4.0
Unity UI	1.0.0 ✓
Universal RP	14.0.11 ✓
User Generated Content	3.0.1
User Generated Content Bridge	3.0.0
User Reporting	2.0.11
Version Control	2.6.0 ✓

XR Interaction Toolkit

2.6.3 · July 29, 2024 Release

From **Unity Registry** by Unity Technologies Inc.

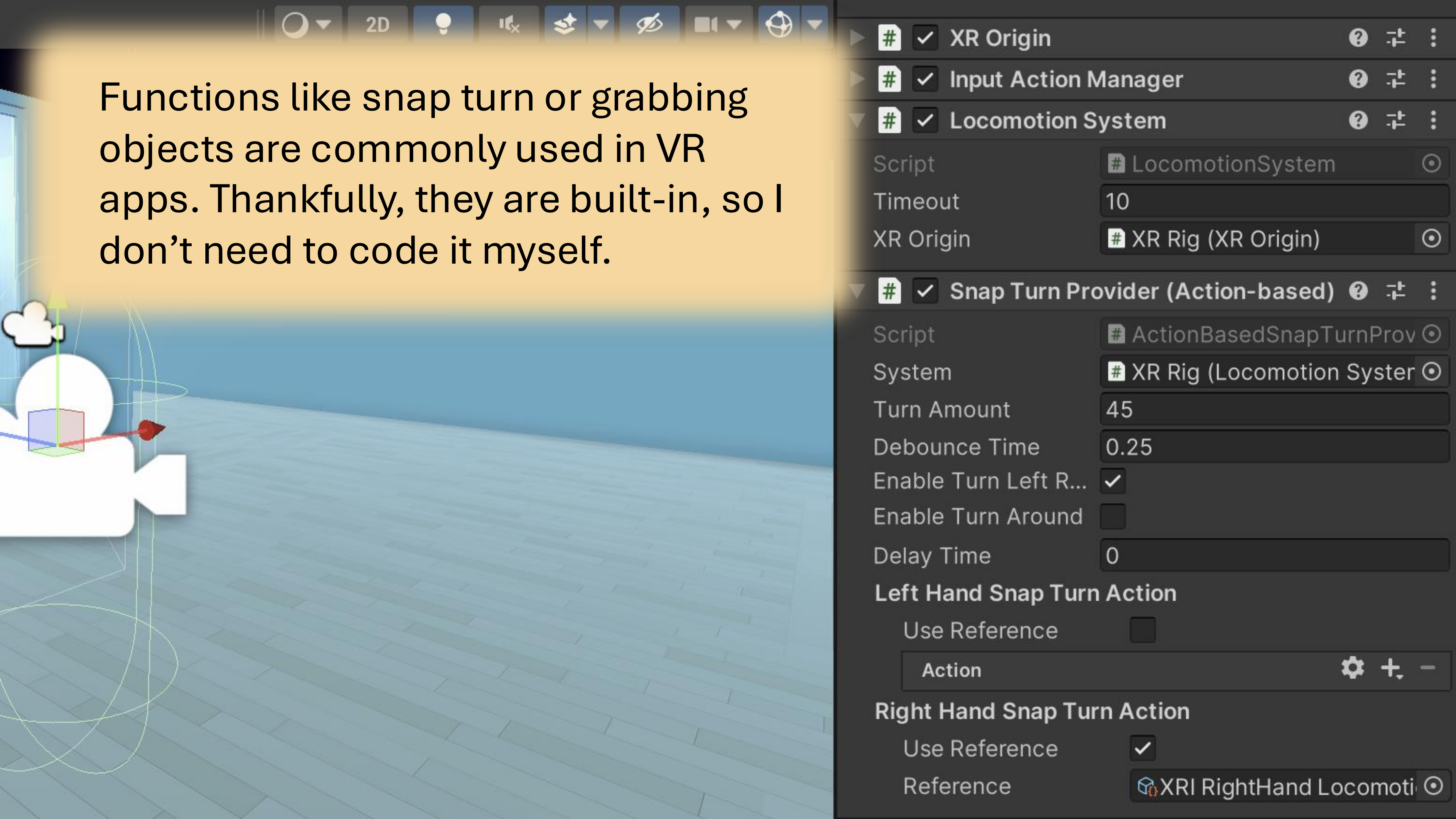
com.unity.xr.interaction.toolkit

Documentation | Changelog | Licenses

Description | Version History | Dependencies | Samples

A high-level, component-based, interaction system for creating VR and AR experiences. It provides a framework that makes 3D and UI interactions available from Unity input events. The core of this system is a set of base Interactor and Interactable components, and an Interaction Manager that ties these two types of components together. It also contains components that you can use for locomotion and drawing visuals.

