

PLATEAU Hands-On 03

PLATEAU

を使った

Unity VR アプリ開発

ユニティ・テクノロジーズ・ジャパン株式会社
Senior Solution Engineer
高橋 忍 shinobu.takahashi@unity3d.com

Agenda

プロジェクト作成

プロジェクトの作成

XR環境構築

Oculus Quest 環境構築

VRの世界に入る

モーションコントローラーの接続

移動する 1/2/3

PLATEAU データの設置

カスタム開発

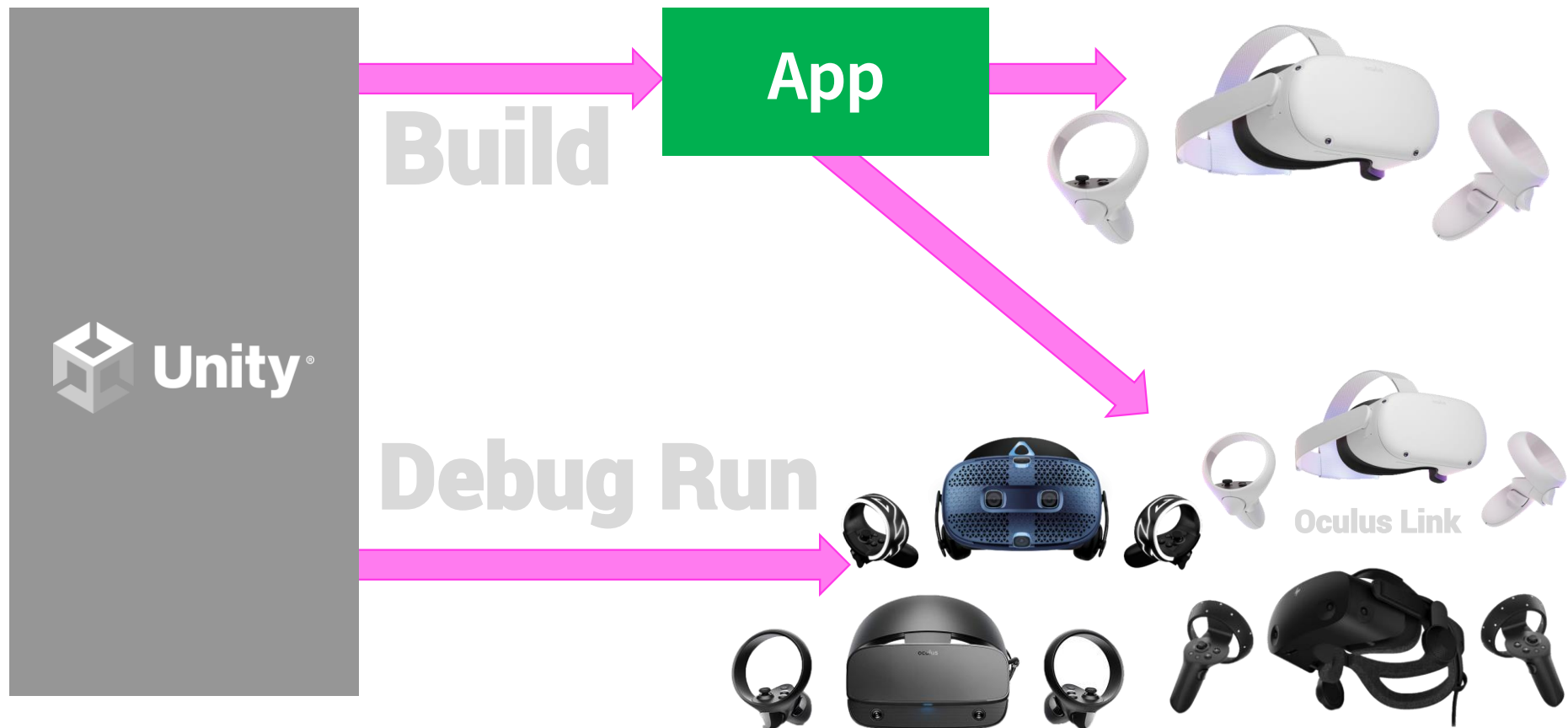
空中移動の実現

太陽の移動/Skybox カスタマイズ

動画再生

01. プロジェクト作成

Unity での開発環境と流れ



VRアプリ開発のためのUnity Package

VR App



XR Interaction Toolkit

XR Subsystem



XR Plugin Management



Oculus XR
plugin



Windows MR
plugin



Open XR
plugin



Unity®

各社VRデバイス用の XR Plugin



Oculus XR plugin

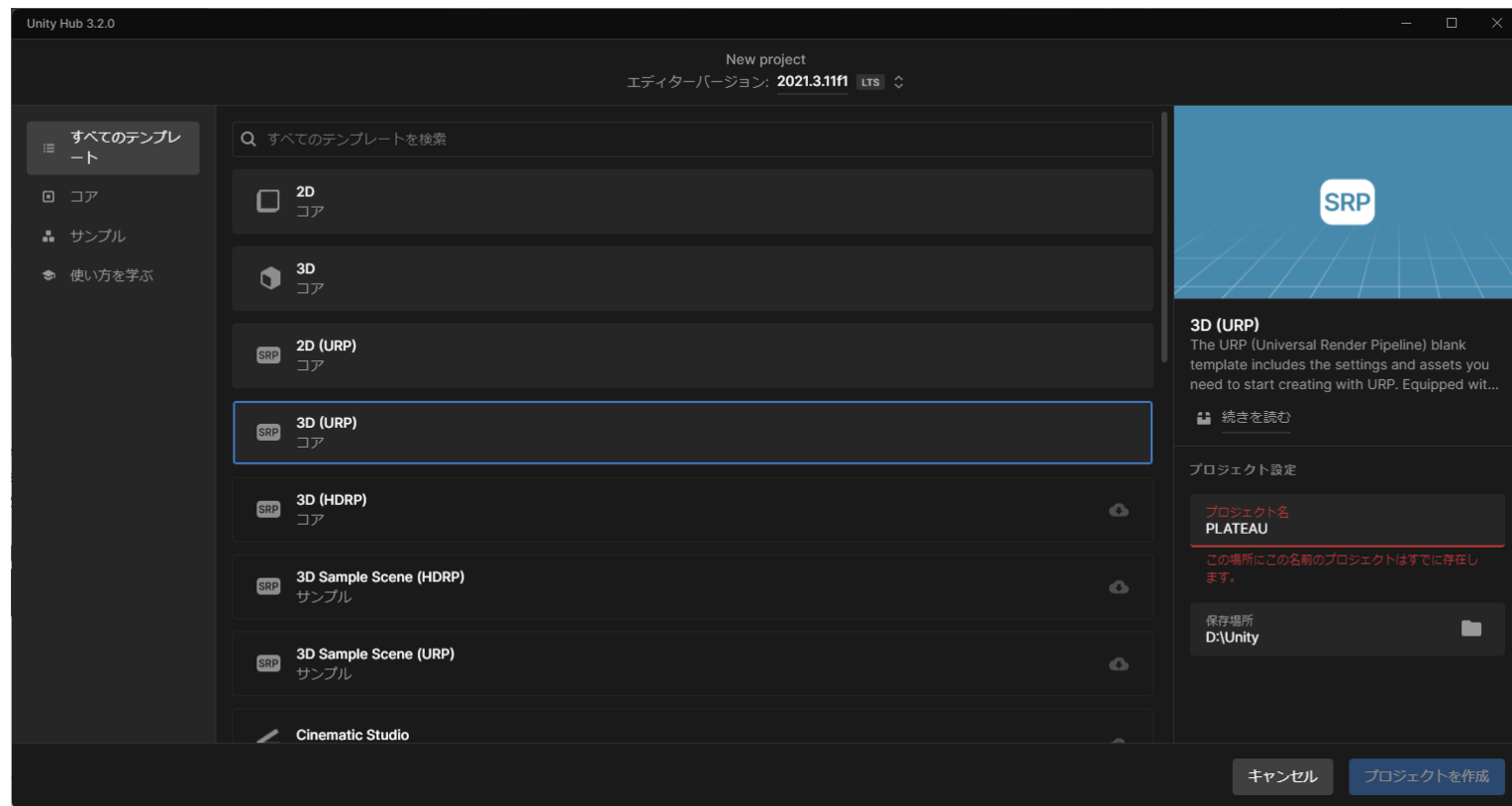
Windows MR plugin

Open XR
plugin

Project 作成

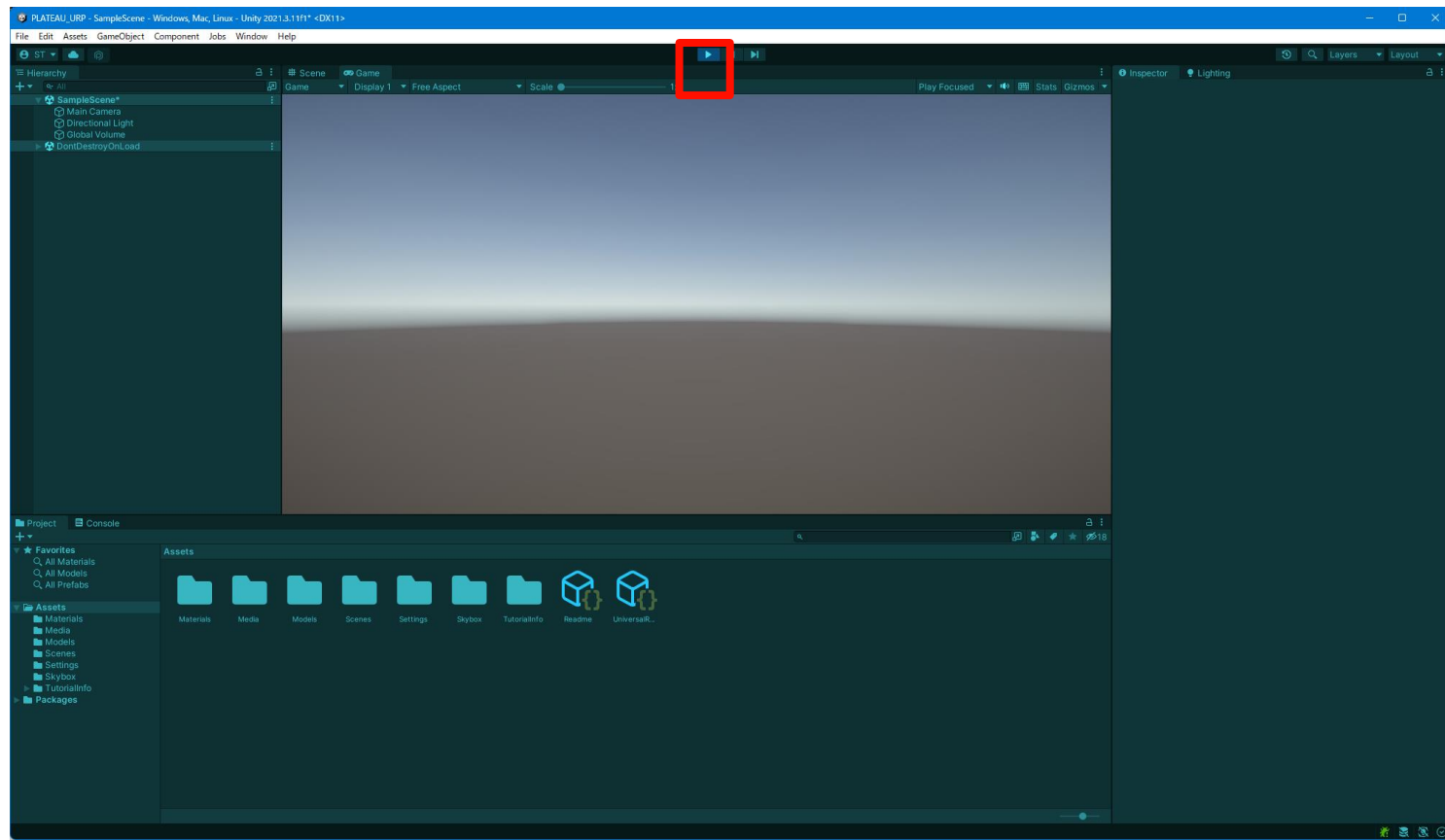
Unity Hub

3D (URP)



Unity Project

ひとまず実行



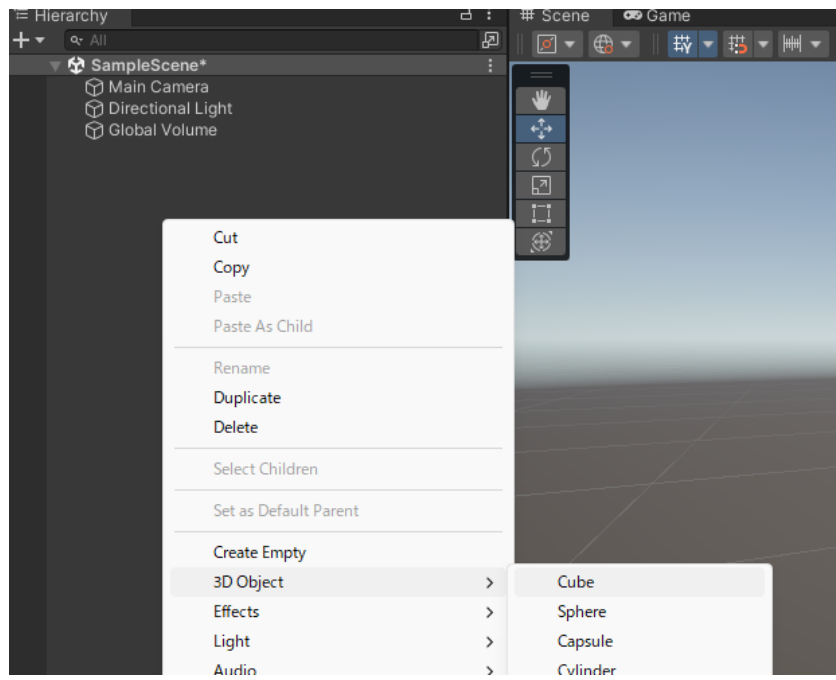
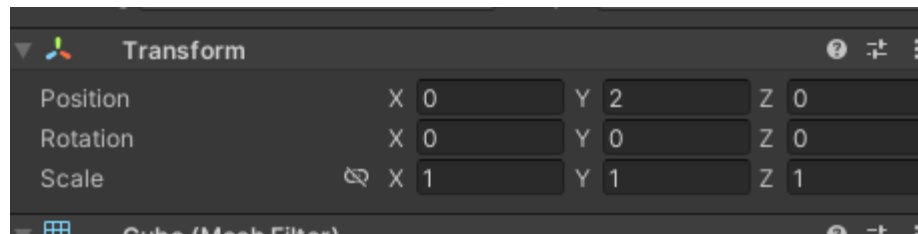
Cubeの追加

Hierarchy

3D Object > Cube

Transform > Position

0,2,0



Cubeの追加

Material 作成

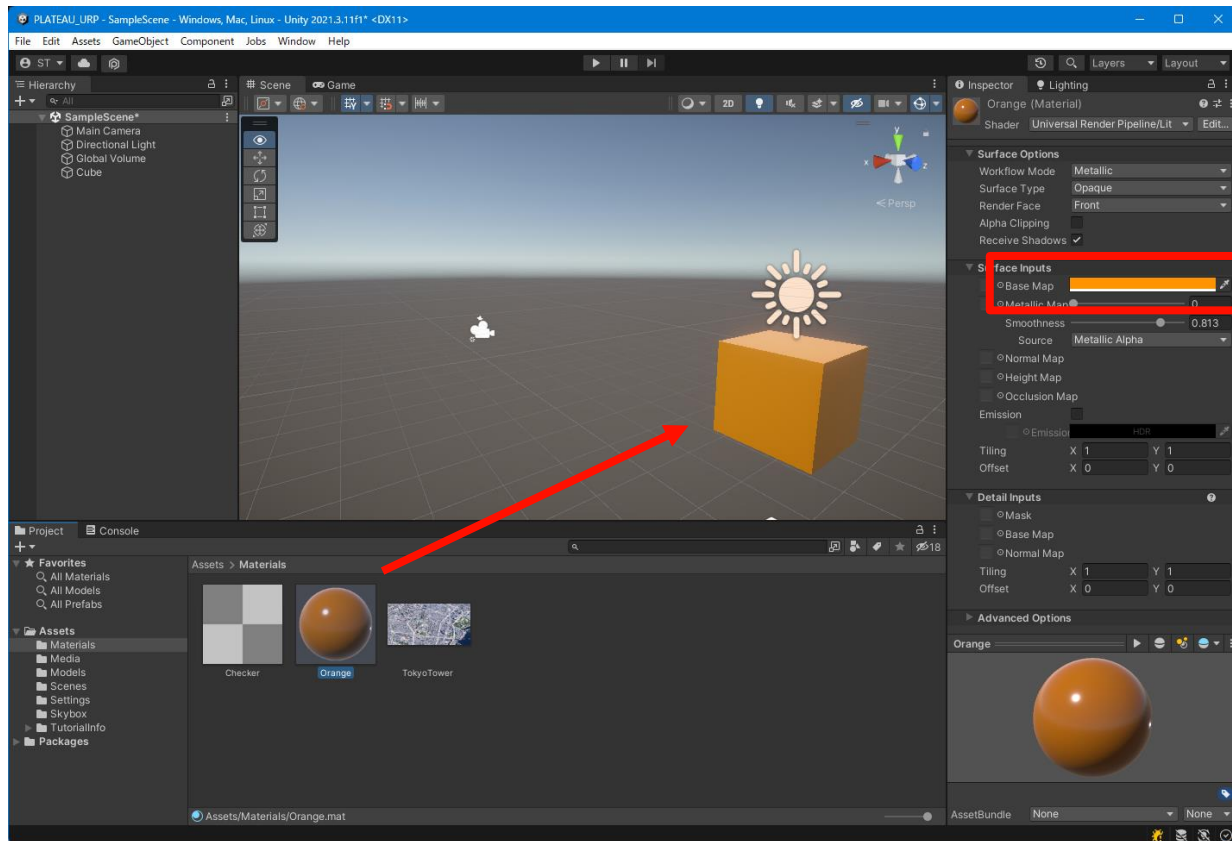
Asset Material フォルダ

Create > Material

BaseMap で色変更

Smoothness 小アップ

CubeにD&D



Plane 作成

Plane追加

3D Object > Plane

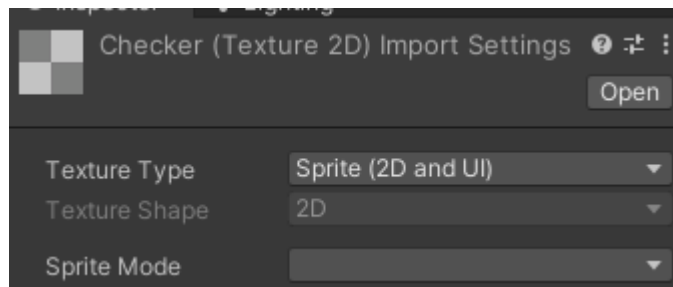
Position 0,0,0



テクスチャ画像変換

Material フォルダ Checker

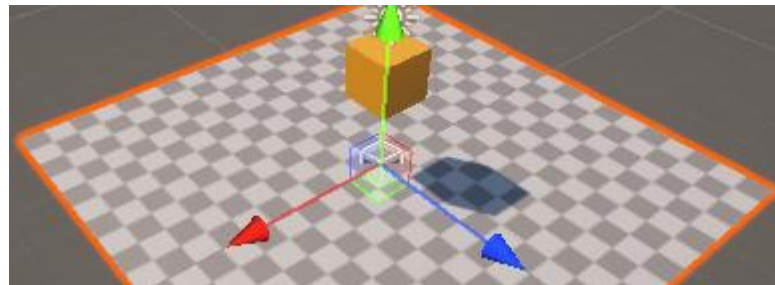
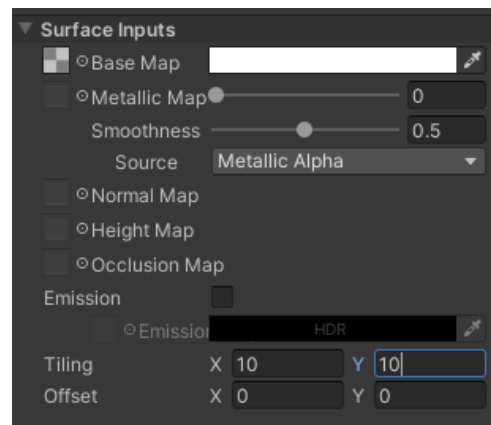
Texture Type : Sprite



