Virtual Reality in Unity 3D

Linya Ruan

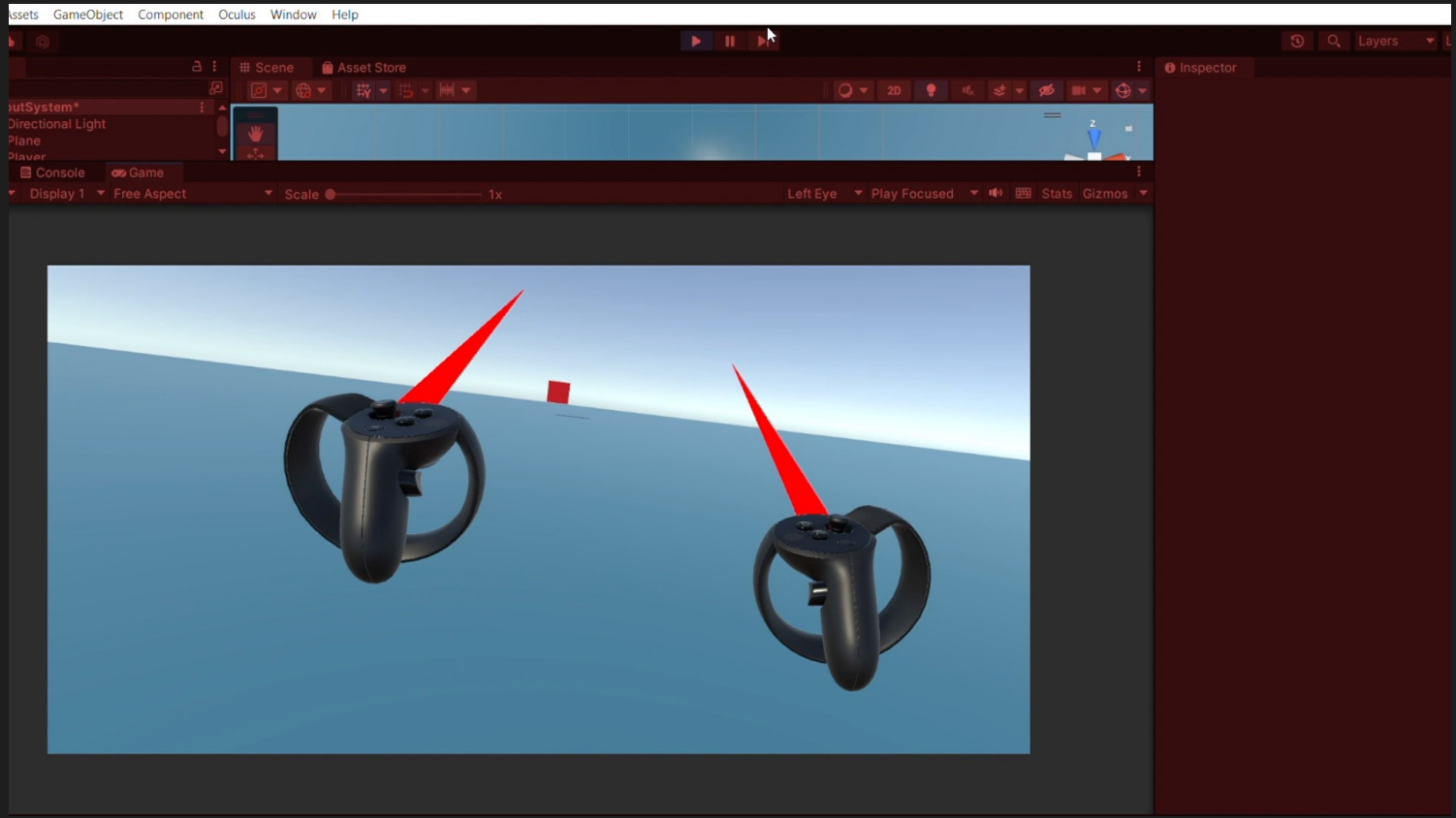
SRH Fachschulen

Stuttgart, 12. Juli, 2023

Zeitplan für den Kurs

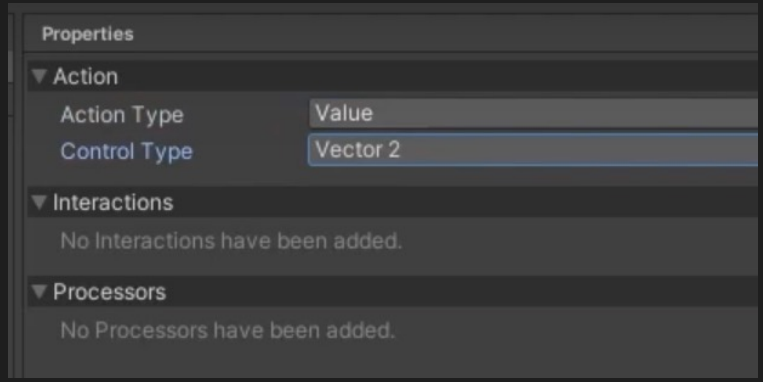
12-07-2023	• **Input System**

Part 7 !


