

UNITYゲーム開発

角田研究室

奥崎 弥

自己紹介

名前：奥崎 弥（おくざき ひろ）

大学：青森大学 ソフトウェア情報学部 3年生

趣味：旅行

好きなアーティスト：TOMORROW X TOGETHER

Unityを使ってみよう

- Unityの概要



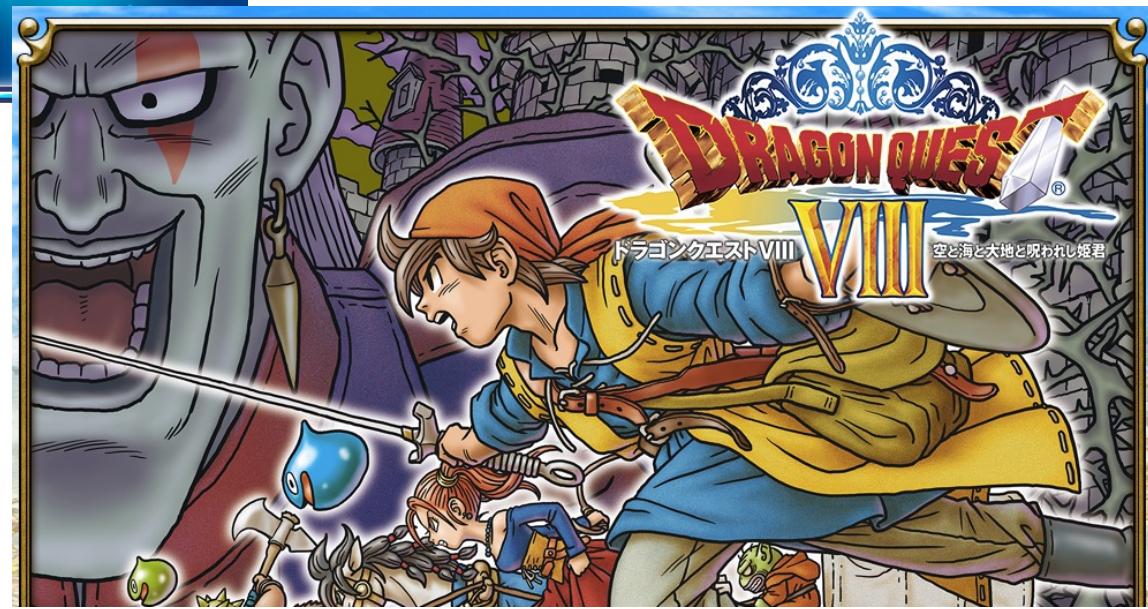
Unityを使っているゲーム







まだ見ぬ世界へ

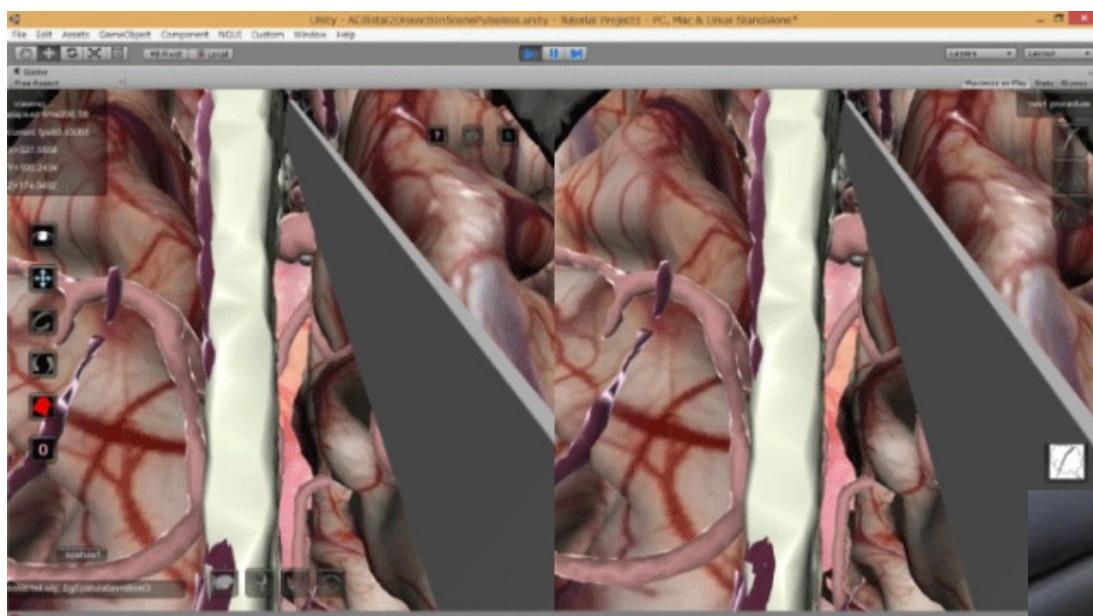


活用事例

- ・メタバース/VRサービス：
世界最大のソーシャルメタバース[VRChat](#)
- ・医療業界：東京大学のVRでの外科手術のシミュレーション
- ・建設業界：BIMデータを活用した3Dモデルの構築

メタバース/VRサービス





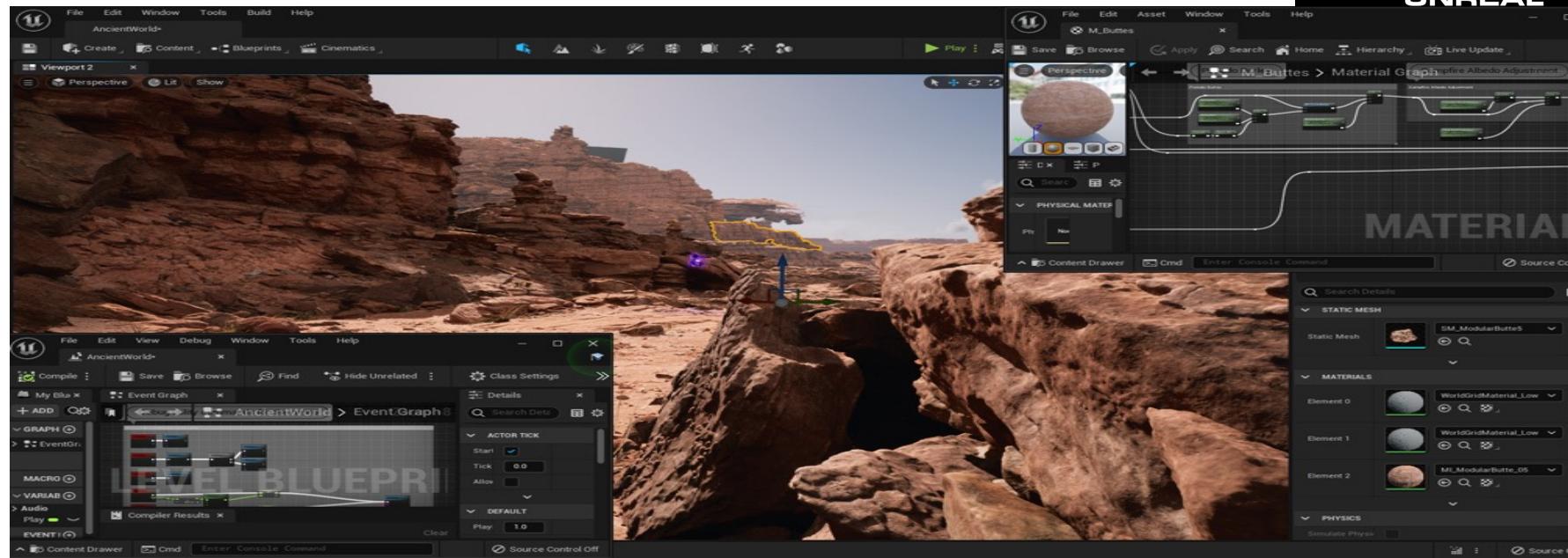
医療業界

建設業界



Unreal Engine (アンリアル・エンジン)