Functions like snap turn or grabbing objects are commonly used in VR apps. Thankfully, they are built-in, so I don’t need to code it myself.



XR Origin

Input Action Manager

Locomotion System

Script	LocomotionSystem
Timeout	10
XR Origin	XR Rig (XR Origin)

Snap Turn Provider (Action-based)

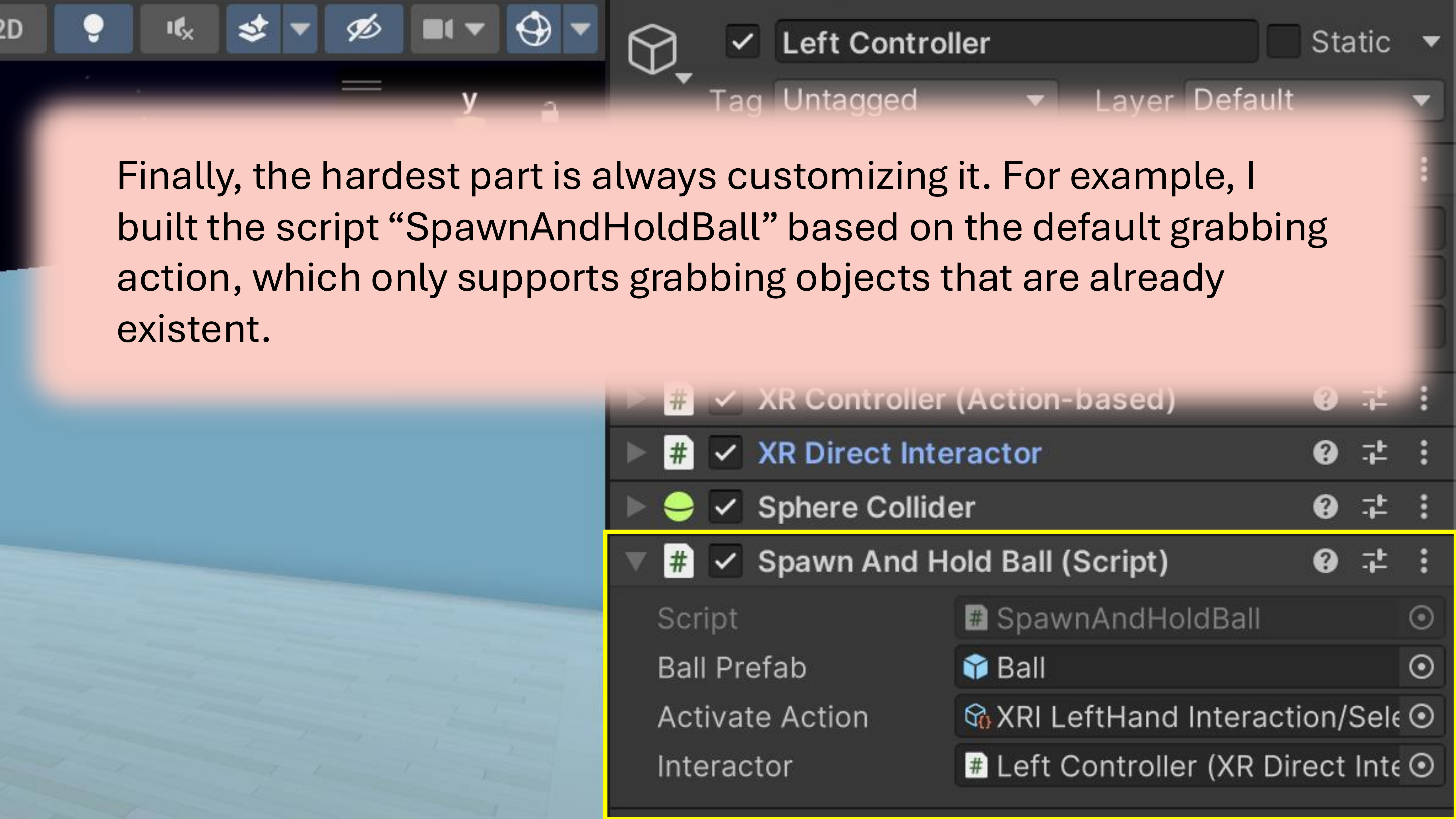
Script	ActionBasedSnapTurnProv
System	XR Rig (Locomotion Syster
Turn Amount	45
Debounce Time	0.25
Enable Turn Left R...	✓
Enable Turn Around	
Delay Time	0