貼り付け

Checker を Plane にD&D

Plane > Checker > Tiling > 10,10



02. XR環境構築

XR Plugin Management

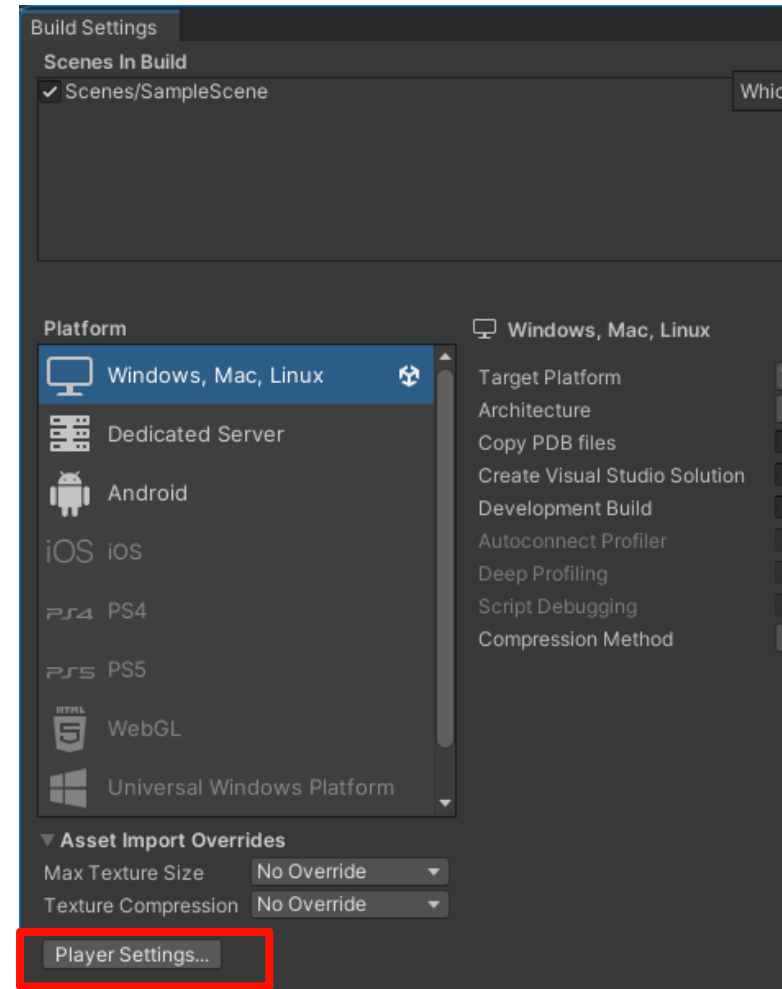
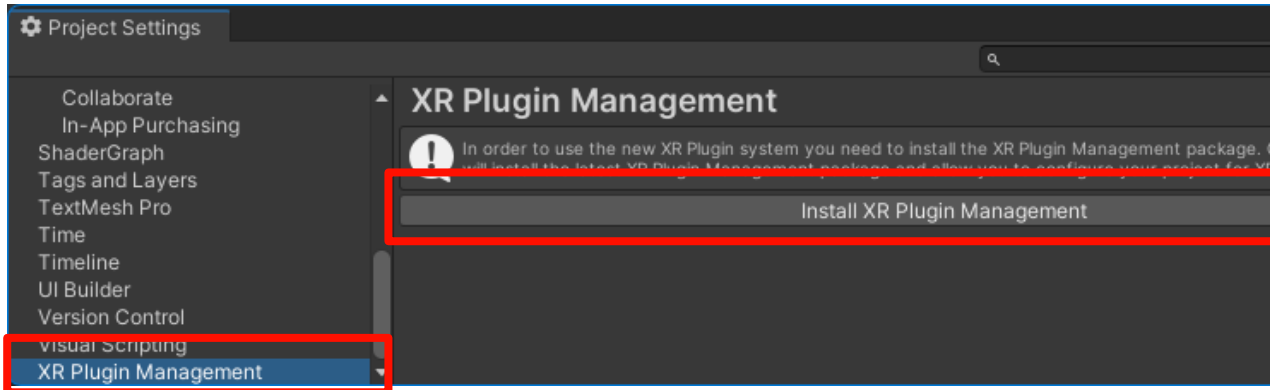
Build Settings

File > Build Settings > Player Settings

Player Settings

XR Plugin Management

Install XR Plugin Management



XR Plugin Management

Player Settings

XR Plugin Management 続き

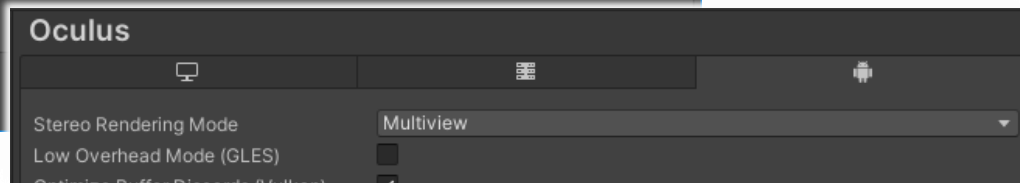
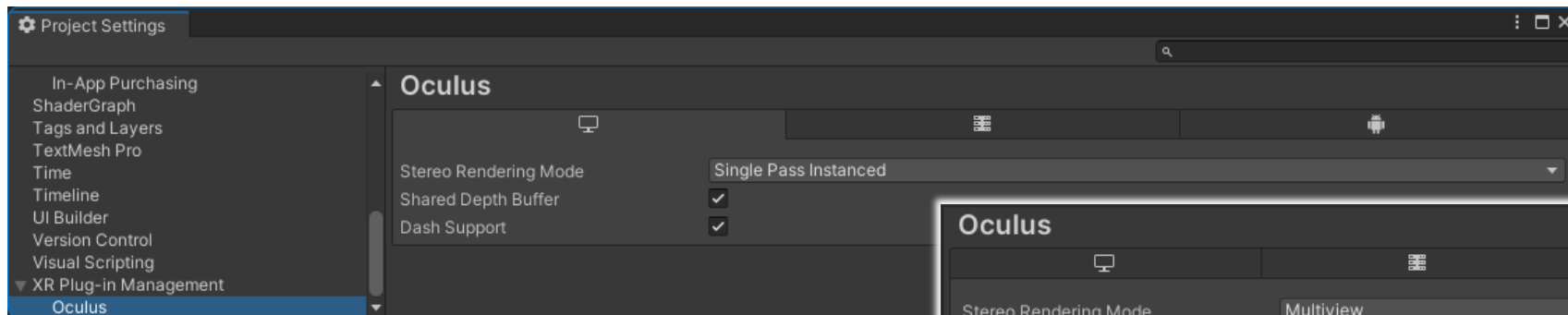
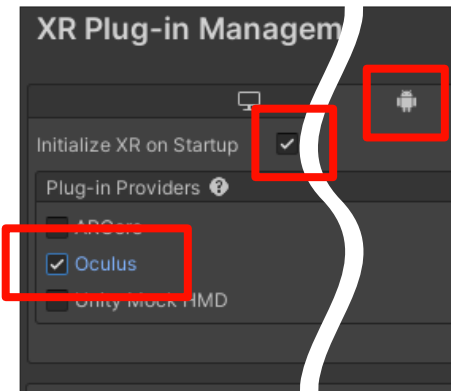
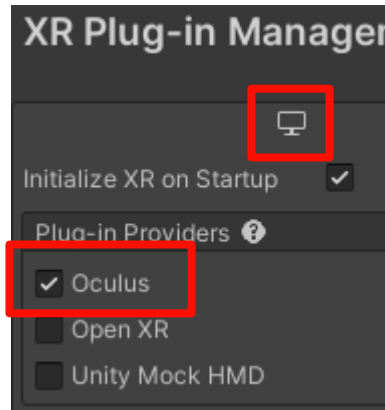
PC **Oculus** にチェック (インストール)

Android **Initialize XR on Start Up** / **Oculus** にチェック

XR Plguinmanagement / Oculus

PC **Single Pass Instance**

Android **Multiview**



XR Plugin Management

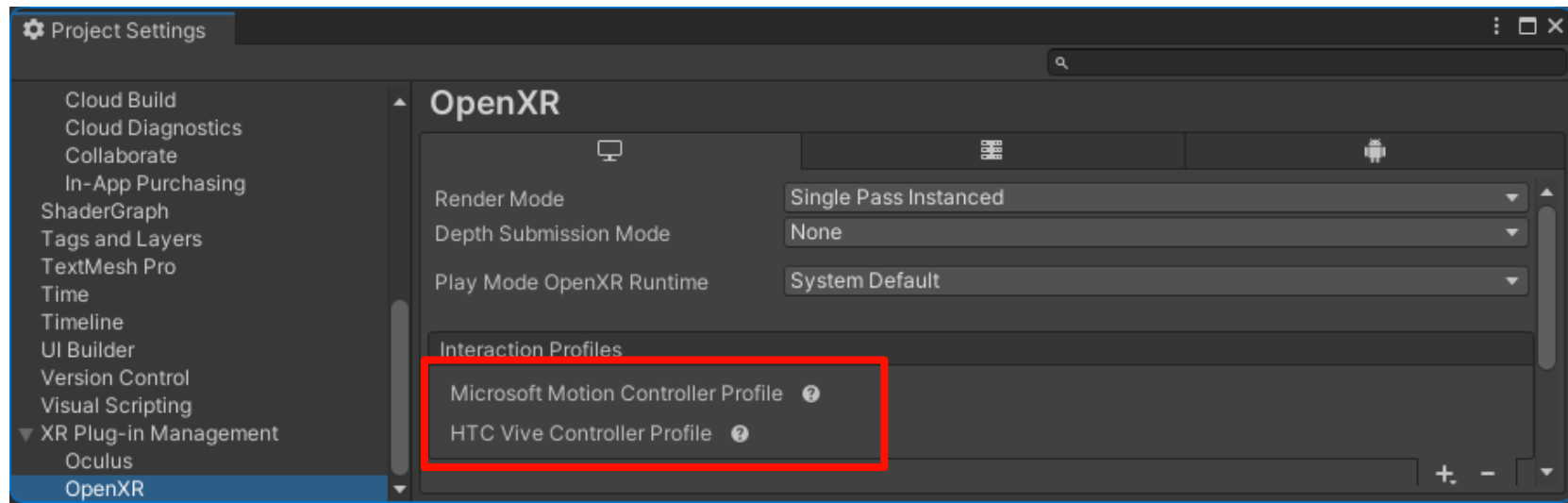
Player Settings

Windows Mixed Reality / HTC VIVE

PC Open XR にチェック (インストール & 再起動)

XR Pluginmanagement / Open XR

PC Interaction Profile をそれぞれ追加



XR Interaction Toolkit

Window > Package Manager

Packages : **Unity Registry**

XR Interaction Toolkit > **Install**

Dialog が出たら **Go Ahead !**

The screenshot shows the Unity Package Manager window. The 'Packages: Unity Registry' tab is active, and the 'XR Interaction Toolkit' package is selected in the list. The package details on the right show it is version 2.0.3, released on August 15, 2022, and is available in the Unity Registry. A dialog box titled 'XR InteractionLayerMask Update Required' is displayed, warning that the project may contain an obsolete method to validate interactions between XR Interactors and Interactables. The dialog offers two options: 'Made a Backup, Go Ahead!' and 'No Thanks'. The 'Install' button at the bottom right of the package details is highlighted with a red box.

Package Name	Version	Status
Universal RP	12.1.7	✓
Version Control	1.17.2	✓
Visual Effect Graph	12.1.7	
Visual Scripting	1.7.8	✓
Visual Studio Code Editor	1.2.5	✓
Visual Studio Editor	2.0.16	✓
XR Interaction Toolkit	2.0.3	
XR Plugin Management	4.2.1	

XR Interaction Toolkit Release

Unity Technologies
Version 2.0.3 - August 15, 2022
Registry Unity
com.unity.xr.interaction.toolkit
[View documentation](#) • [View changelog](#) • [View licenses](#)

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.

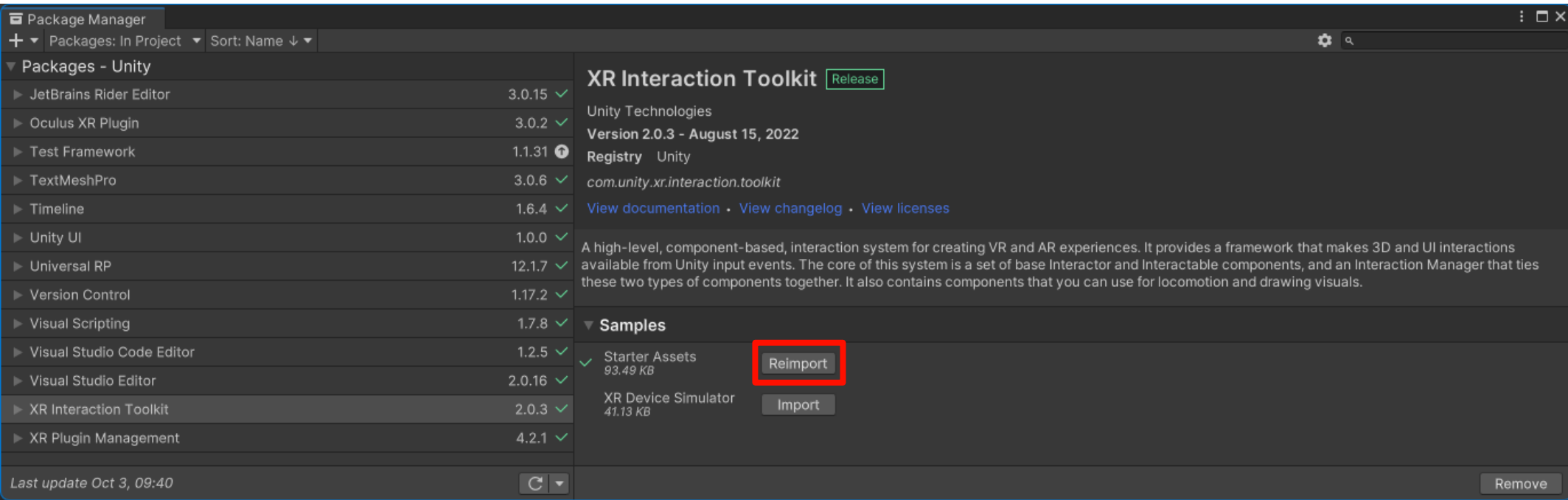
Install

XR Interaction Toolkit

サンプルのインストール

Packages : **In Project**

XR Interaction Toolkit > Samples > **Import**



03. Quest 2 Build環境

Android Build Settings

File > Build Settings

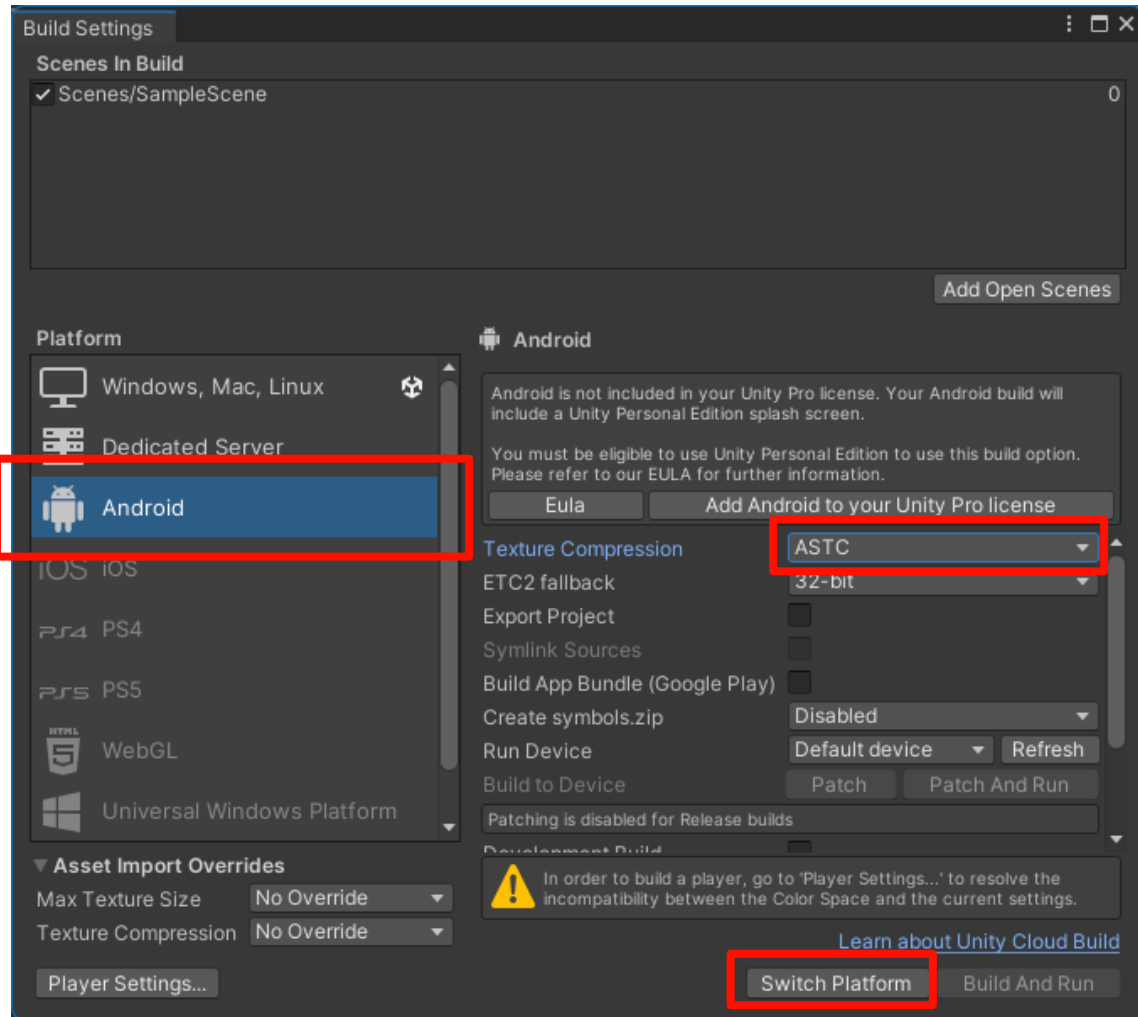
Platform > **Android** に

Texture Compression

> **ASTC**

Switch Platform

Player Settings ...