Assets GameObject Component Oculus Window Help
Scene
Asset Store
Inspector
Layers
Directional Light
Plane
Console
Game
Display 1
Free Aspect
Scale
1x
Left Eye
Play Focused
Stats
Gizmos

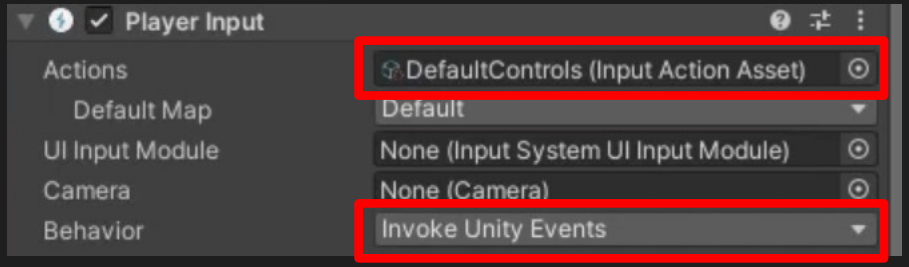
Input System

- Create a plane (blue) and a Cube (red)
- a Cube (red)= Player
- Install Input System
  - Window -> Package Manager -> Unity Registry -> Input System
  - Player Settings -> Play -> Active Input Handling (Input System Package (New) / Both)
- Create a Input Actions (rename: DefaultControls)
  - Add a Action Map (Default)
  - Add a Action (Move)
  - Change Action Type (Value) and Control Type(Vector 2)
  - Delete No Binding
  - Add Up/Down/Left/Right Composite (WASD)
  - Add Path (WASD)



Input System

- Add Play Input to Player Cube (red)



- Add a new c# script (PlayerMovement)

C# Program

API	Explain
Start ( )	
FixedUpdate ( )	
Update ( )	
LateUpdate ( )	
Order:	

C# Program

API	Explain
Start ( )	The code in Start is called when your game starts
FixedUpdate ( )	The code in Fixdate is called just before performing any physics calculations, and this is where your physics code will go. by default it executes every 0.02 seconds (50 times per second)
Update ( )	The code in Update is called once per frame of your game. If the frame rate is 60 FPS, it will execute 60 times a second; if it's 30, it will be 30 times a second.
LateUpdate ( )	It will execute after all the update functions are called (set up position of camera)
Order: Start > FixedUpdate > Update > LateUpdate	

C# Program

API	Explain
RequireComponent ( )	The RequireComponent attribute automatically adds required components as dependencies. [RequireComponent (typeof(Rigidbody))]
GetComponent ( )	To find components attached to a particular GameObject myResults = GetComponent<ComponentType>()
CharacterController	A CharacterController is not affected by forces and will only move when you call the Move function. It will then carry out the movement but be constrained by collisions. character.Move(moveVector * speed * Time.fixedDeltaTime);

C# Program

API	Explain
InputAction.CallbackContext	Information provided to action callbacks about what triggered an action. ReadValue<TValue>()
Vector2 ( )	Constructs a new vector with given x, y components. one with two dimensions, usually an x and a y. A Vector2 is typically used to represent a point in 2D space in a Unity game. Vector2 direction = new Vector2 (float x , float y) Vector2 direction = new Vector2 (0.0f , 1.0f)
Vector3 ( )	Constructs a new vector with given x, y, z components. This structure is used throughout Unity to pass 3D positions and directions around. Vector3 movement = **new** Vector3 (float x , float z, float y) Vector3 movement = **new** Vector3 (0.0f , 1.0f, 2.0f)

C# Program

API	Explain
Time.deltaTime Update ()	
Time.fixedDeltaTime FixedUpdate()	

C# Program

API	Explain
Time.deltaTime **Update ()**	FPS = Frames Per Second Better PC = Less time between frames (PC1=20 FSP ; PC2=500 FPS) Time.deltaTime = 1 / Framerate Regardless of how fast the computer is, the game will be played at the same speed. Note: Time.deltatime is used to update state of a game object per second instead of per frame
Time.fixedDeltaTime FixedUpdate()	As FixedUpdate is called after evey 0.02s. Time.fixedDelateTime does not vary. It always returns 0.02s. Note: Time.fixedDeltaTime is used to update state of a game object per physics update.