Left Hand Snap Turn Action

Use Reference	
Action	

Right Hand Snap Turn Action

Use Reference	✓
Reference	XRI RightHand Locomoti



Left Controller
Static
Tag Untagged
Layer Default

Finally, the hardest part is always customizing it. For example, I built the script “SpawnAndHoldBall” based on the default grabbing action, which only supports grabbing objects that are already existent.

XR Controller (Action-based)
XR Direct Interactor
Sphere Collider
Spawn And Hold Ball (Script)

Script	SpawnAndHoldBall
Ball Prefab	Ball
Activate Action	XRI LeftHand Interaction/Sele
Interactor	Left Controller (XR Direct Inte

Goal 1 – Unity for VR Game Dev

- **Challenge: Troubleshooting**
  - Setting up VR project for is not easy: need to consider working systems, VR device, preferred packages etc.
  - Copying things from the tutorial project to my personal project: e.g., missing material
- **Solution**
  - Patiently figure out the root cause of the problem instead of immediately find a new method
  - Do a lot of research
  - Follow systematic tutorials

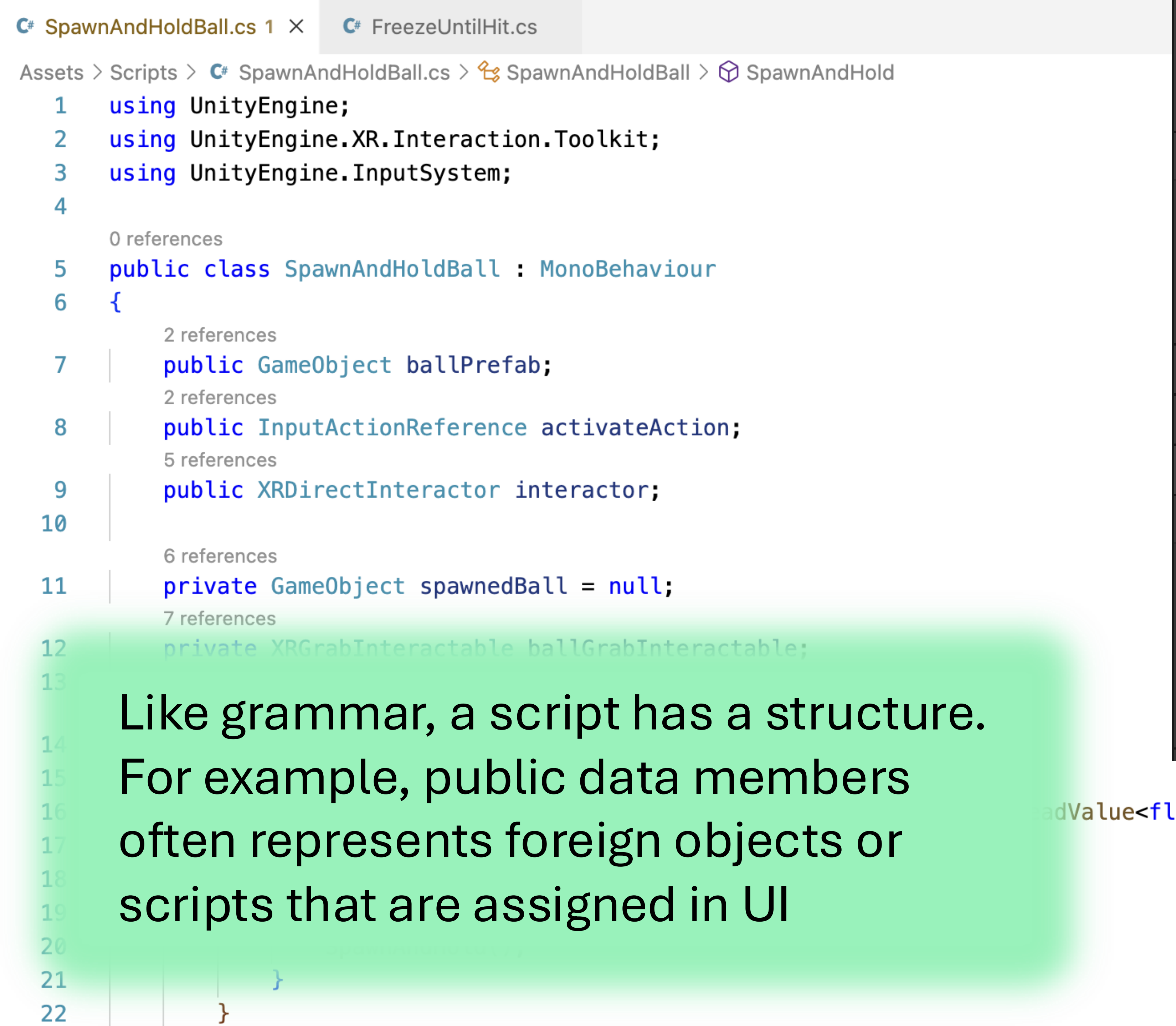
Goal 1 – Unity for VR Game Dev

- **Learning resources**
  - Hello World | Meta Horizon OS Developers
  - Unity Learn: Unity Essentials & VR Development pathways
  - Quality YouTube channels like Valem
  - Unity Documentation

Goal 2 – Programming in C# for Unity

- **Learning pathway**

Easy	▪ basic structure of a C# script
Medium	▪ common functions of Unity
Hard	▪ integration with external libraries like XR Interaction Toolkit