Android Player Setting

Player Settings

Rendering

Color Space : **Linear** のまま

Auto Graphics API : **チェック外す**

Graphics API : Vulkan 消す。**OpenGL ES3**

Identification

Pakcagename : **com.企業名.プロジェクト名**

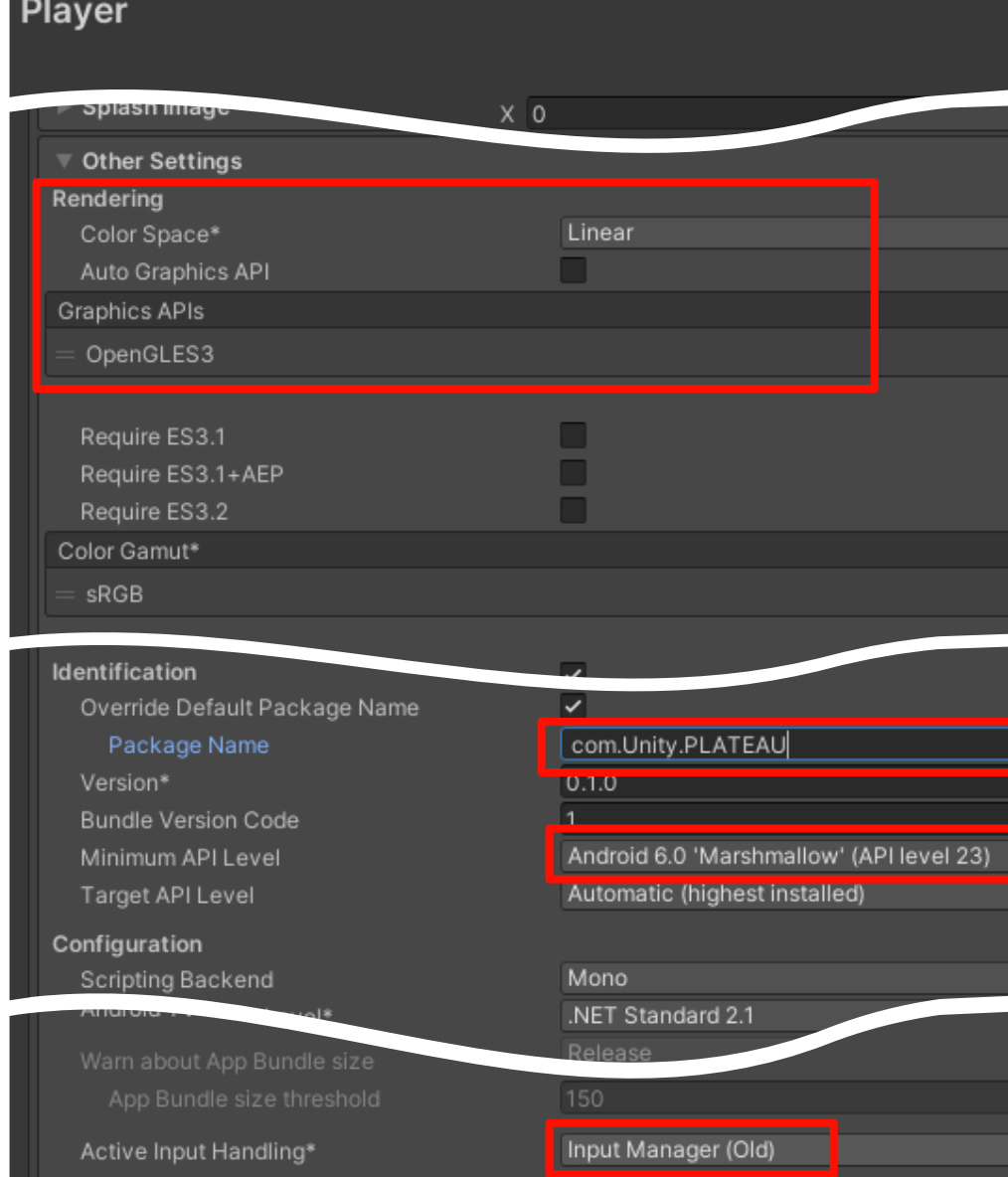
Minimum API Level : **Android 6.0** のまま

Configuration

Active Input Handling :

Input System Package (New)

Apply → **再起動**



テスト実行@Quest 2

Oculus 2 接続

USB接続

ファイル転送許可 ON

Build Settings

Build And Run

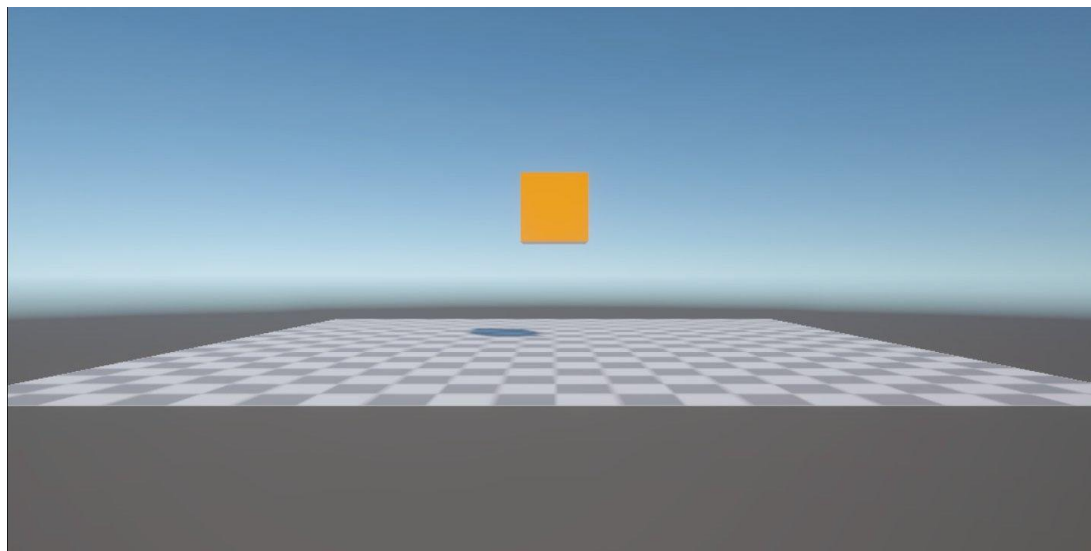
APK格納フォルダ作成

Apk名指定してbuild

動作確認

Skybox が出て動かないでOK

メニューボタンでアプリ終了



Windows 環境に戻す

Build Settings

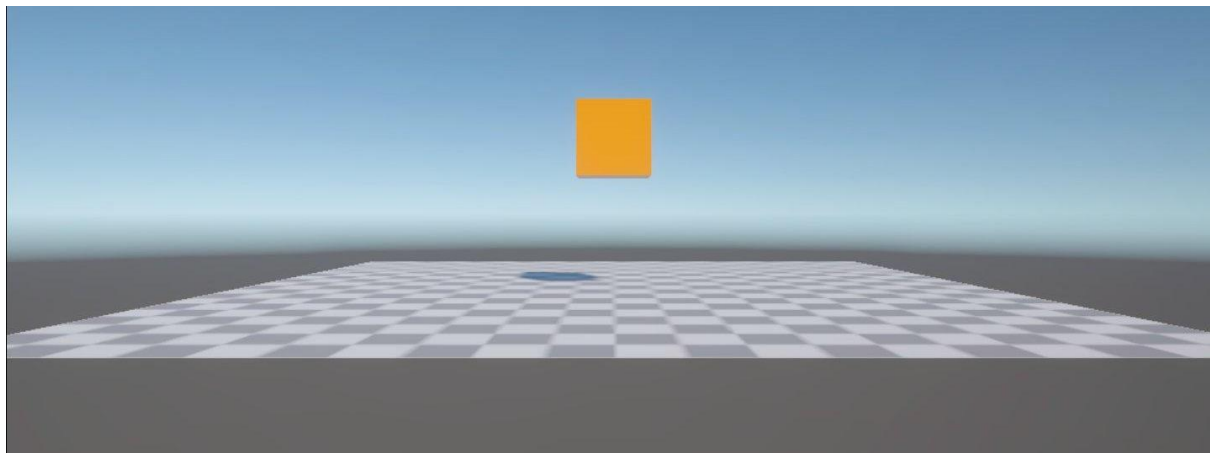
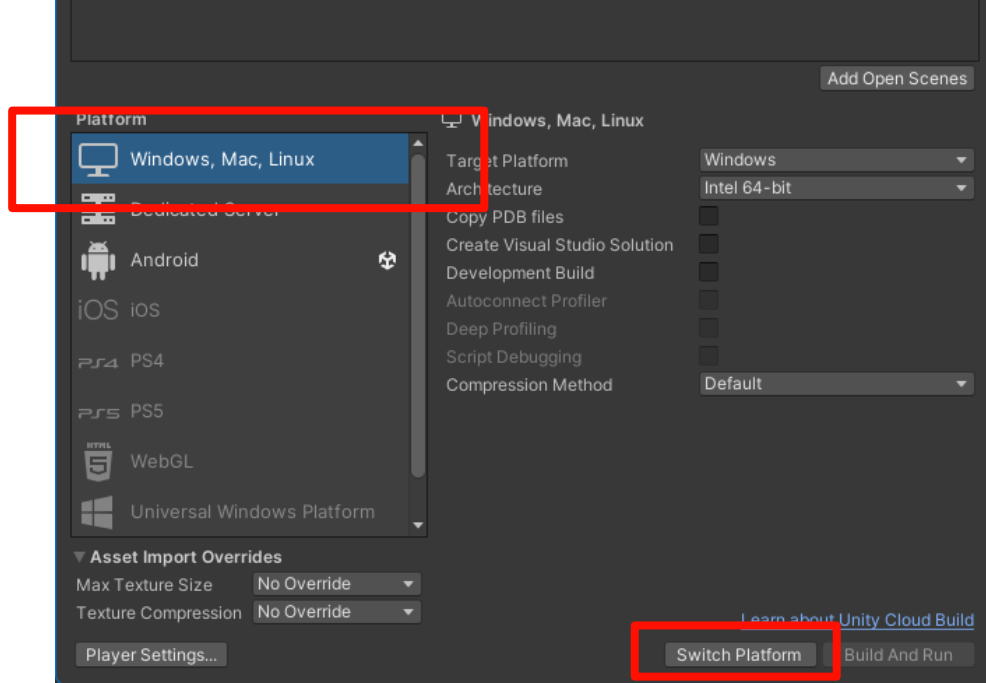
Platform > Windows, Mac, Lunx に
Switch Platform

Oculus Linkを実行

Oculus Rift UI を確認

Unity テスト実行

頭に合わせて景色も揺れる
動かないSkybox が表示

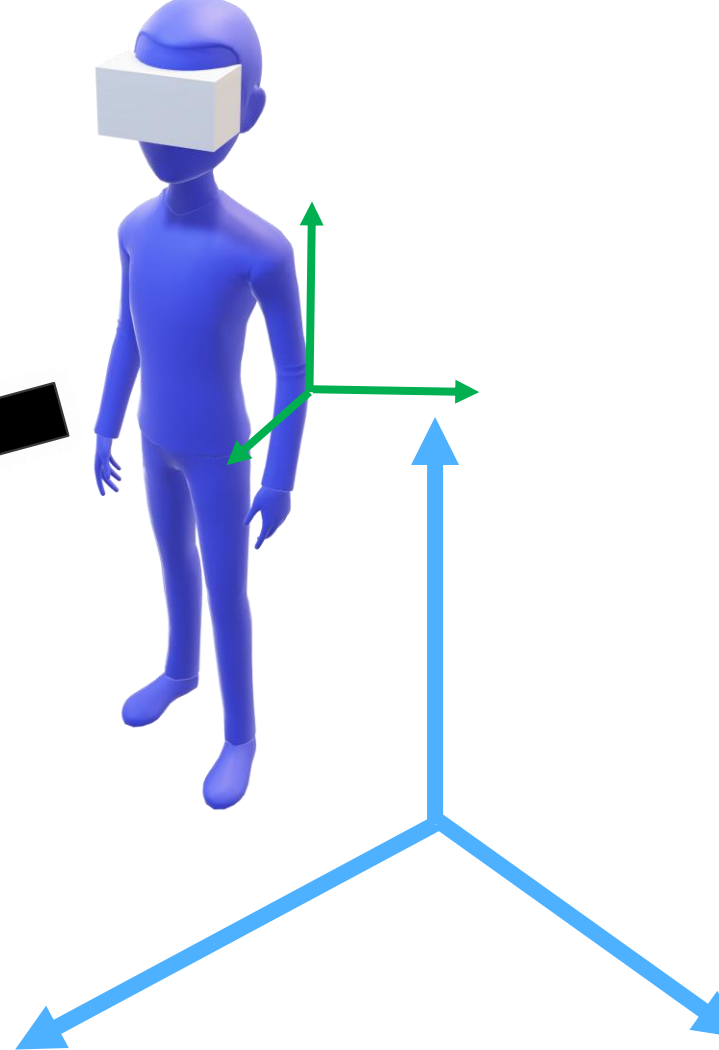
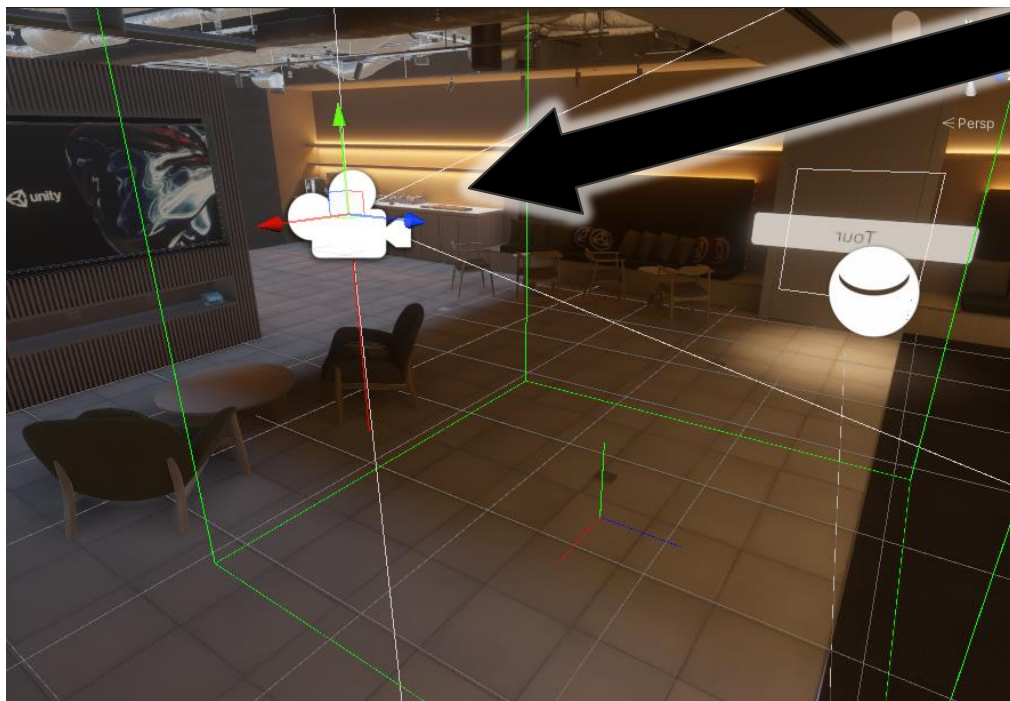