- ・ゲームエンジントップクラスのリアルなグラフィック表現



比較

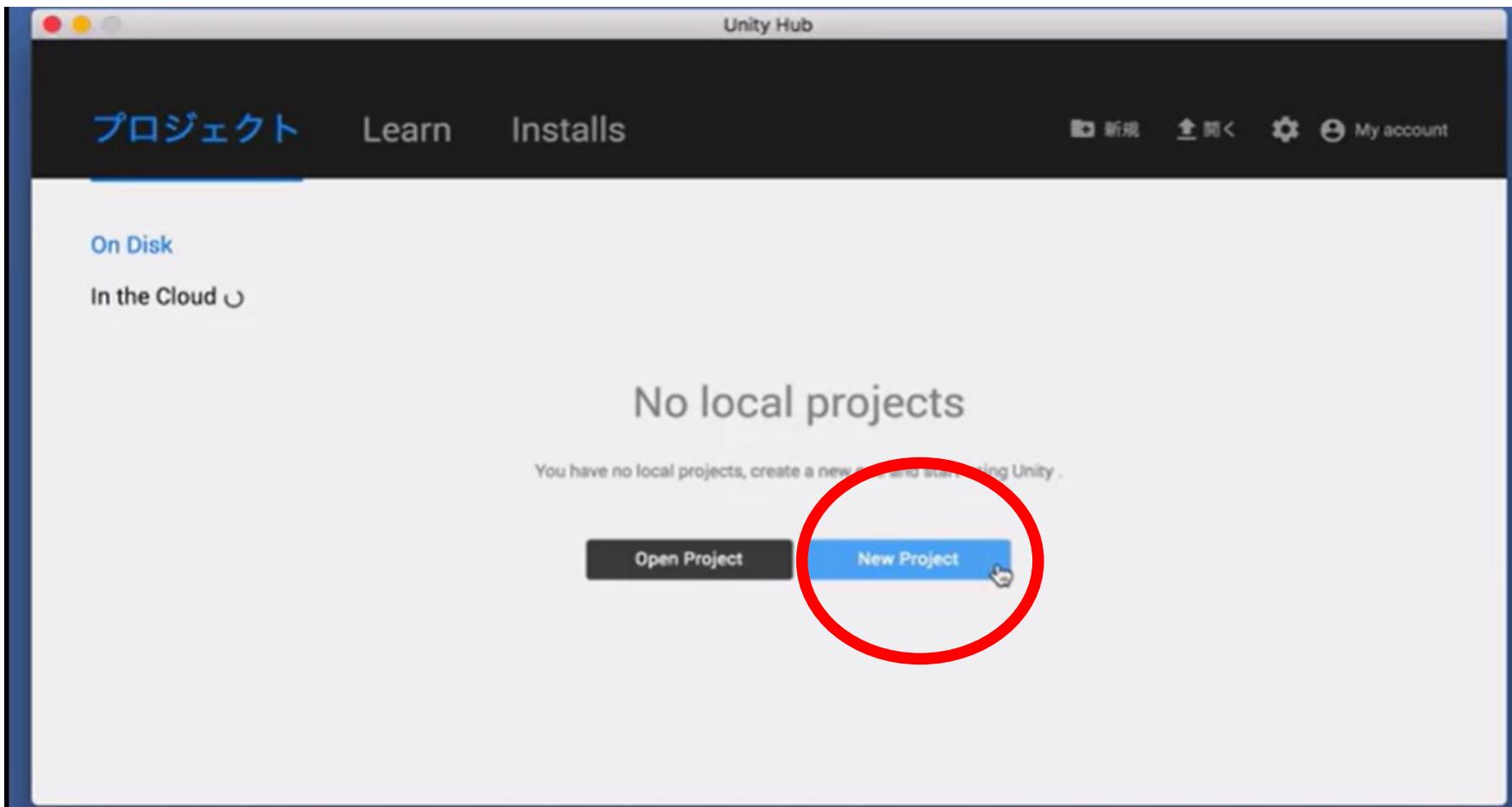
モバイルARゲームや2Dゲームやモバイル・スマホ
ゲームアプリ
→ 「Unity」