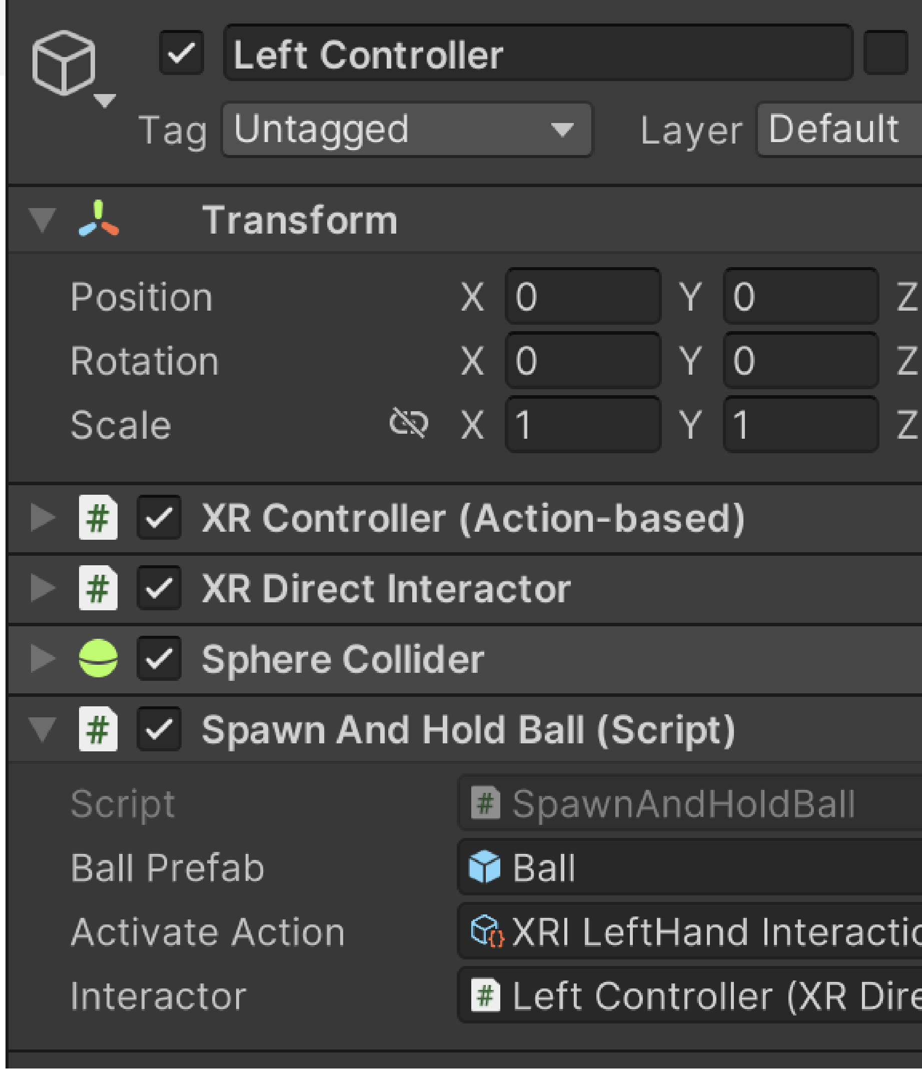

SpawnAndHoldBall.cs 1
FreezeUntilHit.cs
Assets > Scripts > SpawnAndHoldBall.cs > SpawnAndHoldBall > SpawnAndHold

```
using UnityEngine;
using UnityEngine.XR.Interaction.Toolkit;
using UnityEngine.InputSystem;

0 references
public class SpawnAndHoldBall : MonoBehaviour
{
    2 references
    public GameObject ballPrefab;