04. VRの世界に入る

VR空間に入る

HMD = Cameraにする

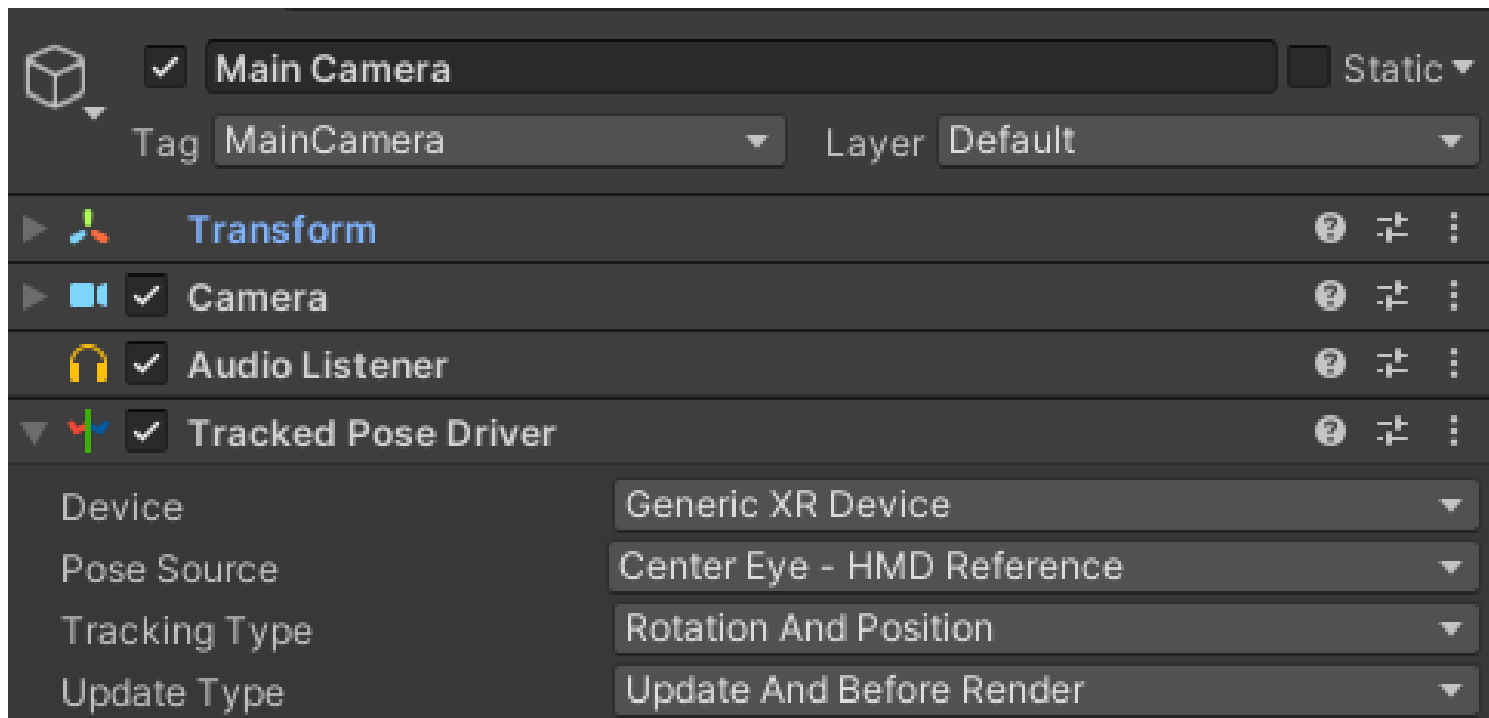
常にCamera Transform が上書きされる



VR空間に入る

HMDとカメラを同期させる Tracked Pose Driver

HMDの情報をメインカメラに同期させるコンポーネント

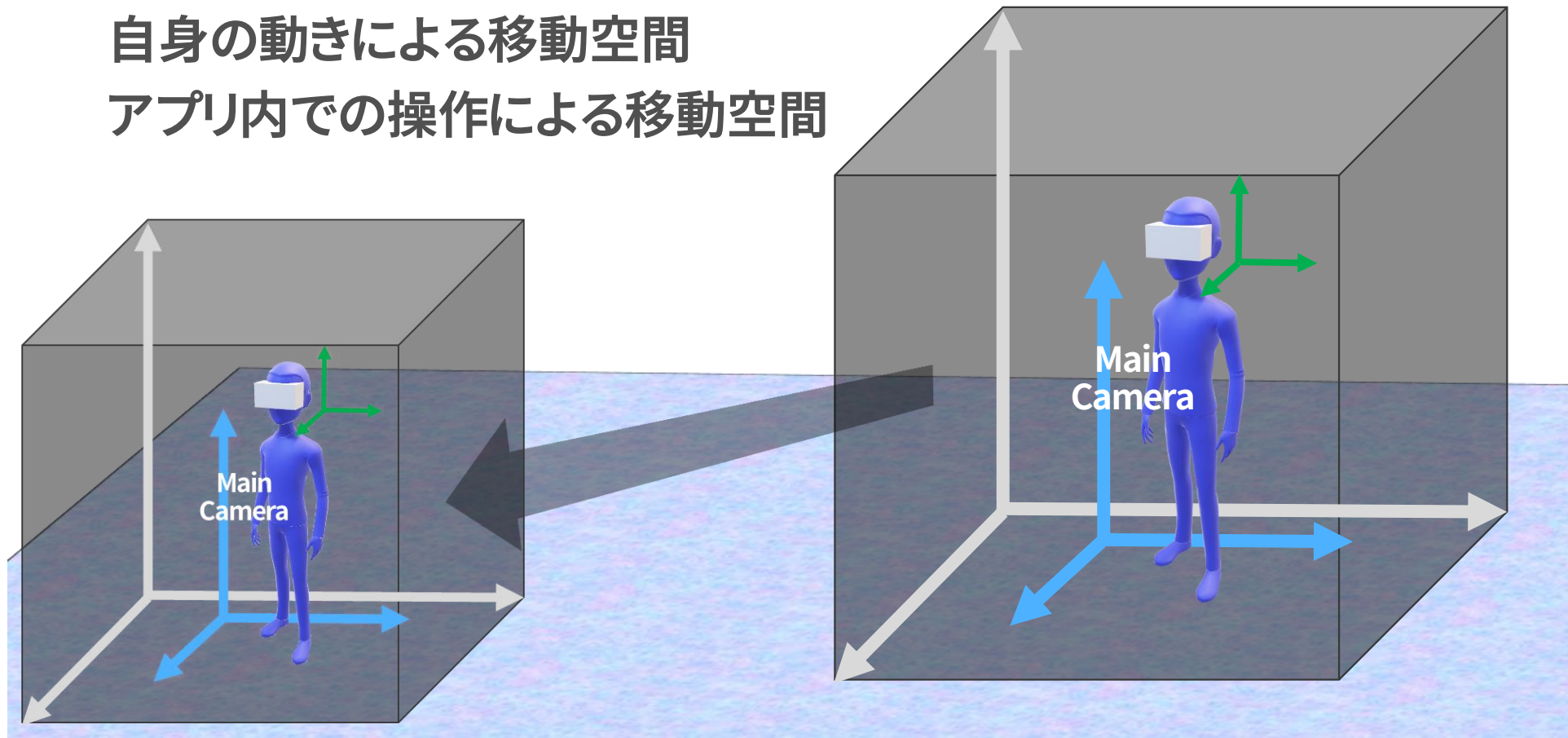


VR用の移動

2つの階層化された移動空間

自身の動きによる移動空間

アプリ内での操作による移動空間



XR Origin

Camera Offset

VR空間内移動用のGameObject

Programingで移動可能

Main Camera

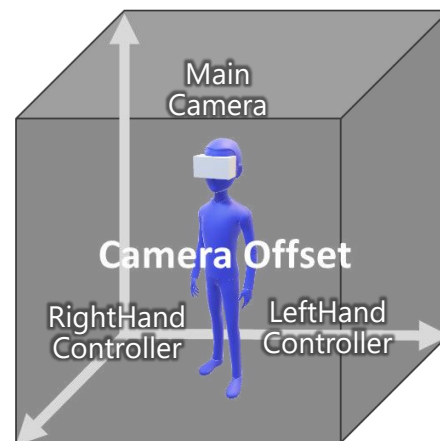
Tracked Pose Driver が入ったHMD連動のMain Camera

Program では操作できない

LeftHand/RightHand Controller

モーションコントローラー オブジェクト

Program では操作できない



VR デバイスとの接続

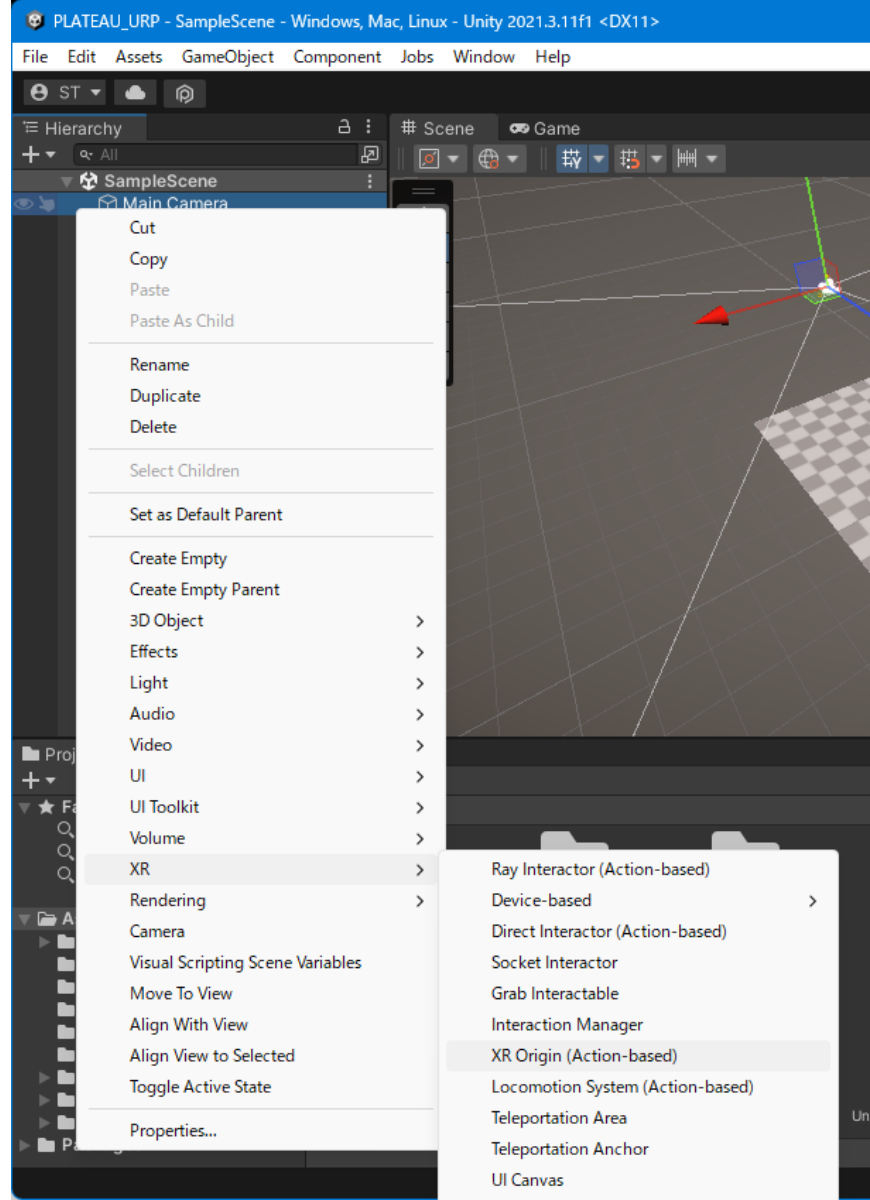
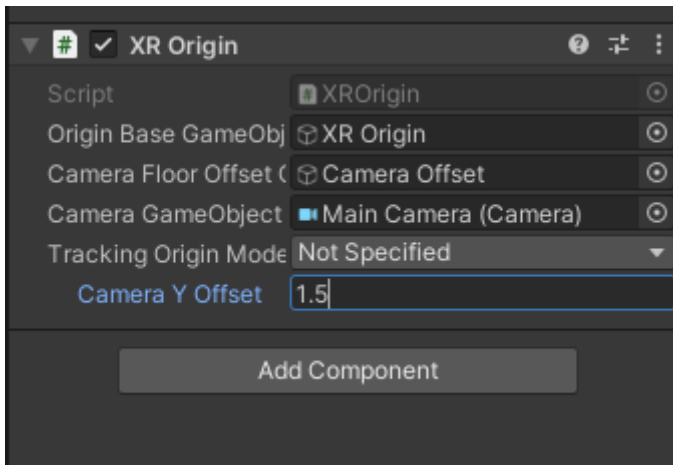
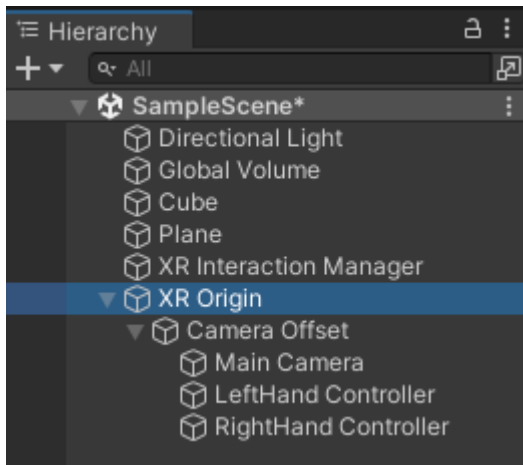
XR Origin の追加

XR > XR Origin(Action Based)

Main Camera が XR Origin 内に移動

XR Origin > Camera Y Offset 1.5

テスト実行



05. モーションコントローラー接続

Motion Control Object

<https://developer.oculus.com/downloads/package/oculus-controller-art/>

左右のモデルを分解

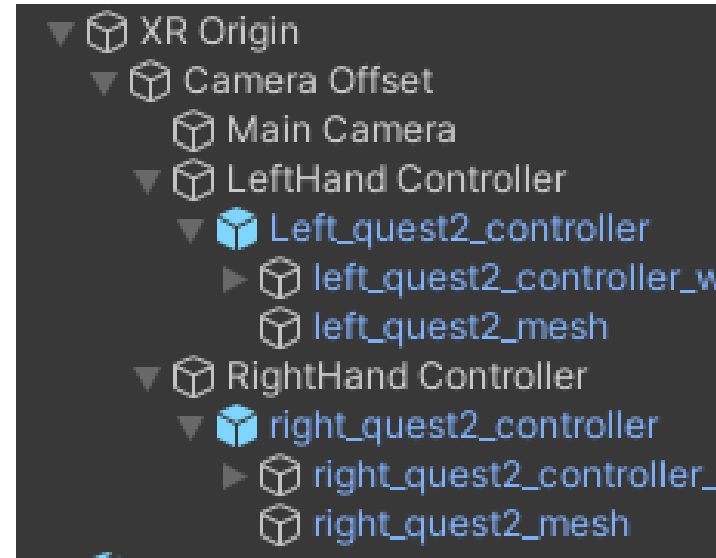
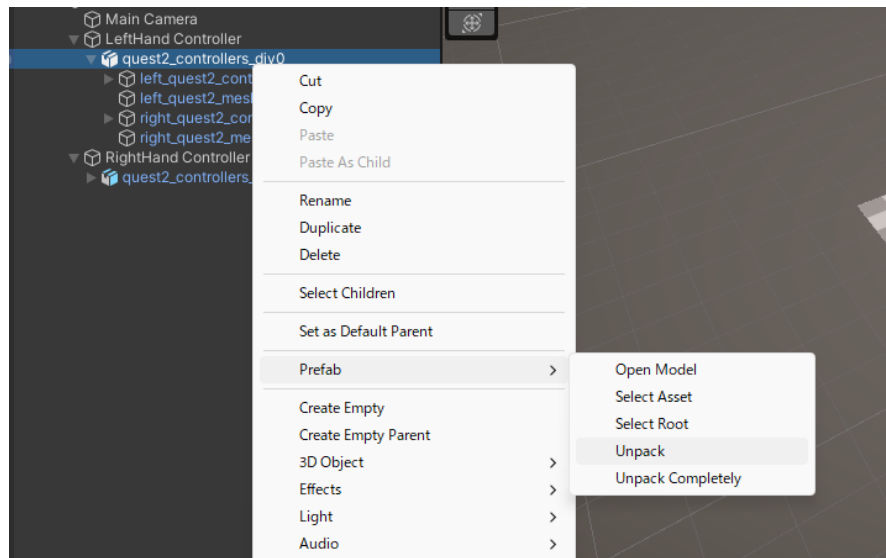
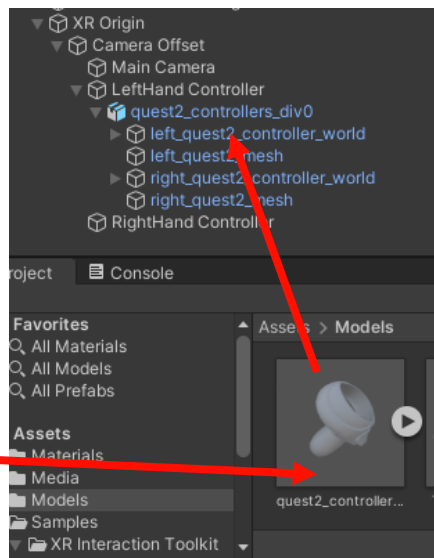
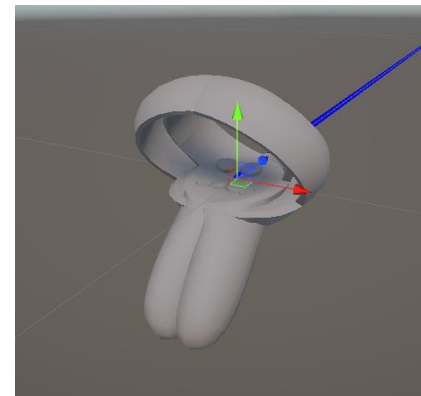
quest2_controllers_div0.fbx を Modelsフォルダにドロップ

更に XR Origin / Camera Offset / Left handcontroller にドロップ

ドロップしたPrefabをUnpack

right_quest2_controller_world, right_quest2_mesh を削除

Right Handcontrollerも同様に

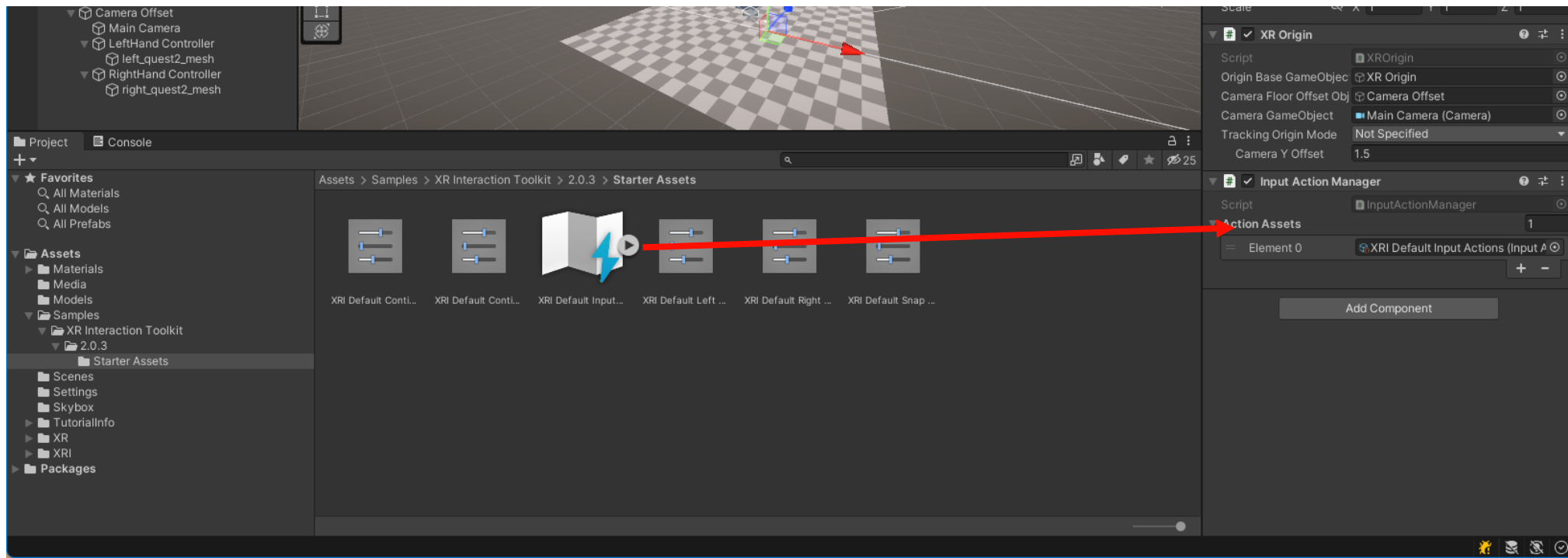


Motion Control の動きを設定

XR Origin | Input Action Manager 追加

XR Origin | Add Component | Input Action Manager

Asset/Samples/XR Interaction Toolkit/2.0.3/Starter Asset/XR Default Input Actions を追加した Input Action Manager の Action Asset にドロップ