PCやコンシューマ向けゲーム
→ 「Unreal Engine」

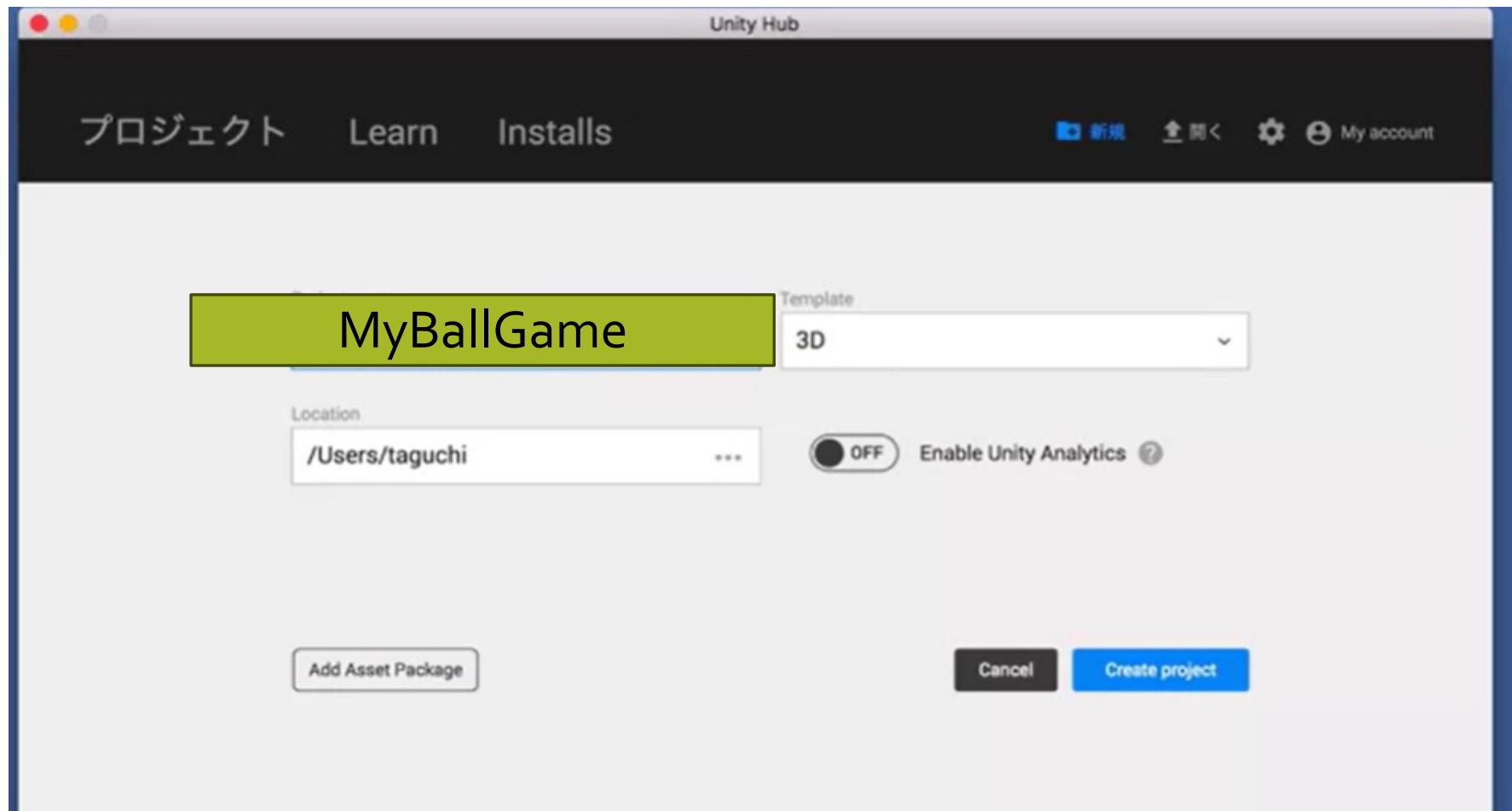
プロジェクトを作ろう

- ・プロジェクトの作成

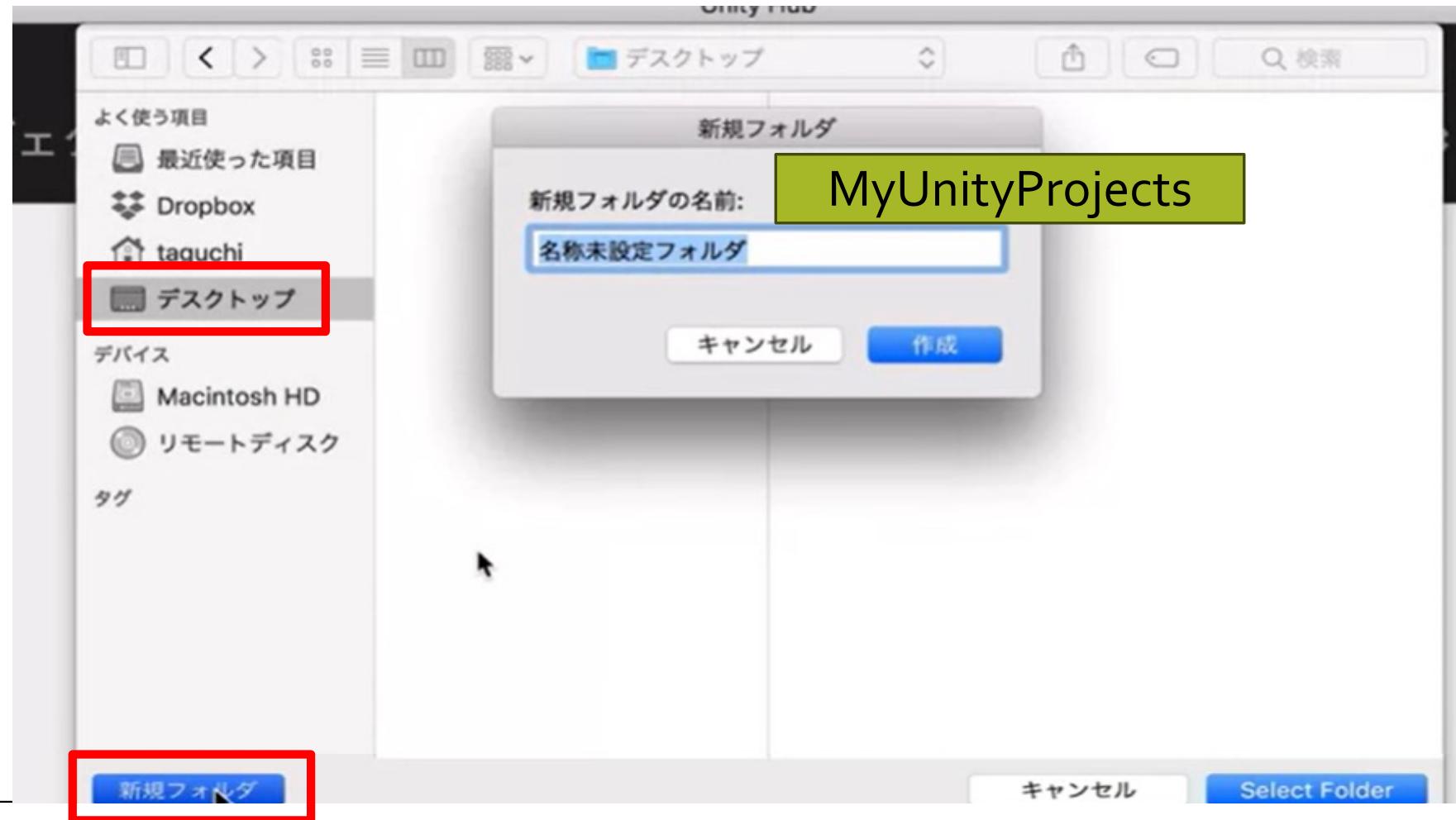
プロジェクトの作成



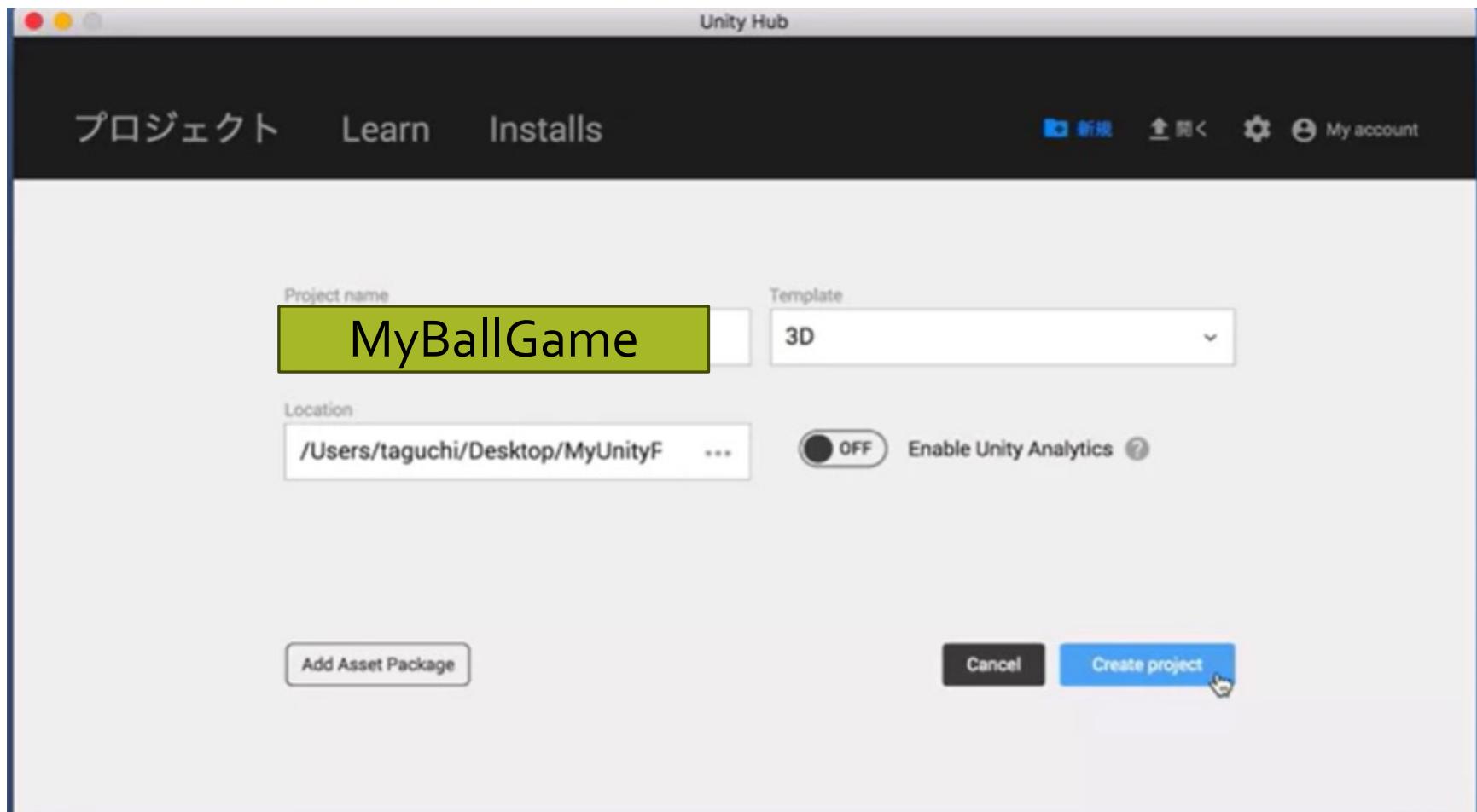
プロジェクトの作成



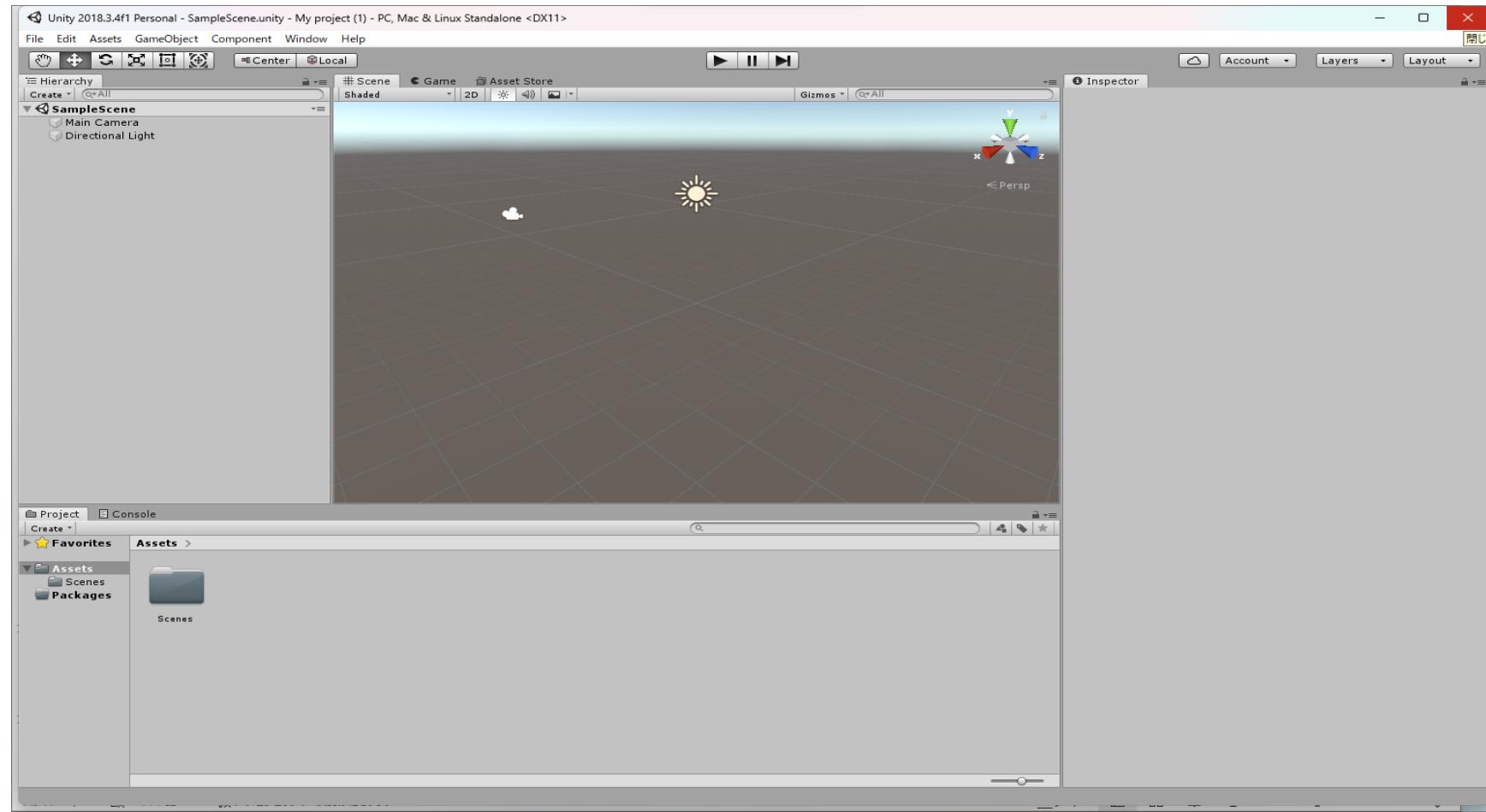
プロジェクトの作成