Input System

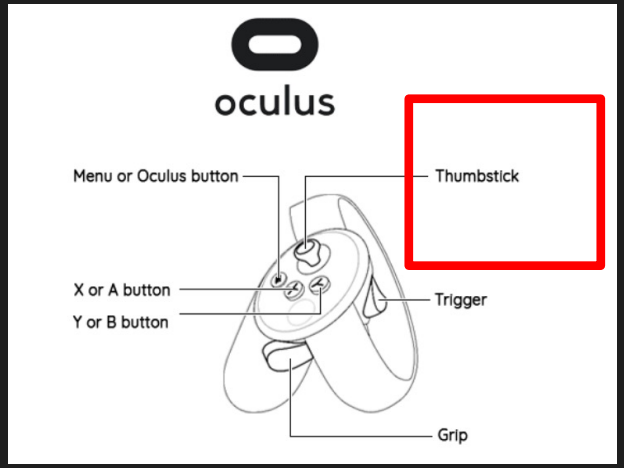
PlayerMovement.cs

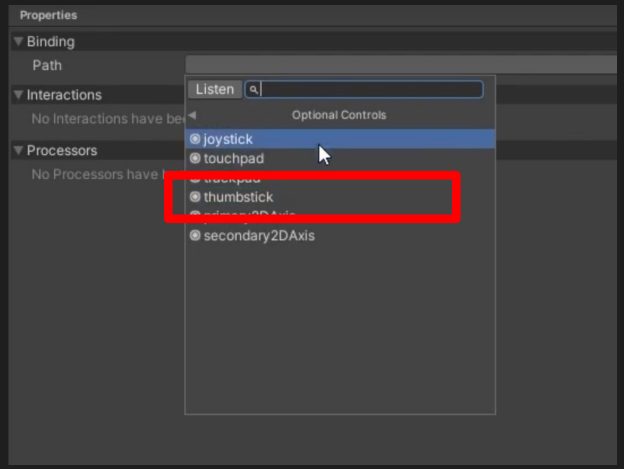
PlayerMovement › 无选择

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.InputSystem;

[RequireComponent (typeof(CharacterController))]
public class PlayerMovement : MonoBehaviour
{

    CharacterController character;
    Vector3 moveVector;
    public float speed = 10f;


    private void Start()
    {
        character = GetComponent<CharacterController>();

    }

    private void FixedUpdate()
    {
        character.Move(moveVector * speed * Time.fixedDeltaTime);
    }

    public void OnMovementChanged (InputAction.CallbackContext context)
    {
        Vector2 direction = context.ReadValue<Vector2>();
        moveVector = new Vector3(direction.x, 0, direction.y);
    }