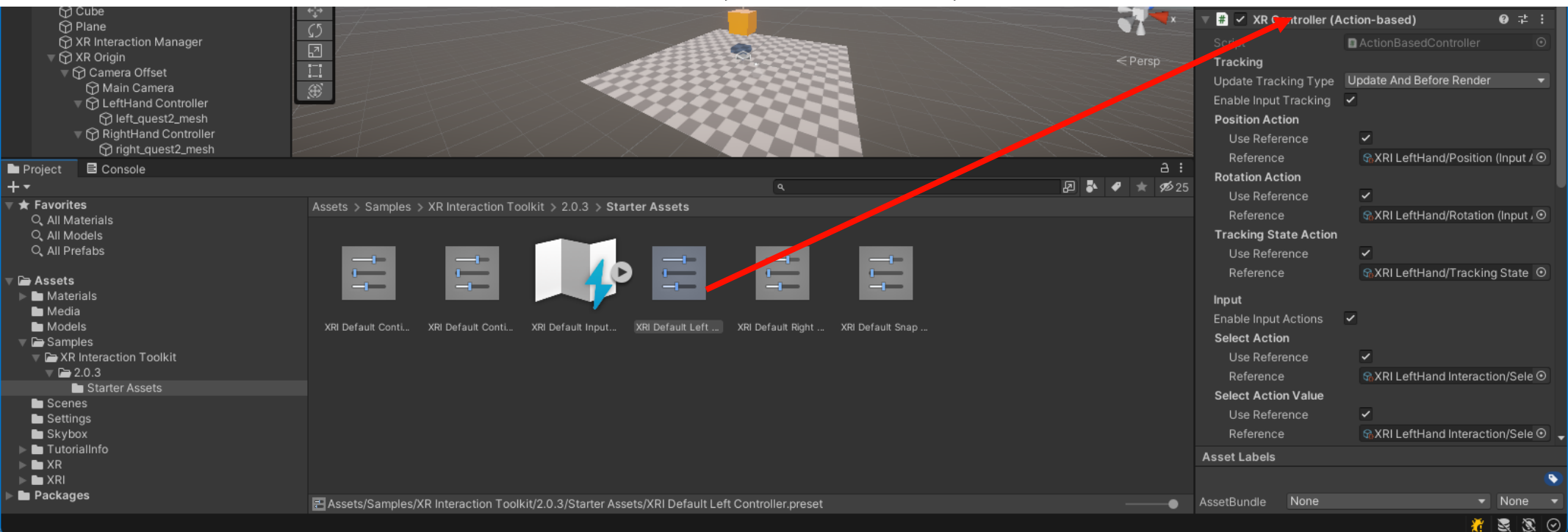


Motion Control の動きを設定

XR Origin || LeftHand Controller 設定定義

XR Origin | Camera Offset | LeftHand Controller

Asset/Samples/XR Interaction Toolkit/2.0.3/Starter Asset/XRI Default Left Controller
を LeftHand Controller の XR Controller (Action-based) にドロップ

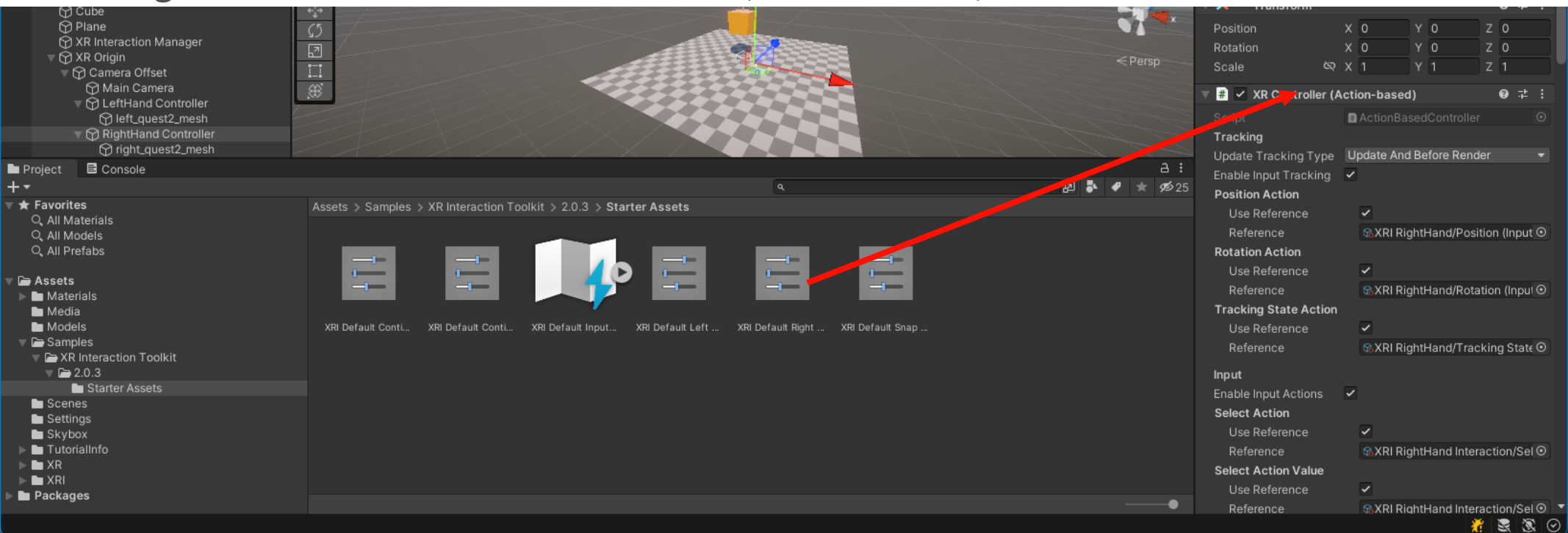


Motion Control の動きを設定

XR Origin || RightHand Controller 設定定義

XR Origin | Camera Offset | RightHand Controller

Asset/Samples/XR Interaction Toolkit/2.0.3/Starter Asset/XRI Default Left Controller
を RightHand Controller の XR Controller (Action-based) にドロップ

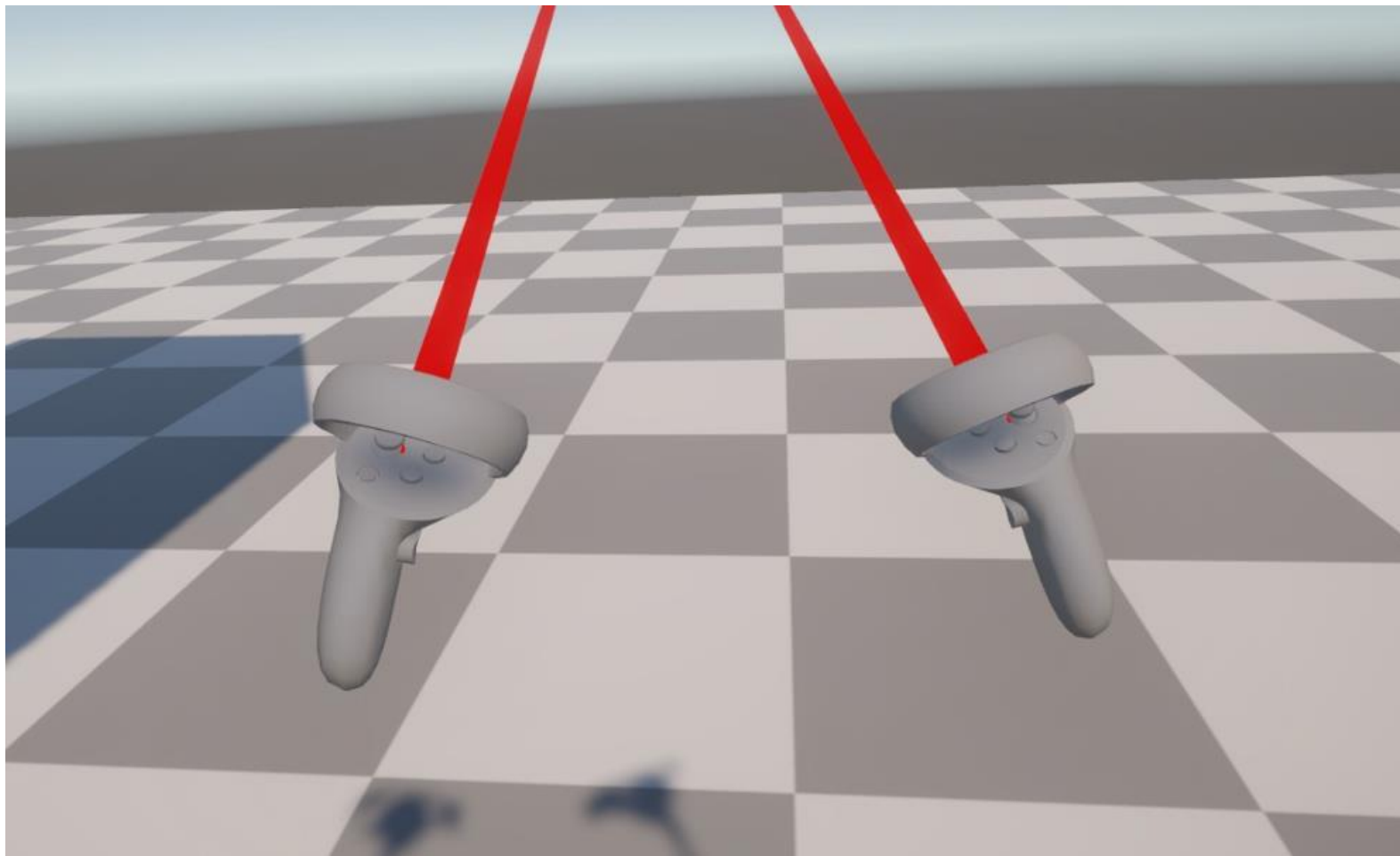


Motion Control の動きを設定

実行テスト

動く

モデル表示



06. 移動

移動システム

テレポーターション移動

Locomotion System : 移動管理システム

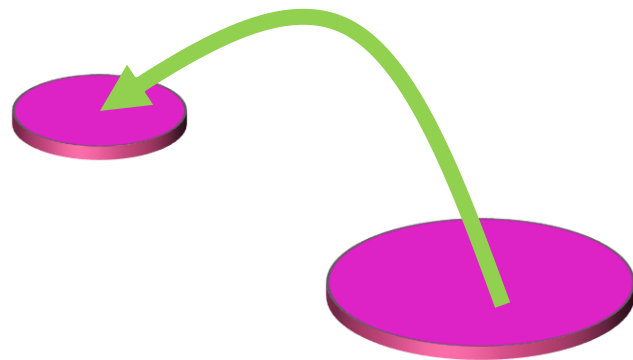
Telepotation Provider : 移動先となるコンポーネント

その他の移動

Continuous Move : スティック操作による平面移動 (酔い注意)

Snap Turn : スティックによる一定角度の回転

Continuous Turn : スティックによる連続回転 (酔い注意)



VR酔い

脳と体の動機ずれ

自分は止まっているのに、視界は移動しているように見える

VR酔いの対策

視野角広く、フレームレートを高くする

移動はテレポートを使う

視界だけが加速度運動をするのを避ける

ユーザーの意志とは異なる移動は極力避ける



テレポーテーション

XR Origin

Locomotion System 追加

XR Origin に **XR Origin** を設定

Teleportation Provider 追加

XR Origin に **XR Origin** を設定

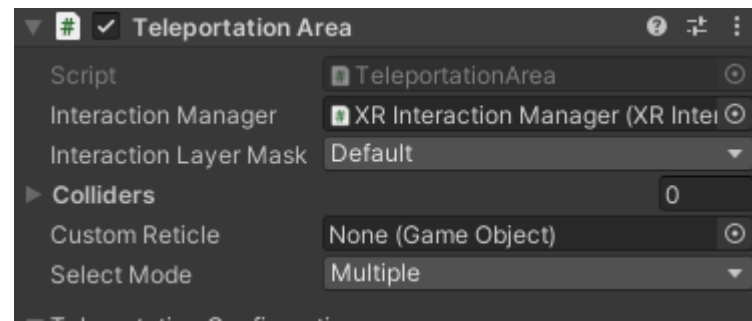
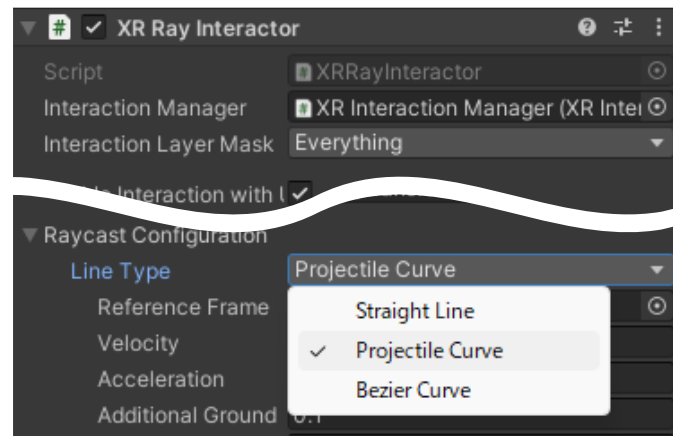
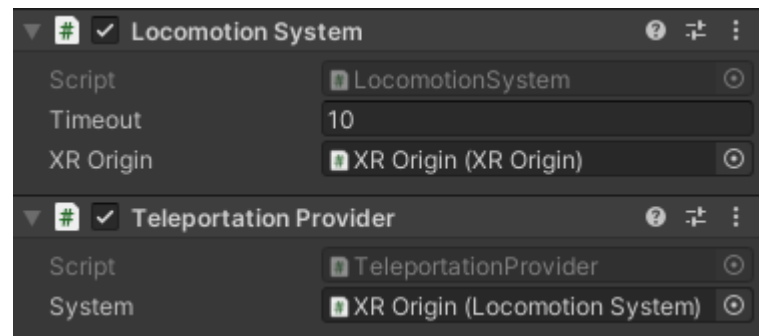
XR Origin || Right Hand Controller

XR Ray Interactor > Raycast Configuration

➤ Line Type > Projectile Curve

Plane

Teleportation Area 追加



連続移動(左ステック)

XR Origin

Continuous Move Provider (Action Based)

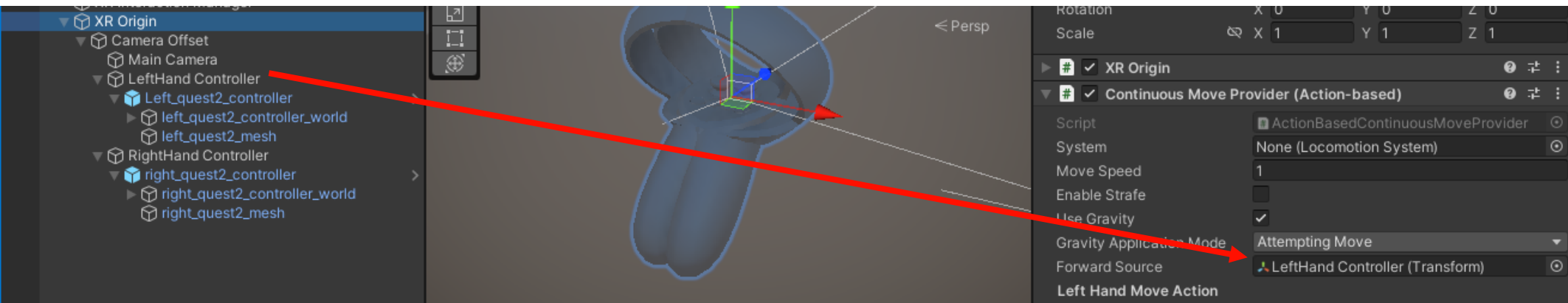
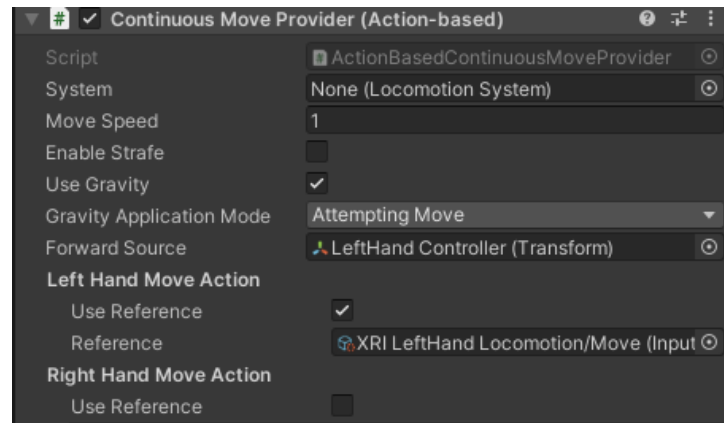
Enable Strafe をアンチェック

Forward Source は LeftHand Contorller をドロップ

Left hand Move Action

User Reference にチェック

Input Action で ◎ を押して Left hand Locomotion / Move



回転(定角: 右 / 連続: 左)

XR Origin

Snap Turn Provider (Action Based)

Right Snap Turn Action

User Reference に **チェック**

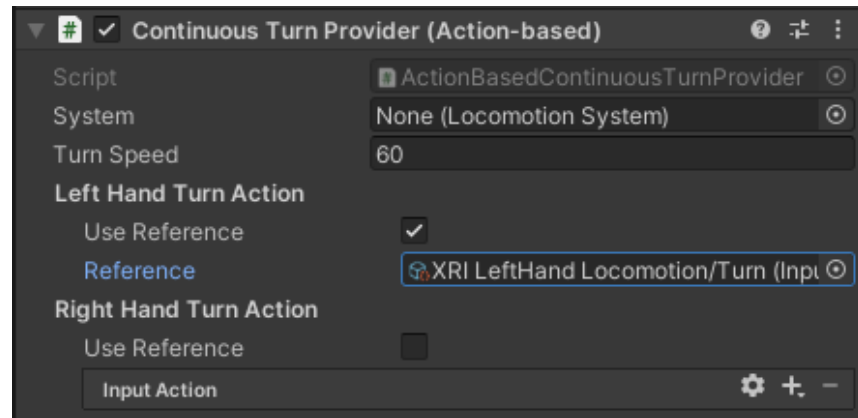
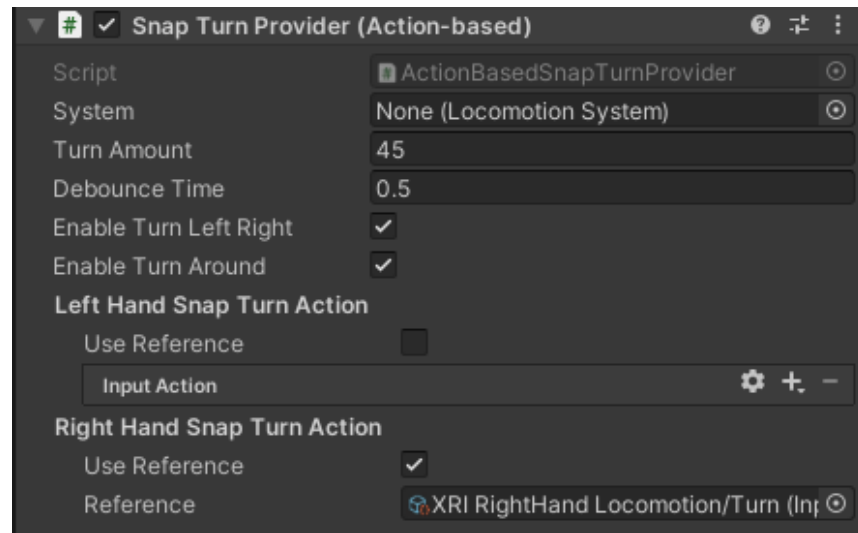
Input Action : Right hand Locomotion / Turn