    2 references
    public InputActionReference activateAction;
    5 references
    public XRDirectInteractor interactor;

    6 references
    private GameObject spawnedBall = null;
    7 references
    private XRGrabInteractable ballGrabInteractable;
```

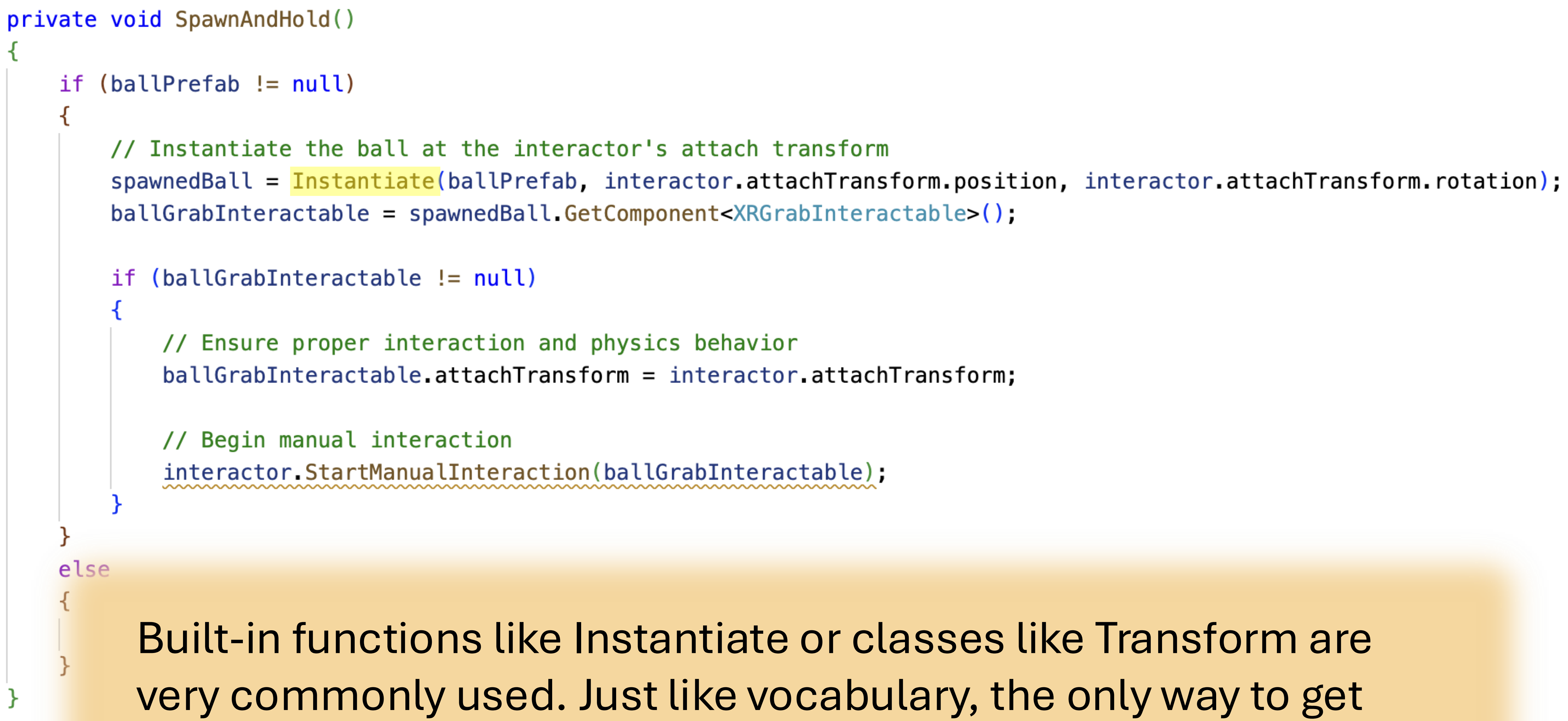
```
adValue<float>() > 0.5f)
        }
    }
```