プロジェクトの作成



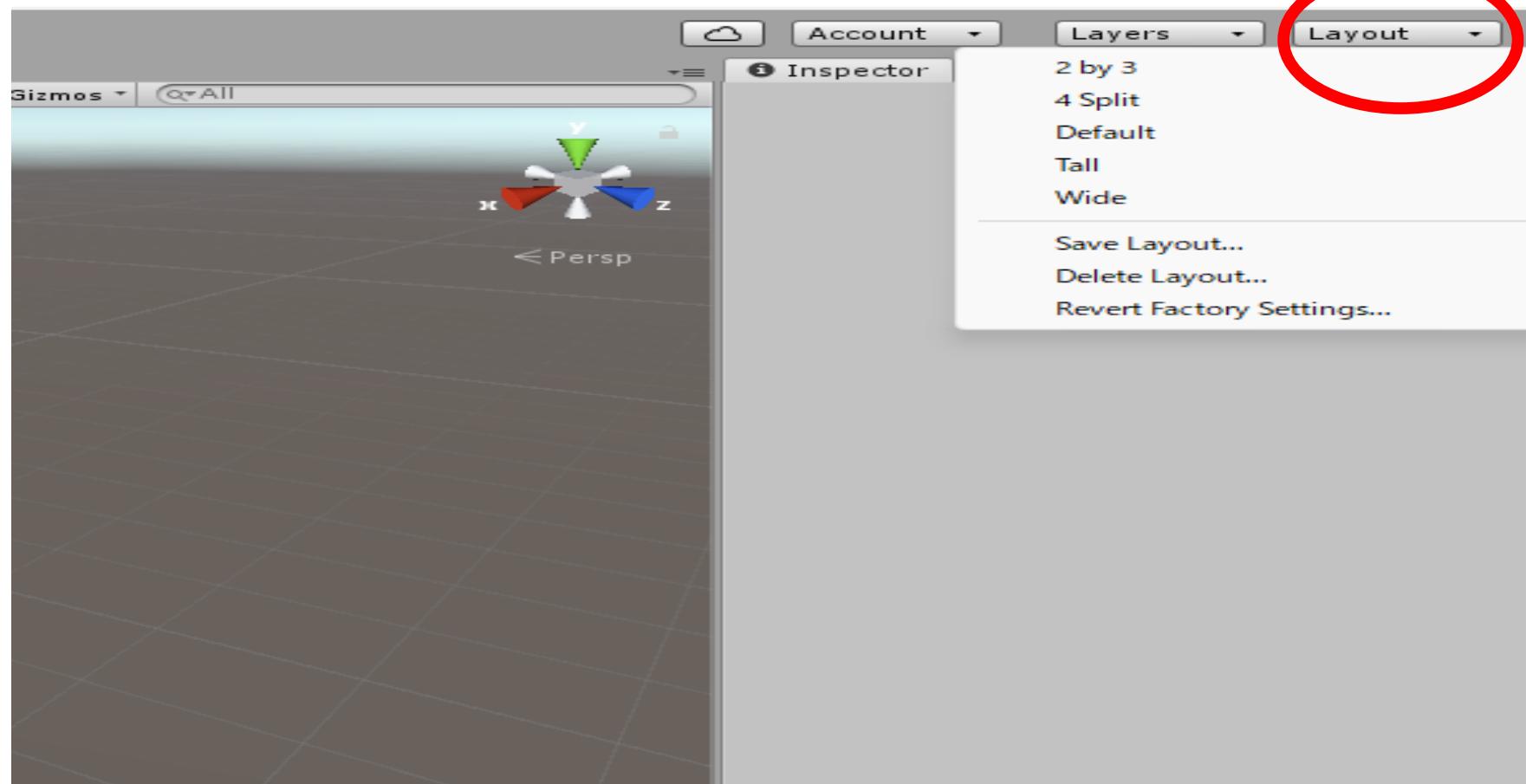
プロジェクトの作成



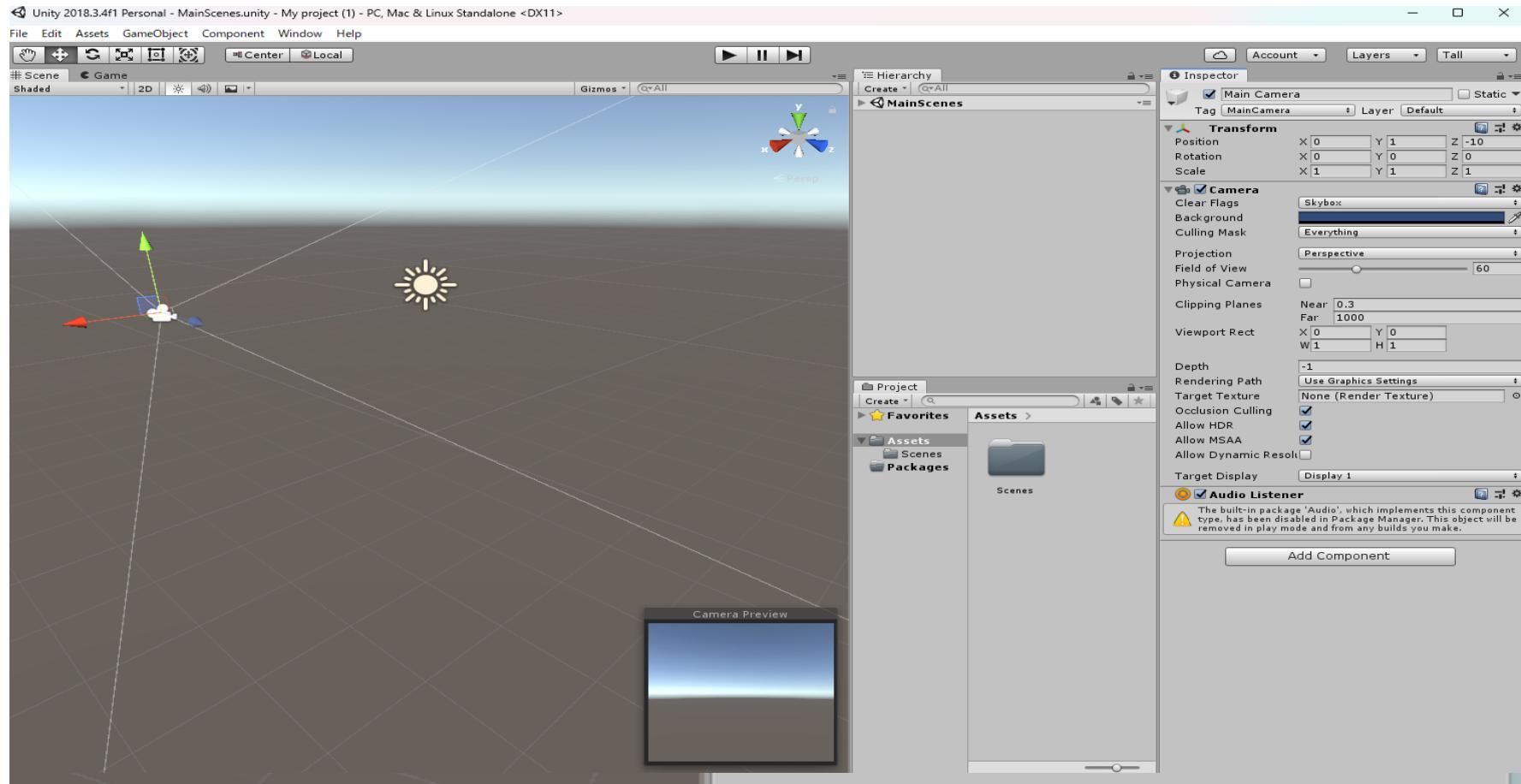
画面の見方

- ・レイアウトの変更
- ・画面の見方

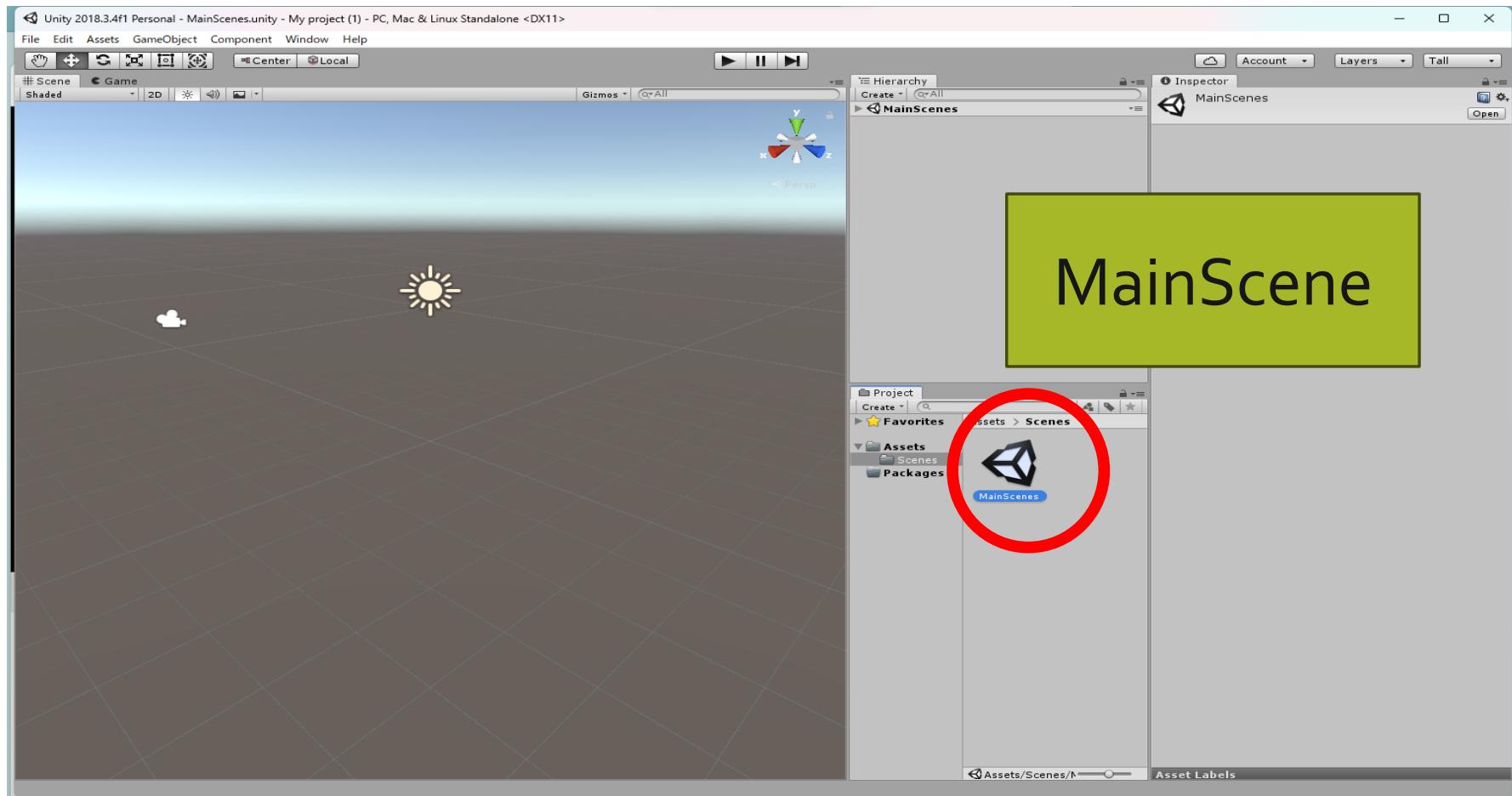
レイアウトの変更



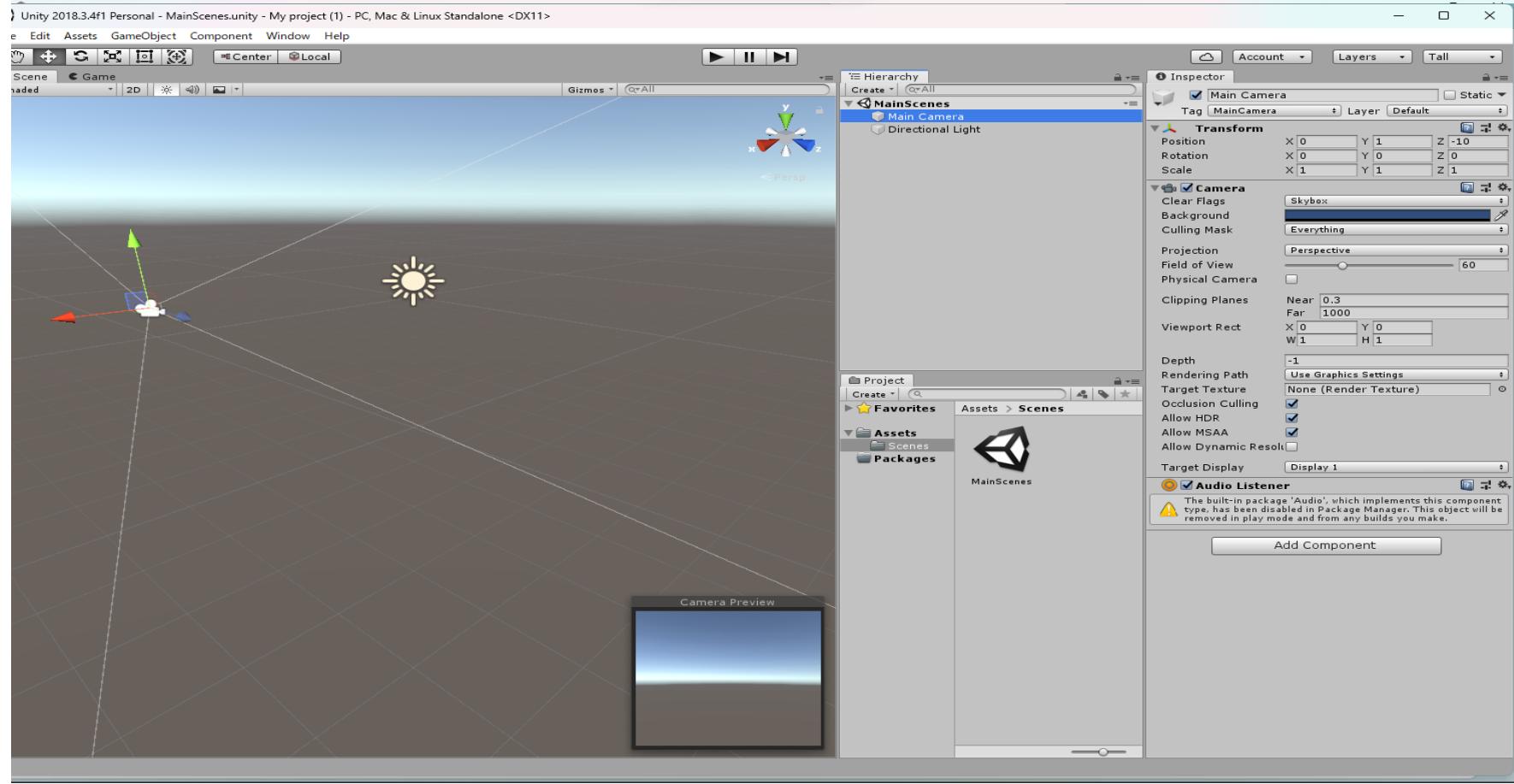
レイアウトの変更



画面の見方



画面の見方