Continuous Turn Provider (Action Based)

Left Snap Turn Action

User Reference に **チェック**

Input Action : Left hand Locomotion / Turn



テスト&操作方法

テレポート

Ray Cast で指定してGrip ボタン

回転

右スティック左右

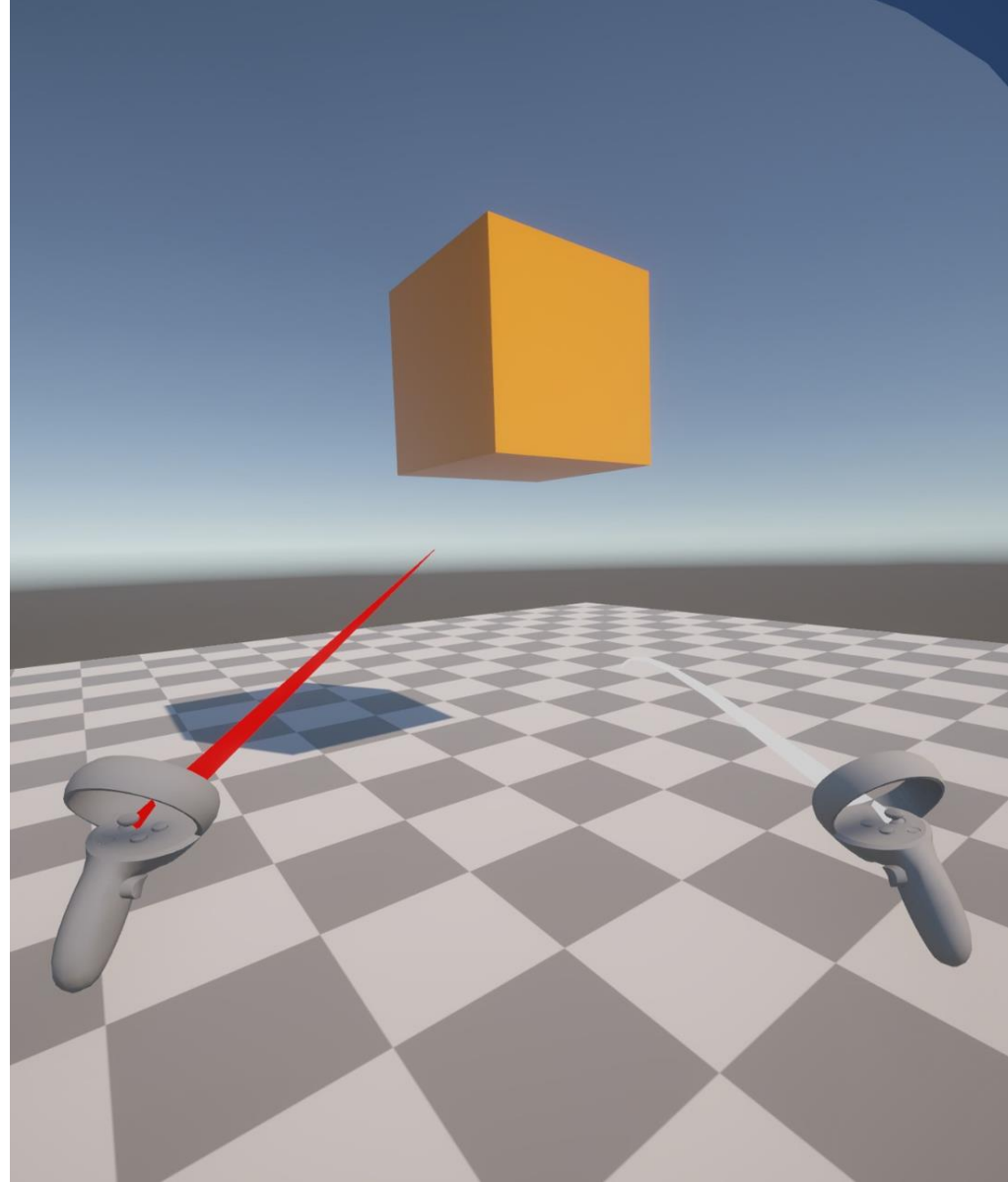
連続移動

左スティック前後

コントローラー方向に移動

連続回転

左スティック左右(酔い注意)



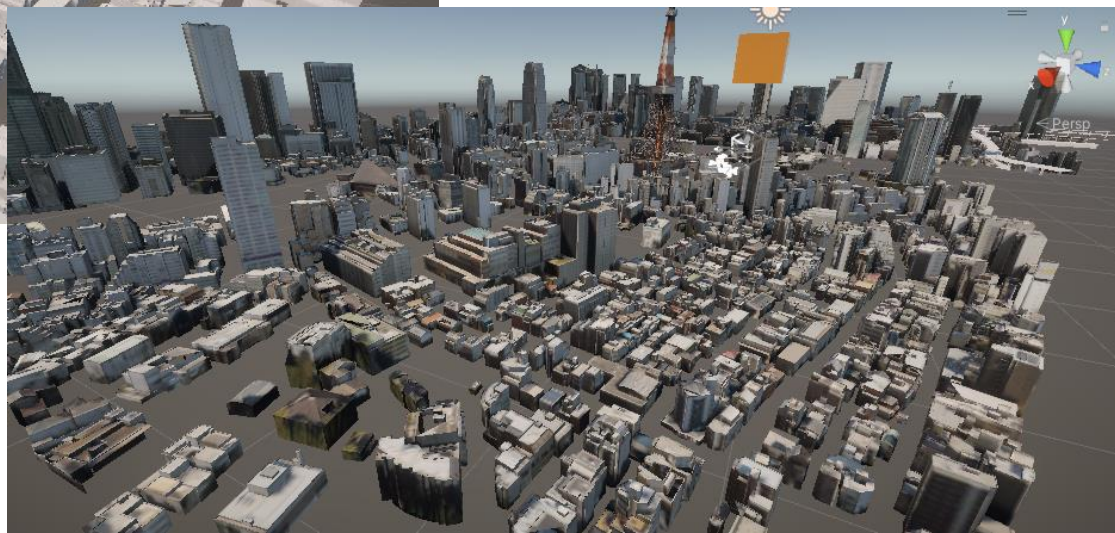
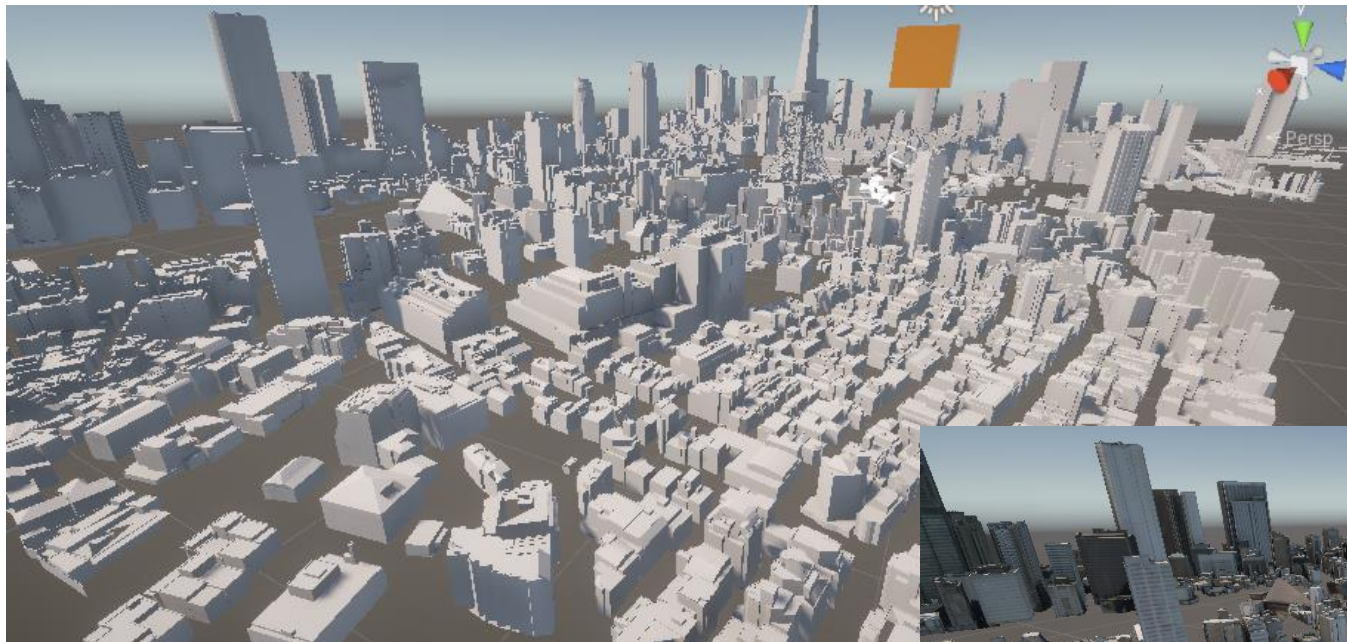
07. PLATEAU設定

Plane のサイズを拡大

TokyoTower

Asset > Model > TokyoTower_bldg./brid を Hierachyにドロップ
2つのファイルを選んで Create Empty Parent

FbxテクスチャのUnity内での再設定



テクスチャデータの設定

TokyoTower

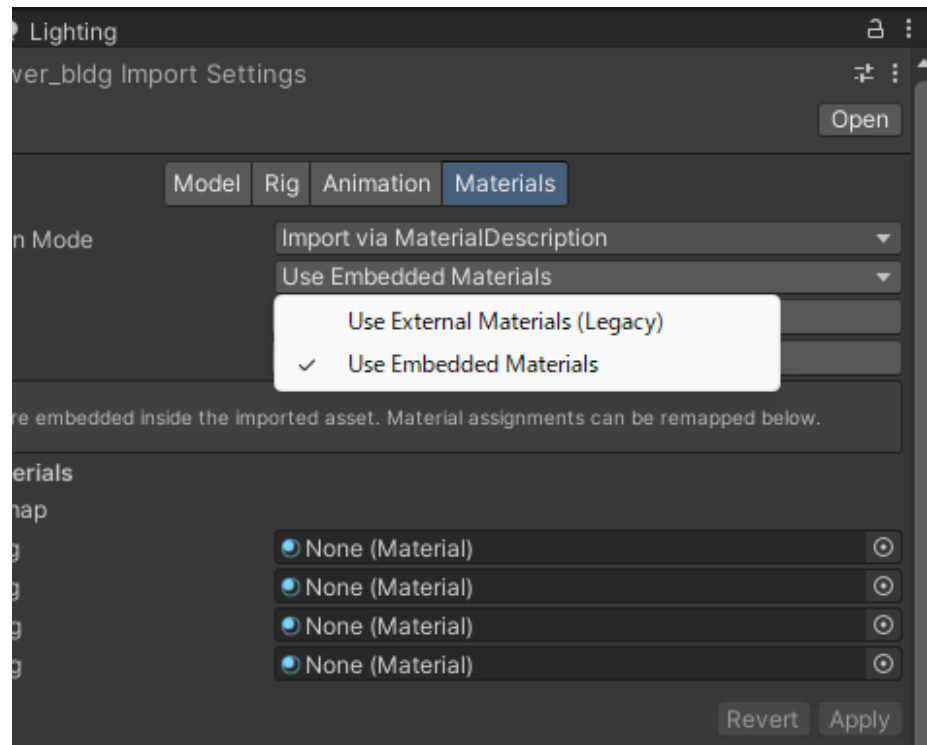
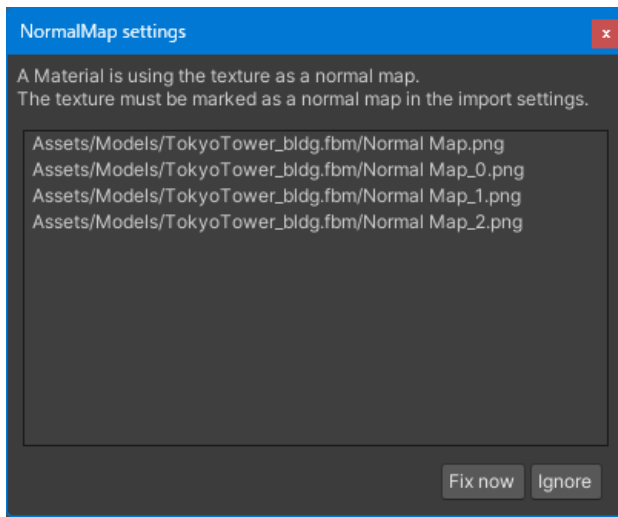
Asset > Model > TokyoTower_bldg を選択

Inspector > Materials

Location を Use External Materials (Legacy) に

Apply ボタン

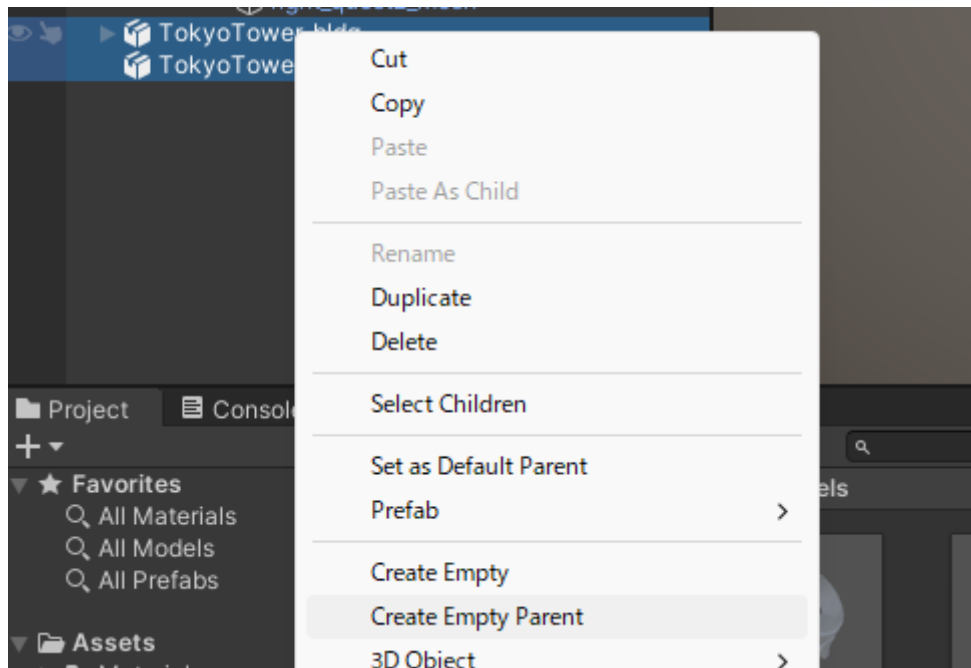
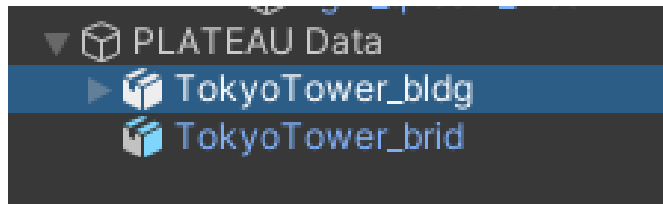
Dialog > Fix now



データ取り込み

TokyoTower

Asset > Model > TokyoTower_bldg./brid を Hierachy にドロップ
2つのファイルを選んで Create Empty Parent
Plateau Data に改名

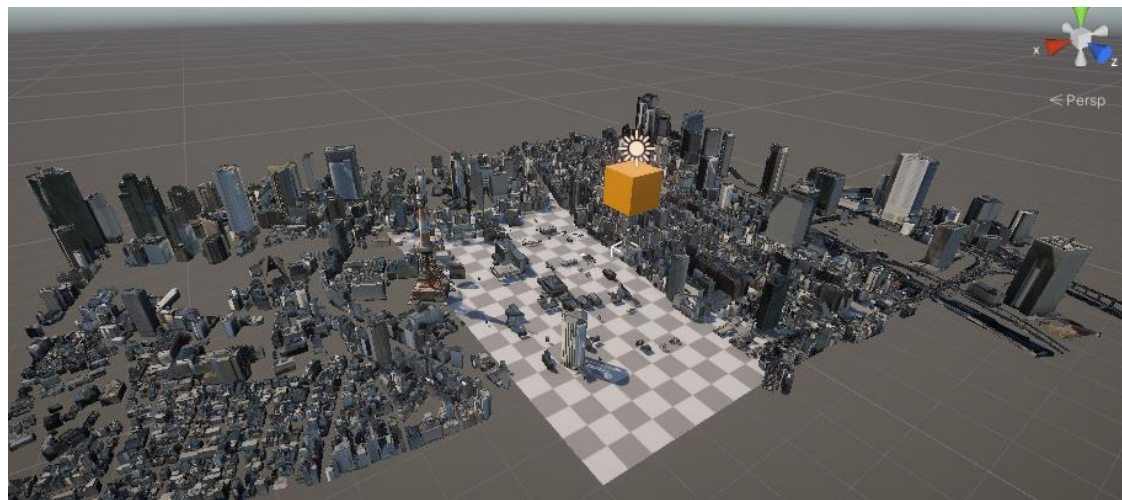
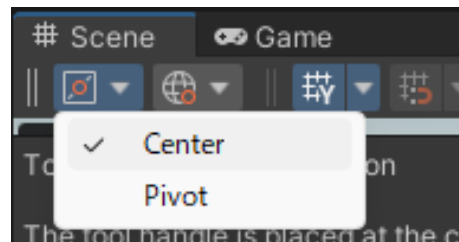


移動

都市データを中心に持ってくる

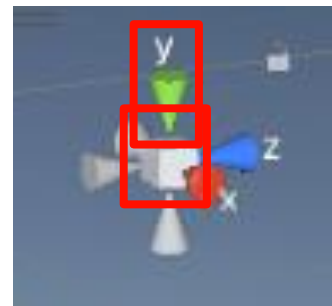
Tool Handle Position を Center に
カメラ位置に都市データを移動させる