	X	Y
Position	0	0
Rotation	0	0
Scale	1	1

Script	SpawnAndHoldBall
Ball Prefab	Ball
Activate Action	XRI LeftHand Interactio
Interactor	Left Controller (XR Dire

Like grammar, a script has a structure. For example, public data members often represents foreign objects or scripts that are assigned in UI



```
private void SpawnAndHold()
{
    if (ballPrefab != null)
    {
        // Instantiate the ball at the interactor's attach transform
        spawnedBall = Instantiate(ballPrefab, interactor.attachTransform.position, interactor.attachTransform.rotation);
        ballGrabInteractable = spawnedBall.GetComponent<XRGrabInteractable>();

        if (ballGrabInteractable != null)
        {
            // Ensure proper interaction and physics behavior
            ballGrabInteractable.attachTransform = interactor.attachTransform;

            // Begin manual interaction
            interactor.StartManualInteraction(ballGrabInteractable);
        }
    }
    else
    {

    }
}
```

Built-in functions like Instantiate or classes like Transform are very commonly used. Just like vocabulary, the only way to get used to them is to use them again and again.

```
private void ReleaseBall()
{
    if (spawnedBall != null && ballGrabInteractable != null)
    {
        ballGrabInteractable.interactionLayers = InteractionLayerMask.GetMask("ThrownBall");

        // End manual interaction to properly release the ball
        interactor.EndManualInteraction();

        // Cleanup
        ballGrabInteractable = null;
        spawnedBall = null;
    }
}
```

Following basic Unity functions are functions from XR Interaction Toolkit—just more features to learn.

Goal 2 – Programming in C# for Unity

- **Challenge: How to Find the Function for This and That**
  - Most functions or classes are pre-written, and the logic is usually not hard, which means my prior coding experience does not help much
  - No other way but to search functions one by one in the documentation, which is inefficient and boring
- **Solution**
  - Since AI has an enormous database, let it to the dull job
  - Follow systematic tutorials to know how to communicate with AI
  - Do a lot of research just to make small fixes—frustrating but unavoidable
- **Learning resources**
  - Unity Learn: Junior Programming pathway
  - Unity Documentation

Future Goals

- Continue my unfinished pathways and get certifications
- Develop another Unity 3D game independently

Special thanks to…

- **ATLAS team** for accepting me into the internship program and providing VR device
- **Michael, Randy, and Anju** for leading the team meetings, giving game ideas, and providing support