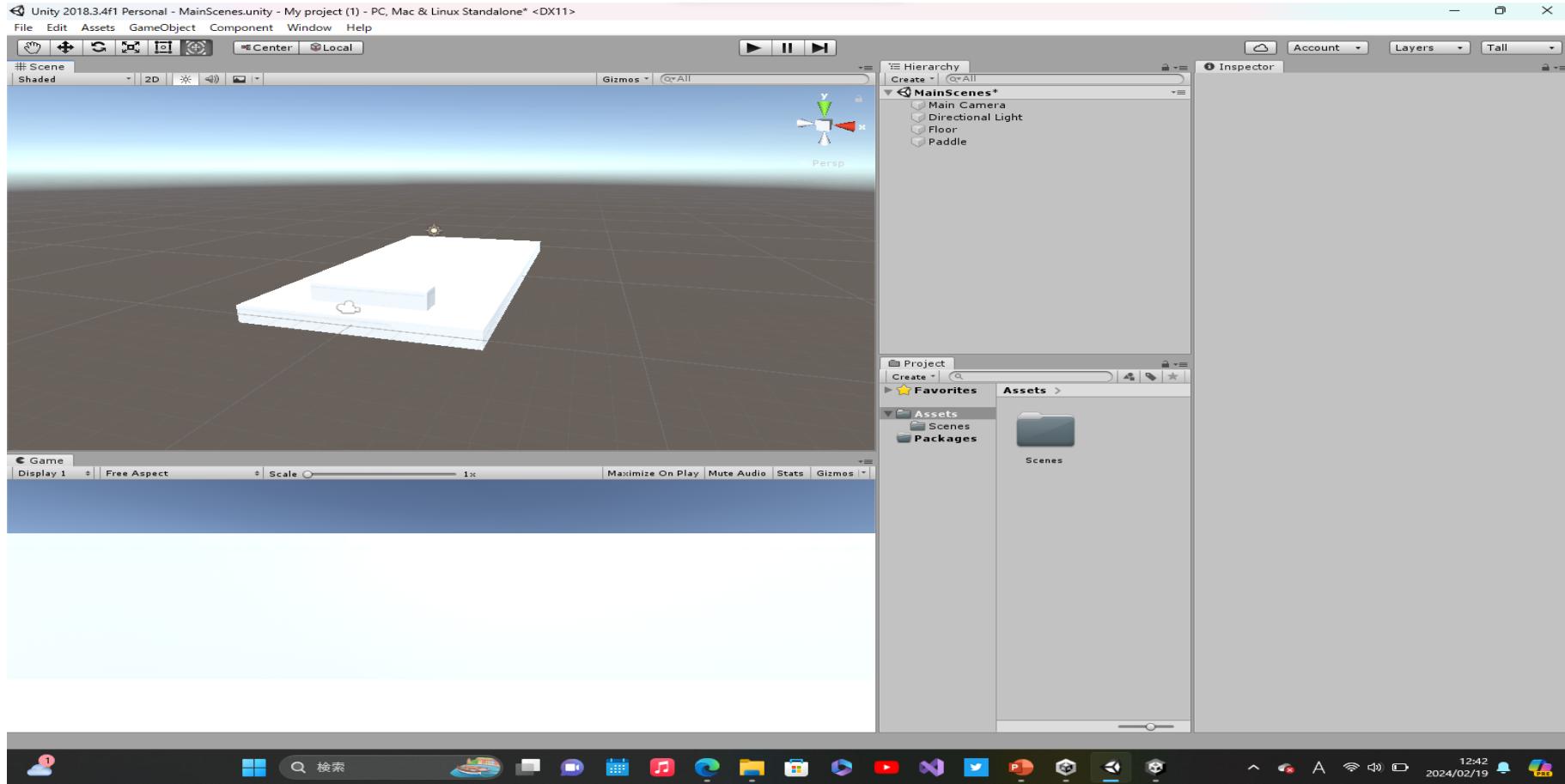


床とパドルを配置していこう

- Floorの配置
- Paddleの配置

Floorの配置

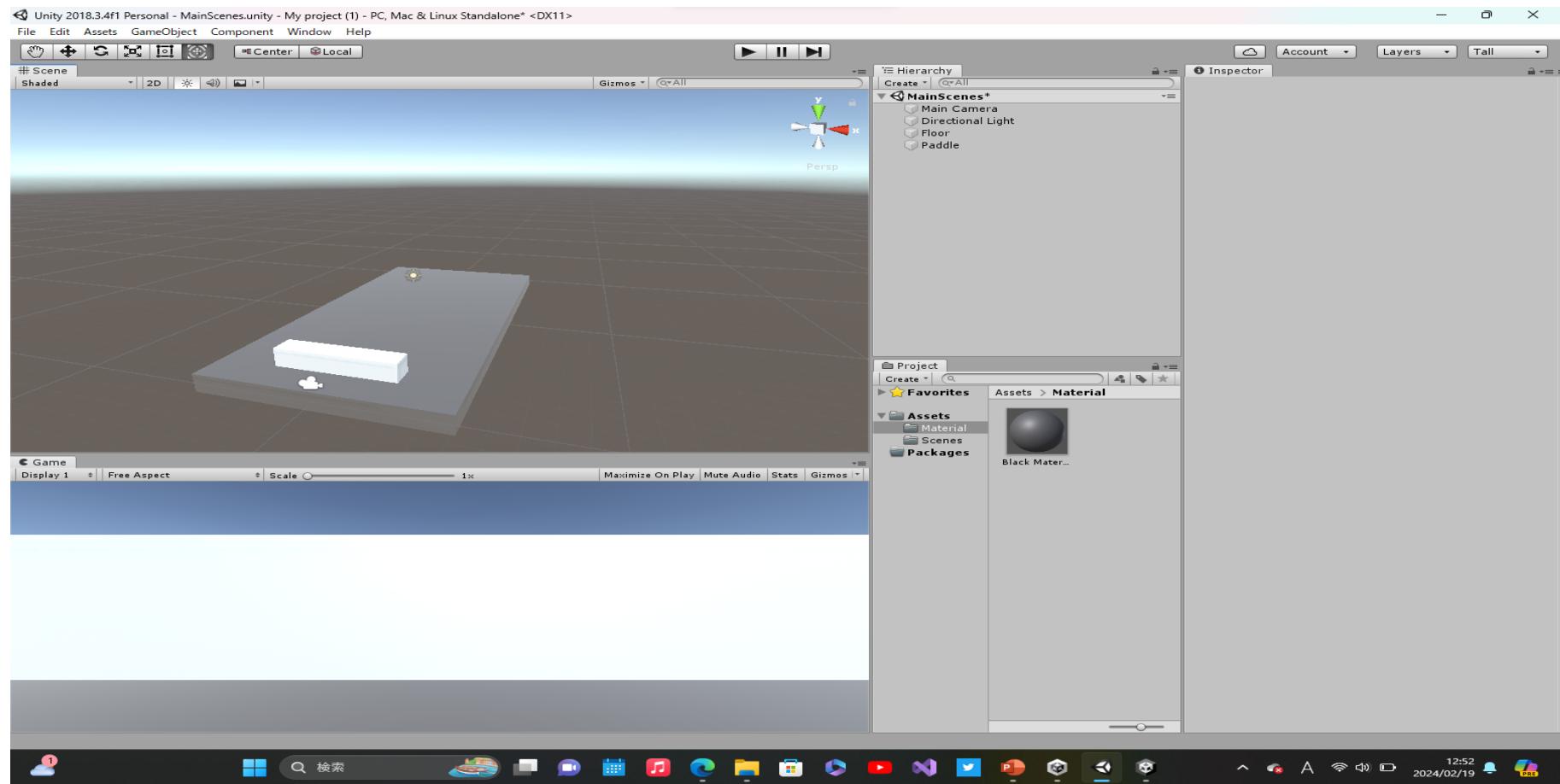
Paddleの配置



Materialを使ってみよう

- Materialの作成
- GameObjectへの適用

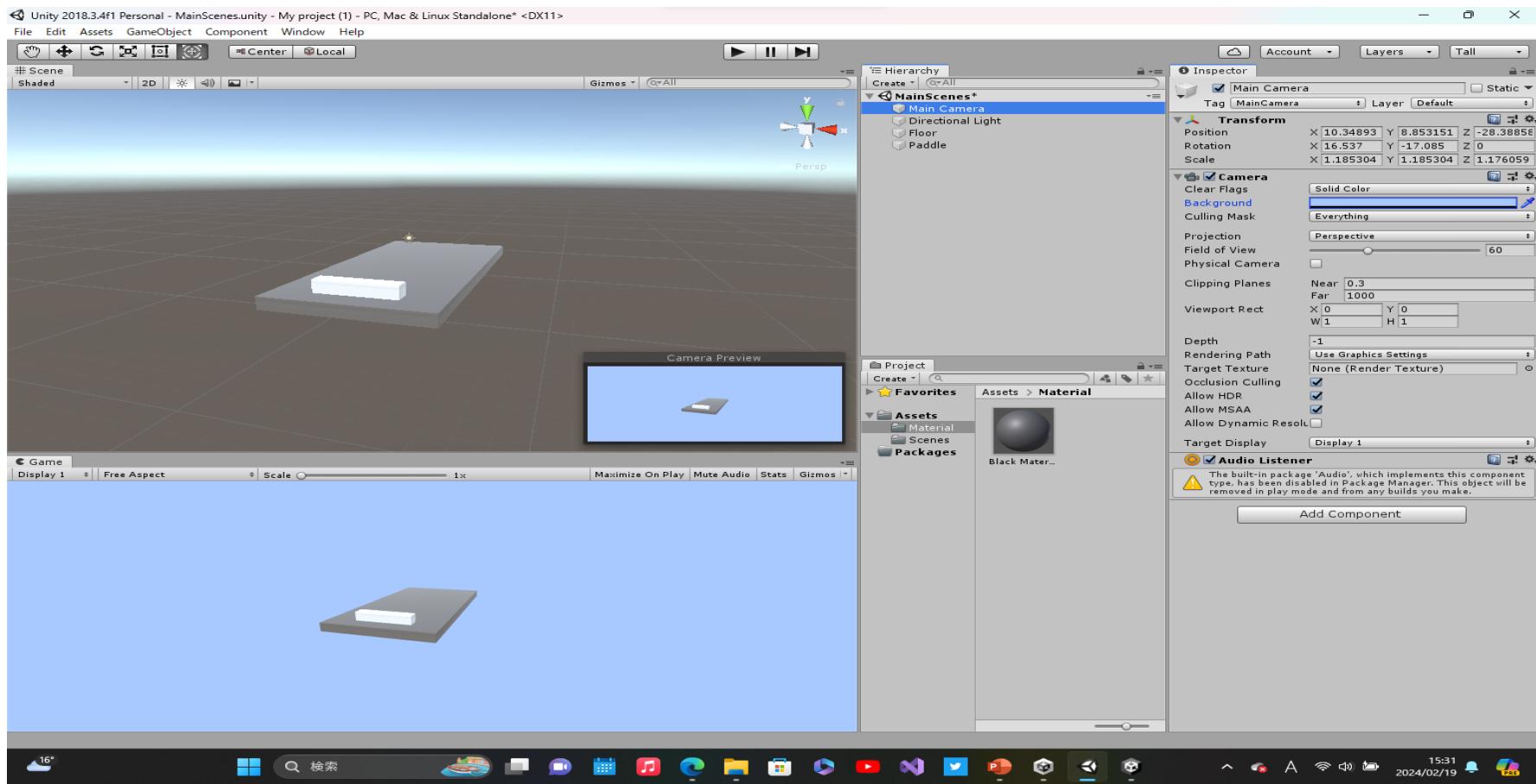
Materialを使ってみよう



カメラの位置を調整しよう

- Camera 位置の調整
- Align With View
- ゲームの背景色を変更する方法

ゲームの背景色を変更する方法



パドルにスクリプトを設定しよう

- ・スクリプトの設定
- ・Scriptsフォルダの設定
- ・スクリプトの編集

スクリプトの編集

The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes 'ファイル(F)', '編集(E)', '表示(V)', 'Git(G)', 'プロジェクト(P)', 'ビルド(B)', 'デバッグ(D)', 'テスト(S)', '分析(N)', 'ツール(T)', '拡張機能(X)', 'ウインドウ(W)', 'ヘルプ(H)', '検索', and 'My project (1)'. The toolbar contains icons for file operations like '新規', '開く', '保存', and '印刷'. The main editor window displays the code for 'PaddleScript.cs' in 'Assembly-CSharp'. The code is as follows:

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

// Start is called before the first frame update
void Start()
{
}

// Update is called once per frame
void Update()
{
}
```

Annotations in the code provide information about the Unity API usage:

- // Start is called before the first frame update
Unity メッセージ 10 個の参照
- // Update is called once per frame
Unity メッセージ 10 個の参照

The bottom status bar shows '準備完了' (Prepared), 'ソース管理に追加' (Add to Source Control), 'リポジトリの選択' (Select Repository), and the date '2024/02/19'. The taskbar at the bottom includes icons for the Start button, Search, Task View, Calendar, Music, File Explorer, YouTube, Twitter, and the Visual Studio icon.

パドルの位置を操作してみよう

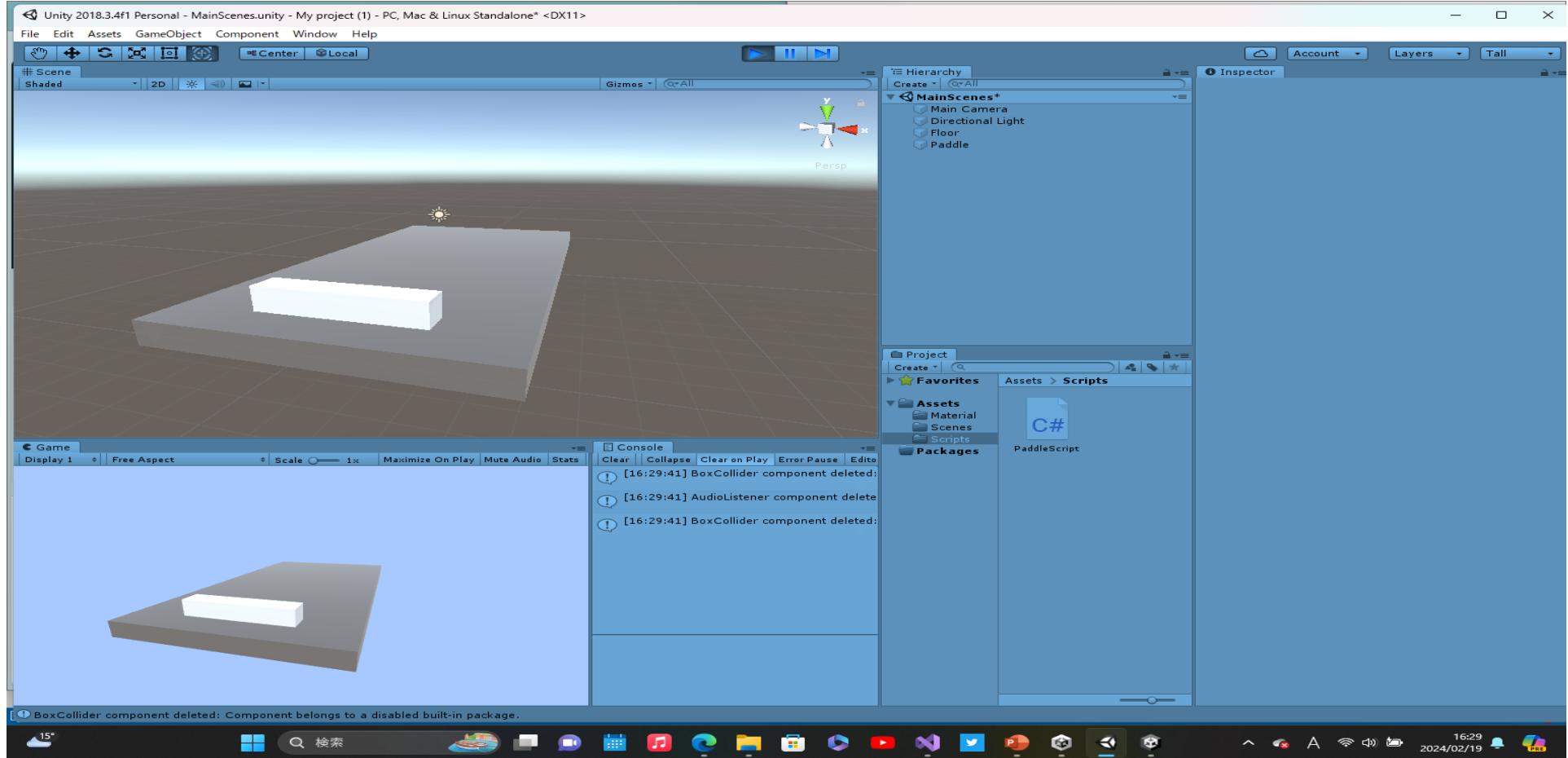
- transform.position
- new Vector3(x, y, z)
- Input.GetAxis()
- Time.deltaTime
- Debug.Log()

パドルの位置

```
void Update()
{
    transform.position += new Vector3
        (Input.GetAxis("Horizontal")
         * Time.deltaTime, 0f, 0f);
}
```

ゲームを動かしてみよう

- ・プレイモードの切り替え
- ・動作確認
- ・Playmode tintの設定



Inspectorで変数の値を操作しよう

- ・スピードの設定
- ・動作確認
- ・public変数の設定

Inspectorで変数の値を操作

```
public class PaddleScript : MonoBehaviour
{
    private float speed;

    void Start()
    {
        speed = 5f;
    }
}
```

Inspectorで変数の値を操作

```
void Update()
{
    transform.position += new Vector3
    (Input.GetAxis("Horizontal")
     * Time.deltaTime*speed, 0f, 0f);
}
```

Inspectorで変数の値を操作

```
public class PaddleScript : MonoBehaviour
{
    //private float speed;
    public float speed;
    void Start()
    {
        //speed = 5f;
    }
}
```

ボールを配置してみよう

- Ballの配置
- BallScriptの作成
- 動作確認

BallScript

```
private float speed;  
// Start is called before the first frame update  
void Start()  
{  
    speed = Random.Range(5f, 15f);  
}  
transform.position += new Vector3(0f, 0f, -1 * speed *  
Time.deltaTime);  
}
```

プレハブを使ってみよう

- Ballのプレハブ化
- BallFactoryの設定
- プレハブの設定

BallFactory

```
public class BallFactoryScript : MonoBehaviour  
{  
    public GameObject ball;
```

ボールのインスタンスを生成しよう

- Instantiate()
- 動作確認
- InvokeRepeating()

ボールのインスタンス

```
public class BallFactoryScript : MonoBehaviour  
{  
    public GameObject ball;  
    // Start is called before the first frame update  
    void Start()  
    {  
        Instantiate(ball,transform.position,transform.rotation);  
    }  
}
```

ボールのインスタンス

```
void Start()
{
    //Instantiate(ball,transform.position,transform.rotation);
    InvokeRepeating();
}
```

一定時間ごとにボールを生成しよう

- ・インスタンスの生成
- ・InvokeRepeating() の設定
- ・動作確認

InvokeRepeating() の設定

```
void SpawnBall()
{
    Instantiate(ball,new Vector3
(Random.Range(5f,5f),transform.position.y,
transform.position.z), transform.rotation);
}
```

InvokeRepeating() の設定

```
void Start()
{
    //Instantiate(ball,transform.position,transform.rotation);
    InvokeRepeating("SpawnBall",0f,1f);
}
```

パドルとの衝突判定をしてみよう

- RigidBodyの設定
- タグの設定
- OnCollisionEnter()
- 動作確認

OnCollisionEnter()

```
private void OnCollisionEnter(Collision collision)
{
    if (collision.gameObject.CompareTag("Paddle"))
    {
        Destroy(gameObject);
    }
}
```

パドルの幅が狭くなるようにしよう

- Paddle幅の変更
- 動作確認

Paddle幅の変更

```
Destroy(gameObject);

collision.gameObject.transform.localScale -= new Vector3
(Random.Range(0.2f,1.0f),0f,0f);
if (collision.gameObject.transform.localScale.x < 1.0f)
{
    collision.gameObject.transform.localScale = new Vector3(1.0f,1.0f,1.0f);
}
```

ゲームオーバー処理を作ろう

- ・ゲームオーバー処理
- ・Time.timeScale

ゲームオーバー処理

```
void Update()
{
    transform.position += new Vector3(of, of, -1 * speed * Time.deltaTime);

    if (transform.position.z < -13.of)
    {
        Time.timeScale = 0;
    }
}
```

新しくシーンを作ってみよう

- GameOverSceneの作成
- 背景色の変更
- Textの配置
- Canvasの設定

テキストを設定していこう

- Textの設定

シーンを切り替えてみよう

- Buttonの配置、設定
- Build Settings
- シーンの呼び出し
- 動作確認

ボタンが動作するように設定しよう

- ButtonScriptの設定
- シーンの呼び出し
- 動作確認

ゲームを書き出してみよう

- ・ゲームの書き出し
- ・動作確認

ありがとうございました

PaddleScript

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

public class ButtonScript : MonoBehaviour
{
    public void ReplayGame()
    {
        SceneManager.LoadScene("main scenes");
    }
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {

    }
}
```

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

public class BallFactoryScript : MonoBehaviour
{
    public GameObject ball;
    // Start is called before the first frame update
    void Start()
    {
        //Instantiate(ball, transform.position, transform.rotation);
        InvokeRepeating("SpawnBall", 0f, 1f);
    }

    void SpawnBall()
    {
        Instantiate(ball, new Vector3(Random.Range(-5f, 5f), transform.position.y, transform.position.z),
        transform.rotation);
    }
    // Update is called once per frame
    void Update()
    {
    }
}
```

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

public class Ballscript : MonoBehaviour
{
    private float speed;
    // Start is called before the first frame update
    void Start()
    {
        speed = Random.Range(5f, 15f);
    }

    // Update is called once per frame
    void Update()
    {
        transform.position += new Vector3(0f, 0f, -1 * speed * Time.deltaTime);
        if (transform.position.z < -13.0f)
        {
            //Debug.Log("Game Over");
            //Time.timeScale = 0;
            SceneManager.LoadScene("GameOverScens");
        }
    }
    private void OnCollisionEnter(Collision collision)
    {
        if (collision.gameObject.CompareTag("Paddle"))
        {
            Destroy(gameObject);
            collision.gameObject.transform.localScale -= new Vector3(Random.Range(0.2f, 1.0f), 0f, 0f);
            if (collision.gameObject.transform.localScale.x < 1.0f)
            {
                collision.gameObject.transform.localScale = new Vector3(1.0f, 1.0f, 1.0f);
            }
        }
    }
}
```

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

public class ButtonScript : MonoBehaviour
{
    public void ReplayGame()
    {
        SceneManager.LoadScene("main scenes");
    }
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {

    }
}
```