1. XR Origin || Main Camera選択
2. Shift-F
3. Plateau Dataを選択
4. Ctrl-Alt-F
5. Transform | Position | Yを0に



Plane の拡大

都市データに合わせてPlane を拡大



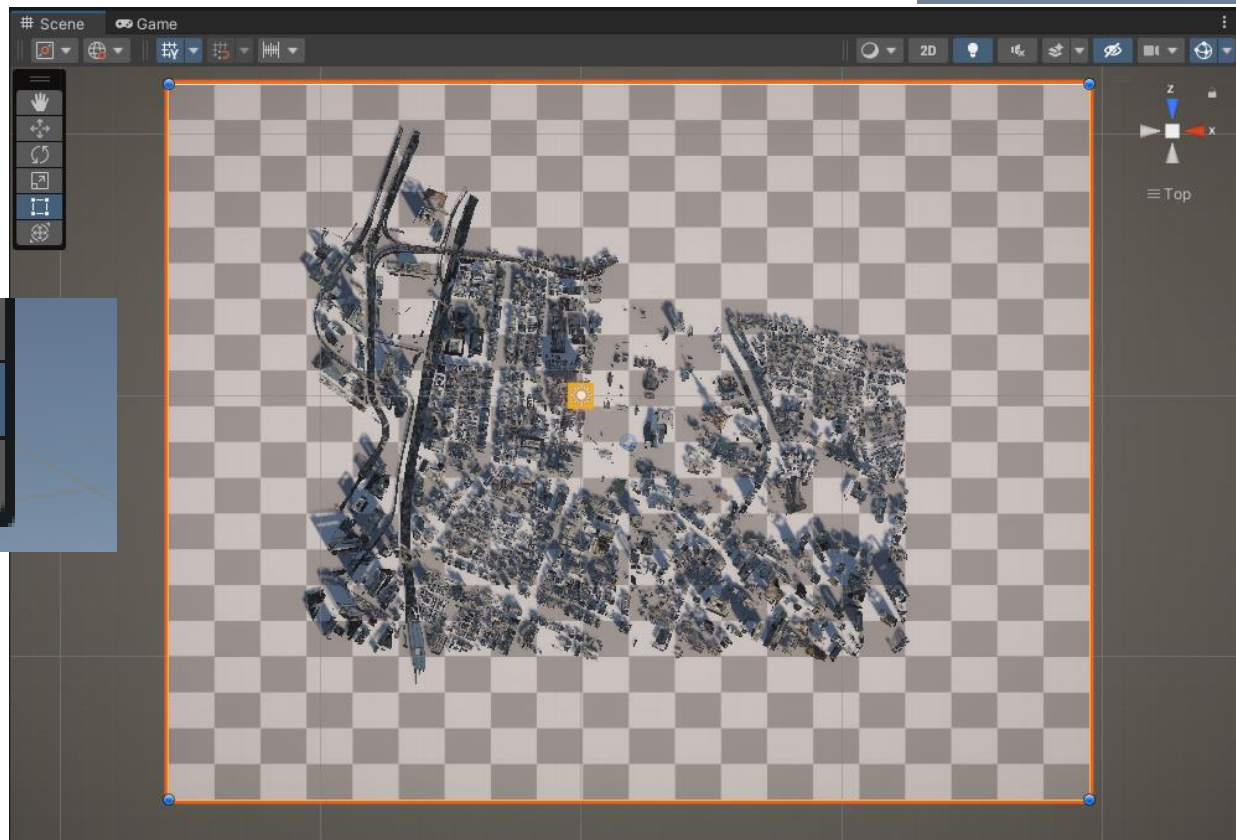
Gizmo

Yを選んで上から俯瞰視線
中心を選んで平行投影に

Tool

RectTool

頂点を持って拡張



地図を貼る(Optional)

Materials > TokyoTower.png

Texture Type

Sprite(2D and UI) に

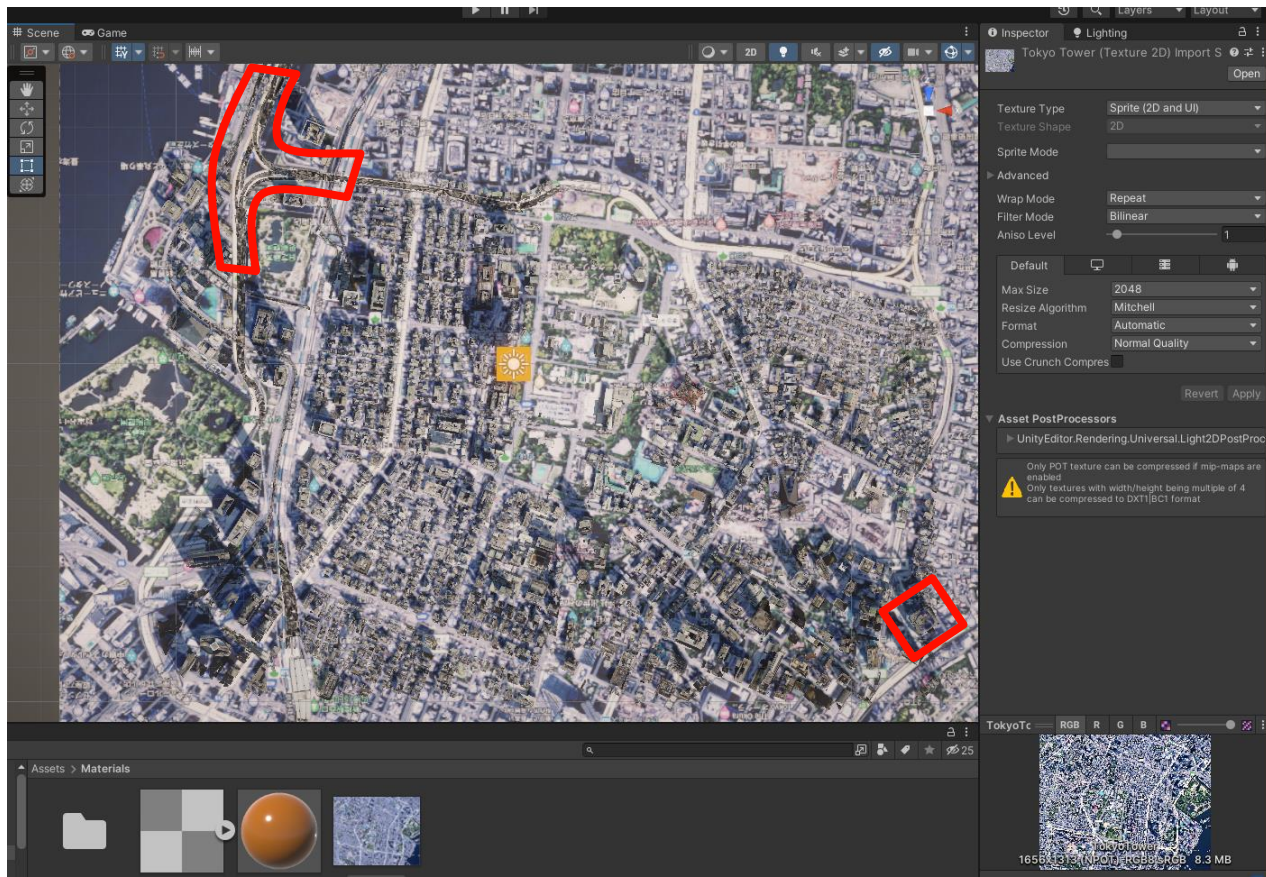
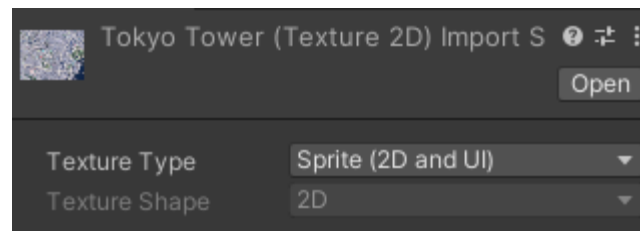
Plane にドロップして貼る

Planeを拡張

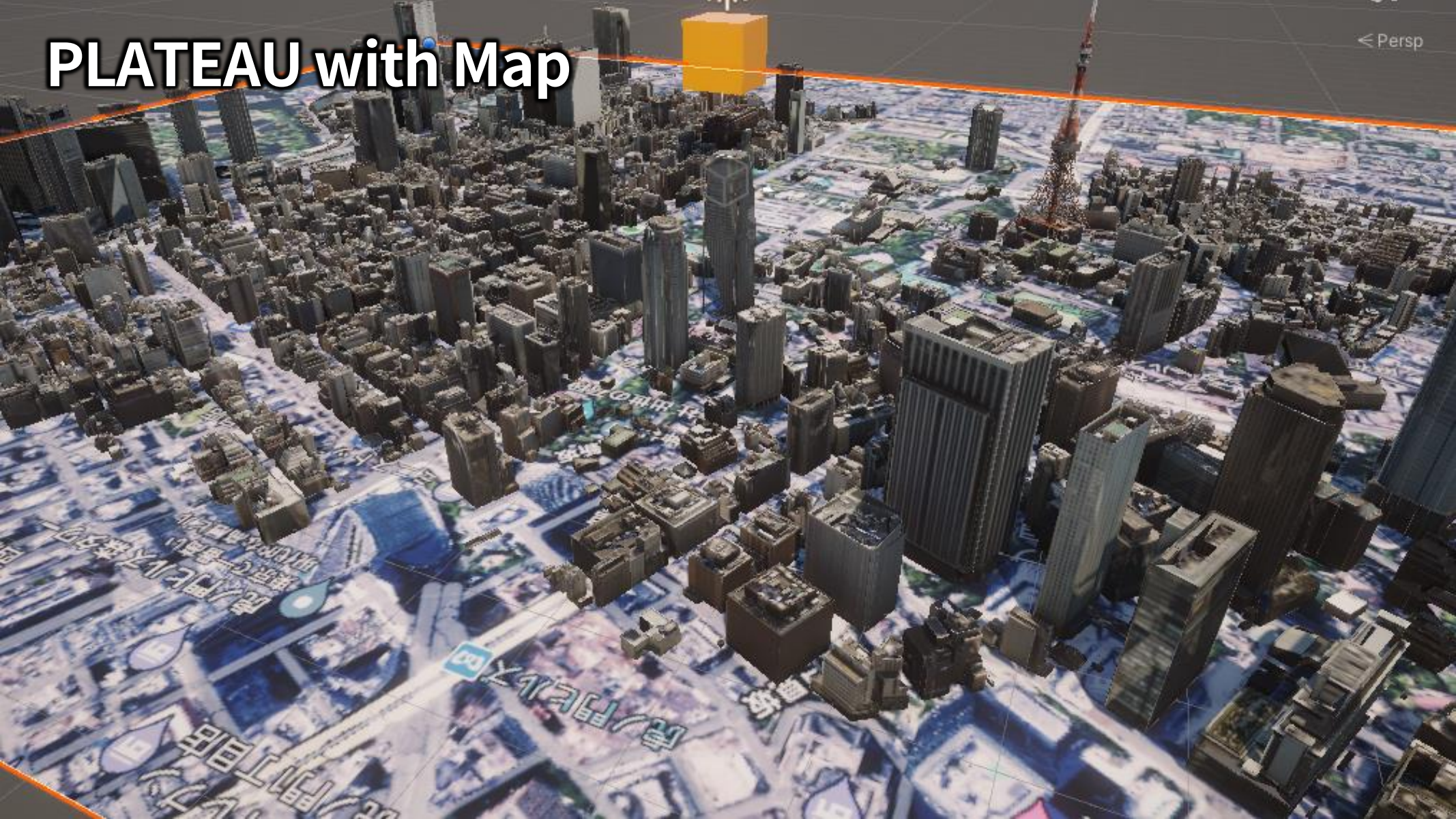
目立つ地上物を目印に

左上の高速道路の形状

右下の建物



PLATEAU with Map



08. 上下移動

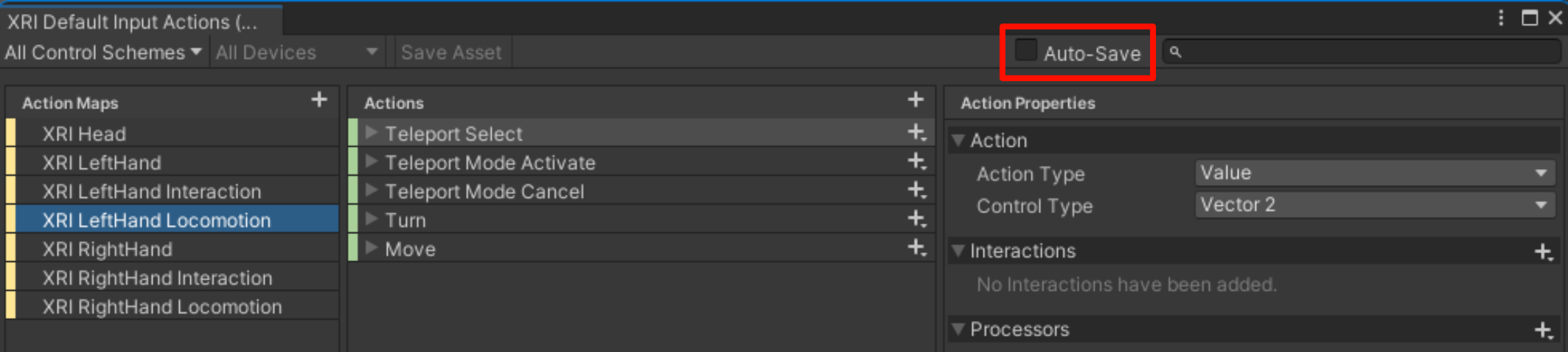
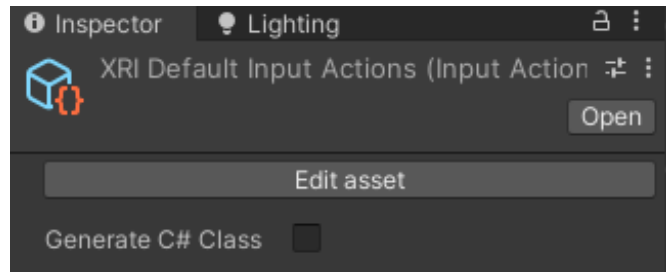
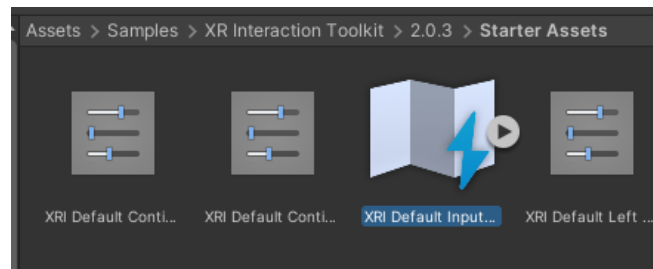
XRI Default Input Actions

Actions 一覧の表示

Asset/Samples/XR Interaction Toolkit/2.0.3/Starter Asset/**XR Default Input Actions** を選択

Inspector から **Edit Asset**

変更を反映するために AutoSave に**チェック**



キーアクションの追加

UP/Down Action の追加と設定

XRI LeftHand Locomotion | Add Action

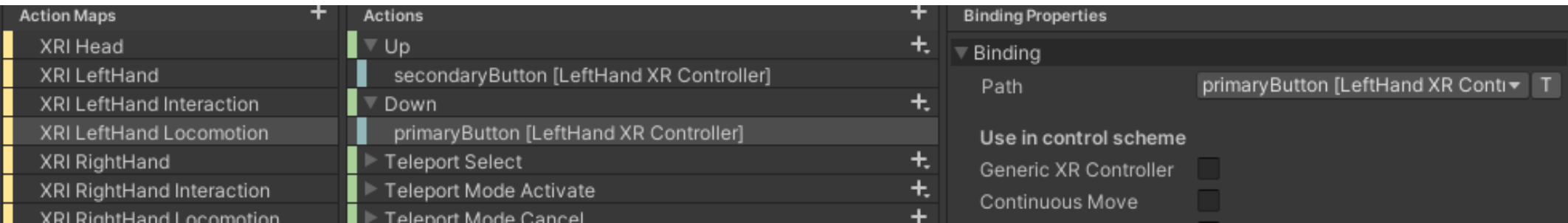
UP と Down を追加

Up の Binding Properties

XR Controller (Lefthand) | Optional Controls | Secondary Button

Down の Binding Properties

XR Controller (Lefthand) | Optional Controls | Primary Button



上下移動用のコード実装

設定したアクションの取得

InputActionReference

イベントの発生設定

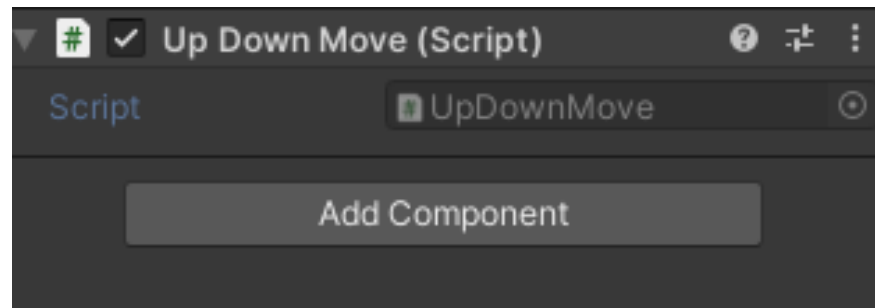
状態 (ボタンが押されている・いない、スティックの位置等) の取得

Action.IsPressed() ボタンが押されているかの状態

XR Origin

XR Origin | Add Component

UpDownMove | New Script | Create & Add



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

```
public class UpDownMove : MonoBehaviour
{
    public float Speed = 1f;
    public InputActionReference up = null;
    public InputActionReference down = null;
```

```
void Update()
{
    if (up.action.IsPressed())
        gameObject.transform.position += new Vector3(0, Speed * Time.deltaTime, 0);
    else if (down.action.IsPressed())
        gameObject.transform.position += new Vector3(0, -Speed * Time.deltaTime, 0);
}
```