}
```

Input System

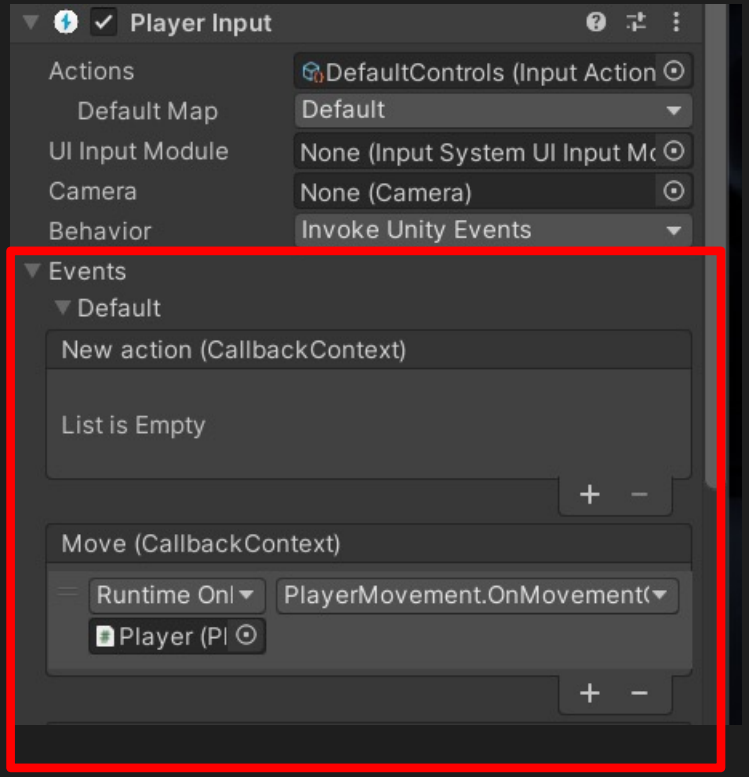
- Delete Main Camera
- Add XR Origin (Deviced-based -> XR Origin)
- Oculus Integration
- LeftHand Controller (LeftControllerPf)
- RightHand Controller (RightControllerPf)
- Go to DefaultControls -> move -> Add Binding -> Path (XR Controller) -> Optional Controls -> thumbstick
- Save Asset



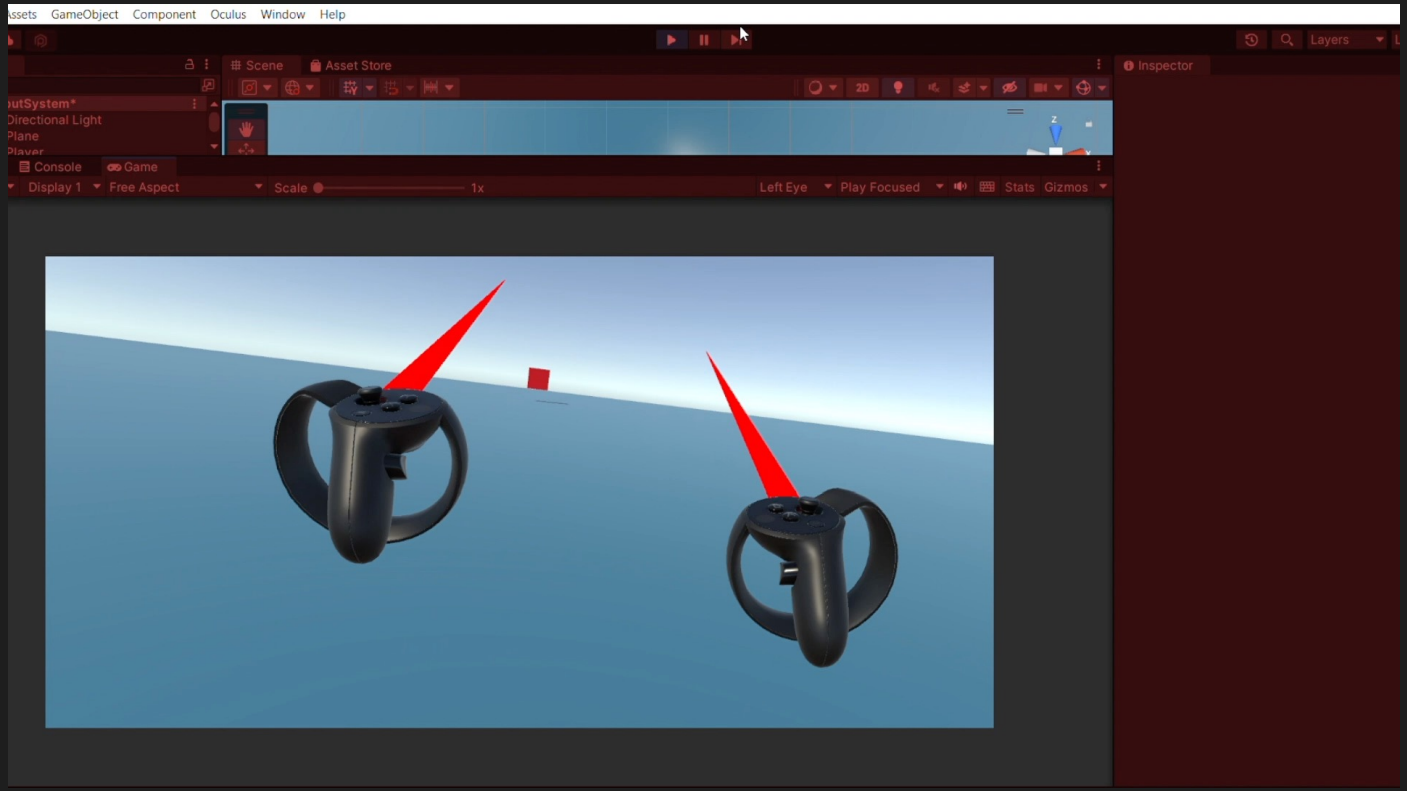


Input System

- Add a Event (Move (CallbackContext))



Finish Part 7 !



Aufgaben