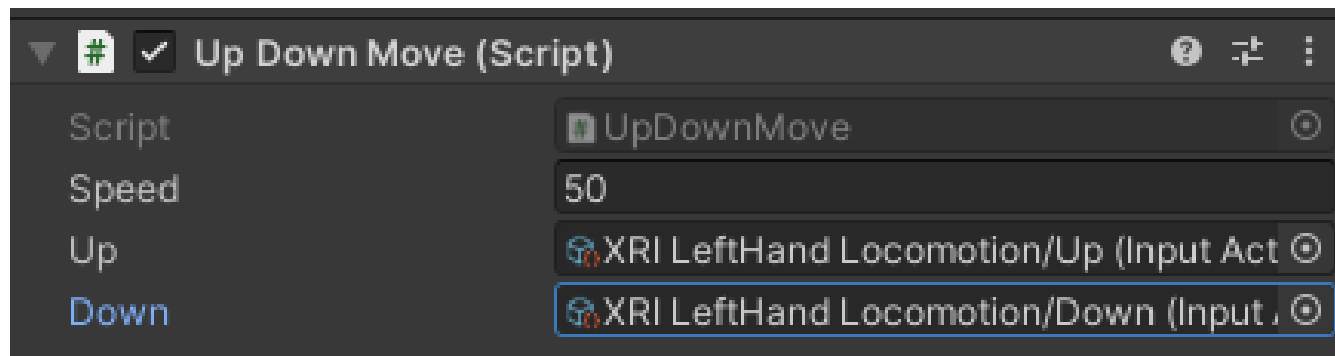
Action とコードの紐づけ

XR Origin

UpDownMove

Up プロパティに XRI LeftHand Locomotion / Up を選択

Down プロパティに XRI LeftHand Locomotion / Down を選択



PLATEAU with Map and Up/Down Move



09. 太陽を動かそう

太陽のシミュレーション

Directional Light

指向性光源

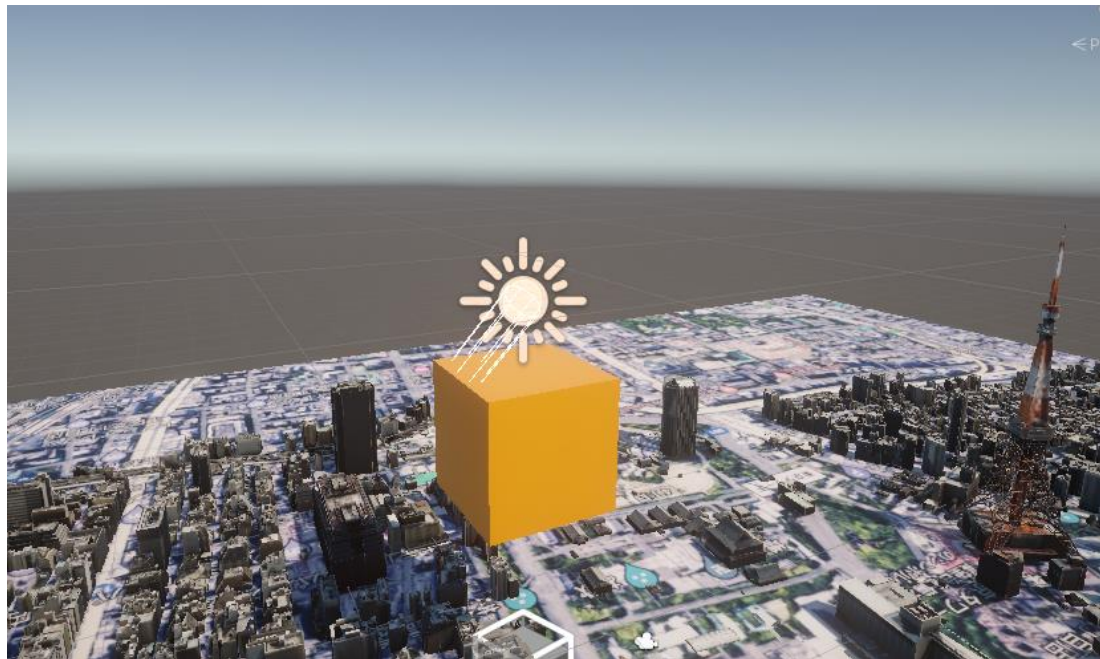
太陽のようなもの

光が指す方角が重要

昼間は(上から)下向きに

機能の実装

Directional Light に RotateTheSun スクリプトを追加



```
using UnityEngine;
```

```
public class RotateTheSun : MonoBehaviour  
{
```

```
    [SerializeField]
```

```
    private float rotateSpeed = 10f;
```

```
    [SerializeField]
```

```
    private Vector3 rot = new Vector3(270f, 330f, 0f);
```

```
    void Update()
```

```
{
```

```
    var rotX = transform.localEulerAngles.x;
```

```
    if (rotX < 0 || rotX > 180)
```

```
        transform.Rotate(-Vector3.right * 10 * rotateSpeed * Time.deltaTime);
```

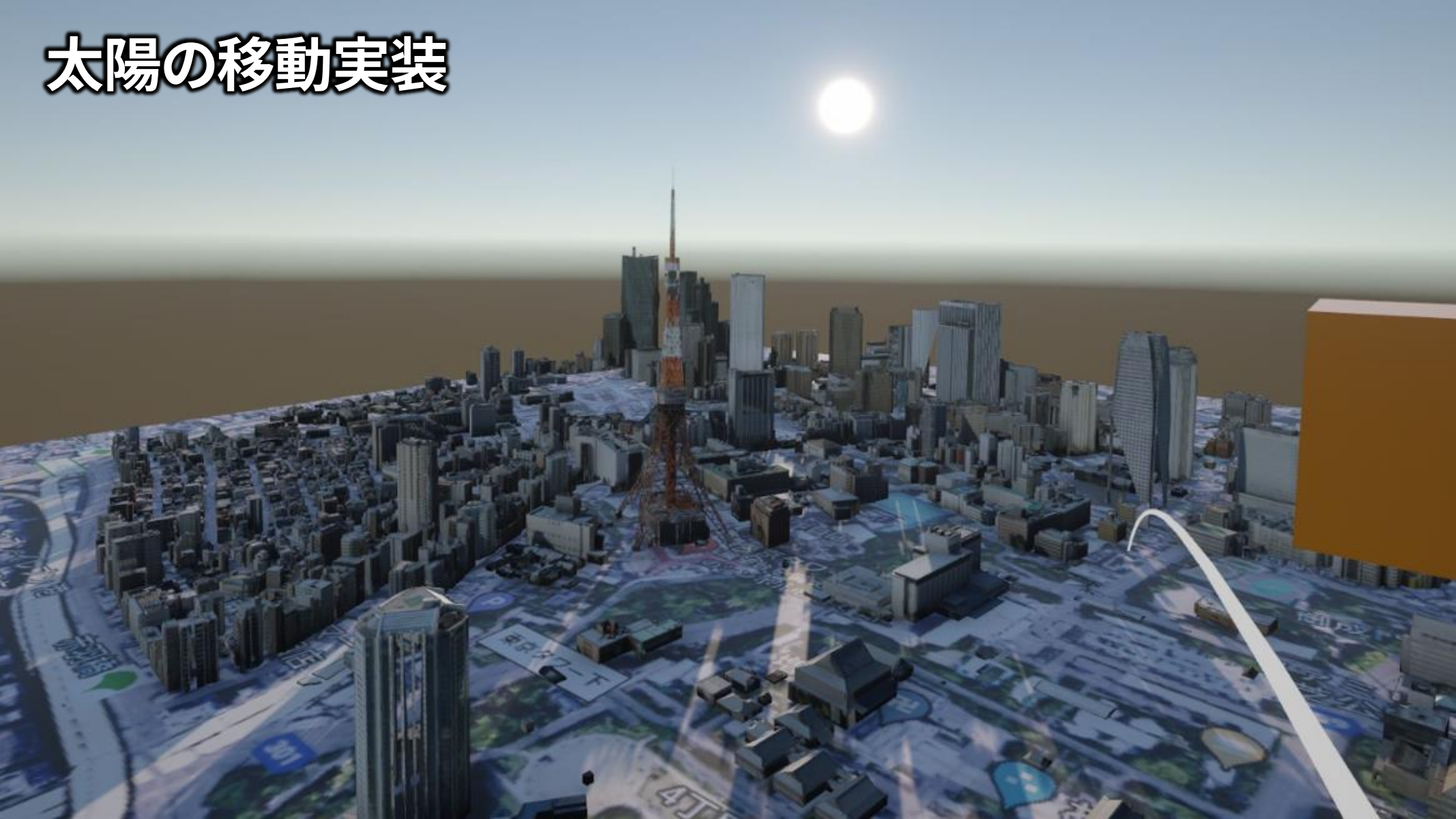
```
    else
```

```
        transform.Rotate(-Vector3.right * rotateSpeed * Time.deltaTime);
```

```
}
```

```
}
```

太陽の移動実装



10. Skyboxの変更

空の景色

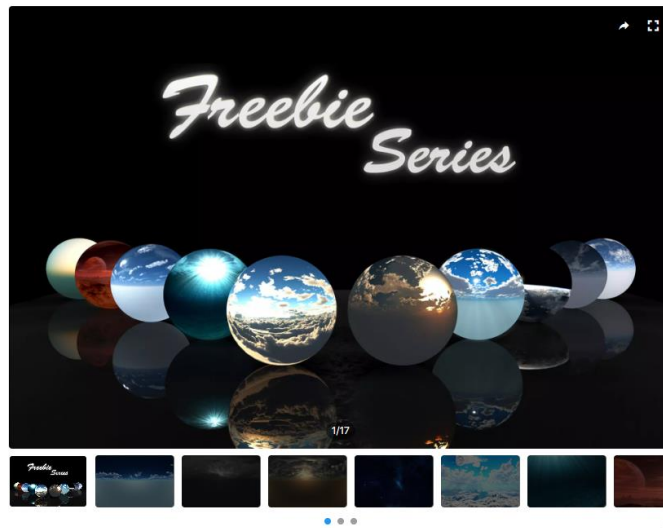
Skybox で変更できる

Window > Rendering > lighting
Environment > Skybox Material

Asset Store で公開

Window > Asset Store > Skybox
Unityにimport

ホーム > 2D > テクスチャ&マテリアル > 空 > Skybox Series Free



Skybox Series Free

Avionx

★★★★☆ (252) | ♥ 18807

FREE

3749 views in the past week

Unityで開く

uotsabchakma

★★★★★ 2日前

A free standard asset in unity assetsstore!

As a free asset, this is hack of a lot! It is one of standard asset for most of my games I make today. I came back here to do this review here.

I've u...

[レビューをさらに読む](#)

使用許諾

[標準Unity Asset Store EULA](#)

ライセンス

[シート](#)

ファイルサイズ

274.9 MB

最新バージョン

4.3

最新リリース日

2022年1月4日

対応する Unity バージョン

2018.4.27 以降

サポート

[サイトを訪問](#)

概要

パッケージの内容

リリース

レビュー

Publisher info

アセットの品質

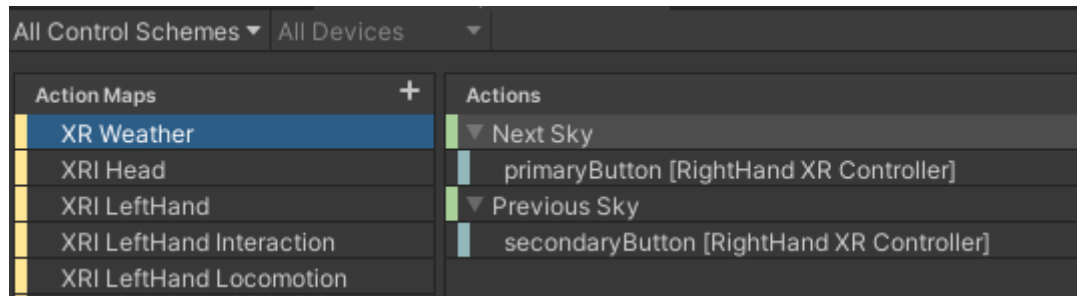
Skybox 変更機能の追加

Input Action

XR Weather 追加

Next Sky | Primary Button (RightHand)

Previous Sky | Secondary Button (RightHand)



Hierarchy

GameObject を追加

Create Empty

ChangeSky スクリプトの追加作成

```
using UnityEngine;
using UnityEngine.InputSystem;

public class ChangeSky : MonoBehaviour
{
    public InputActionReference next = null;
    public InputActionReference previous = null;
    public Material[] sky;
    int num = 0;

    private void ChanageSky(int count)
    {
        num = num + count;
        if (num >= sky.Length)
            num = 0;
        else if (num < 0)
            num = sky.Length - 1;
        RenderSettings.skybox = sky[num];
    }
}
```

```
void Awake()  
{  
    next.action.performed += NextSky;  
    previous.action.performed += PreviousSky;  
    // += をおしてから Tab, Tab でイベントハンドラが自動作成。メソッド名だけ修正  
}
```

```
private void PreviousSky(InputAction.CallbackContext obj)  
{  
    ChanageSky(-1);  
}
```

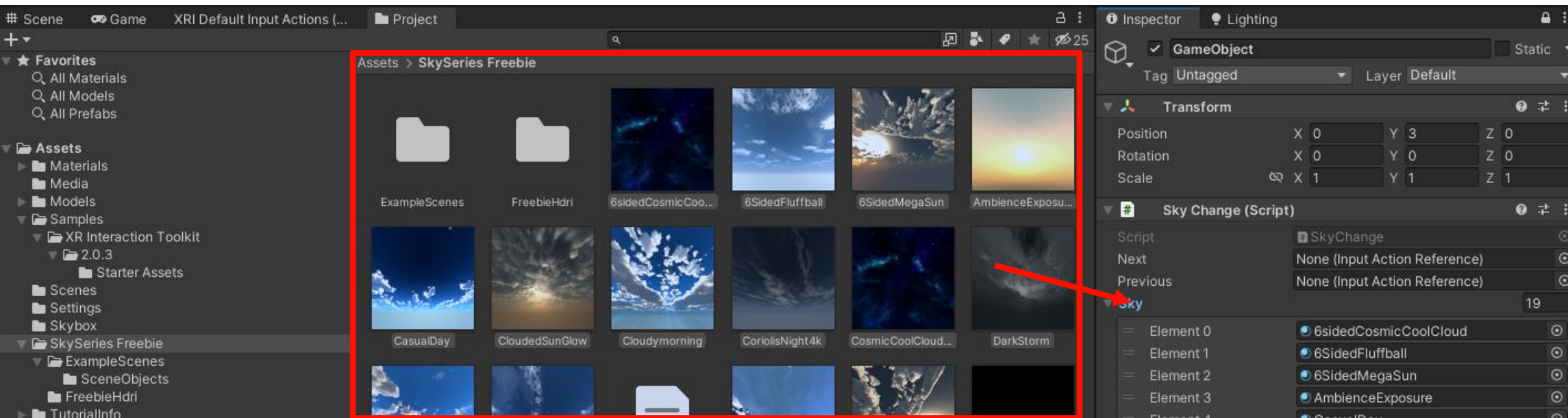
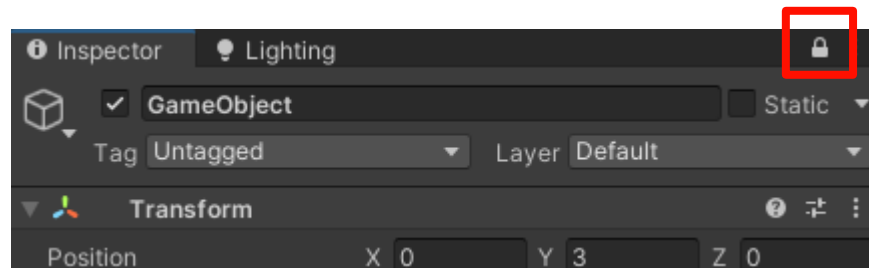
```
private void NextSky(InputAction.CallbackContext obj)  
{  
    ChanageSky(1);  
}  
}
```

Skybox 変更機能の追加

設定の前に

Inspector をロック (作業後にロック解除を忘れずに)

Asset > Sky Series Freebie 以下のSkybox を、Sky Change の Sky に
ドロップする

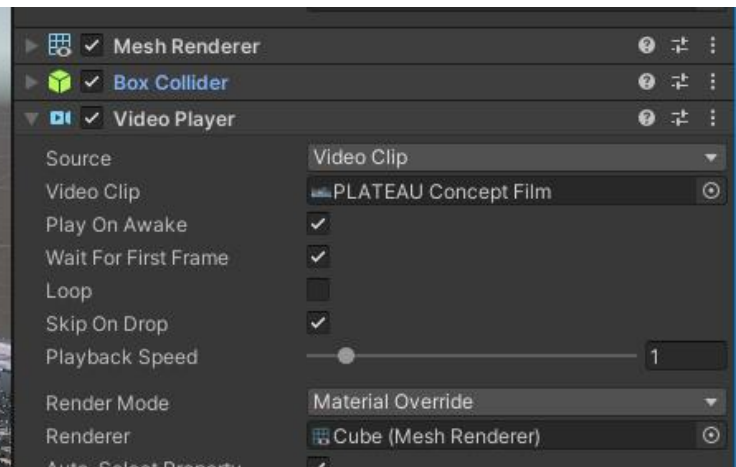
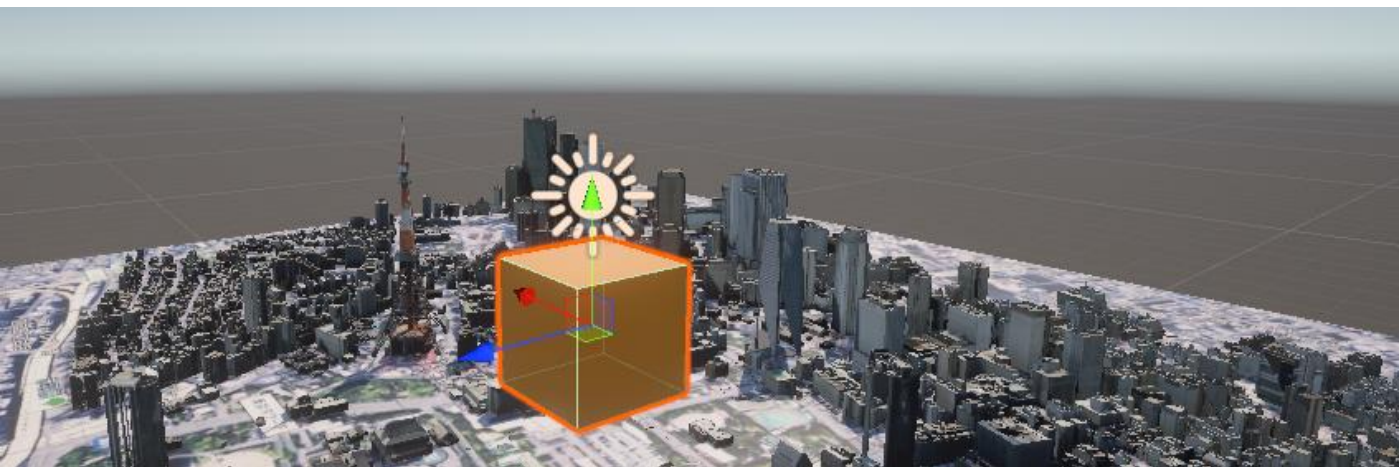


11. 動画再生

動画再生

Video Player

モデル(Plane, Cube等)に貼り付けるコンポーネント
ローカル動画、ストリーミング等が再生可能



The background is a dark, textured field of fine particles. A bright, glowing purple trail of particles curves from the upper left towards the center. Another trail of green particles curves from the upper right towards the center. The overall effect is a sense of dynamic movement and energy.

まとめ

Unity Plateau

VR環境

XR Plugin Management でデバイスと連携

XR Interaction Toolkit で機能を提供

XR Interaction Toolkit

XR Origin が自分自身

Locomotion System 等で移動を実現

New Input System を使ったコントローラーとの連携

Unity Programming

オブジェクト + 機能追加 + 操作との連携

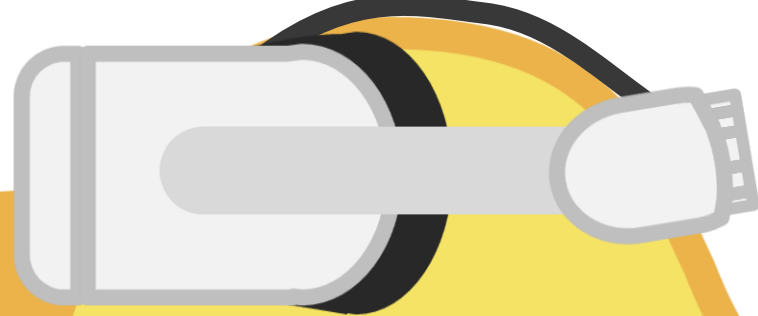


← Persp

Unity で PLATEAUを

Unity ならPLATEAUのデータを直感的に
扱ってアプリを作ることができます。

Unityを使ってPLATEAUの都市モデルを
自由に楽しもう!



Let's PLATEAU
with
Unity