

初心者のためのUnity開発講座

第一章 Unityで覚える要素

概要

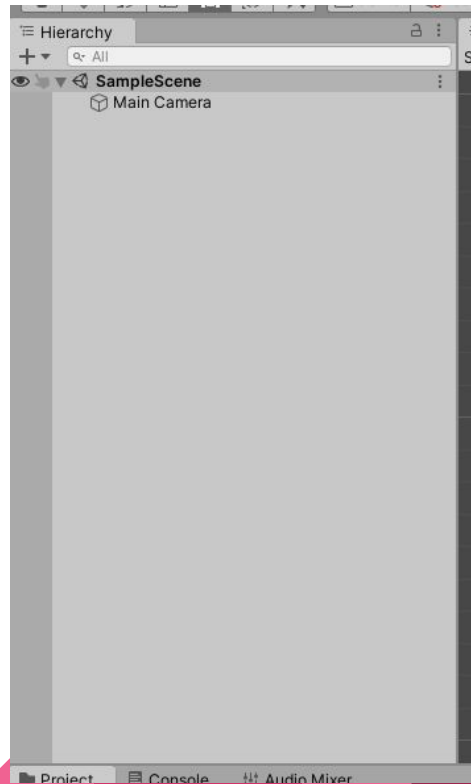
まずUnityでよく使う機能を抑える



ヒエラルキー

実際画面上にある要素を一覧で表示している機能

親子関係という要素がある



ヒエラルキー

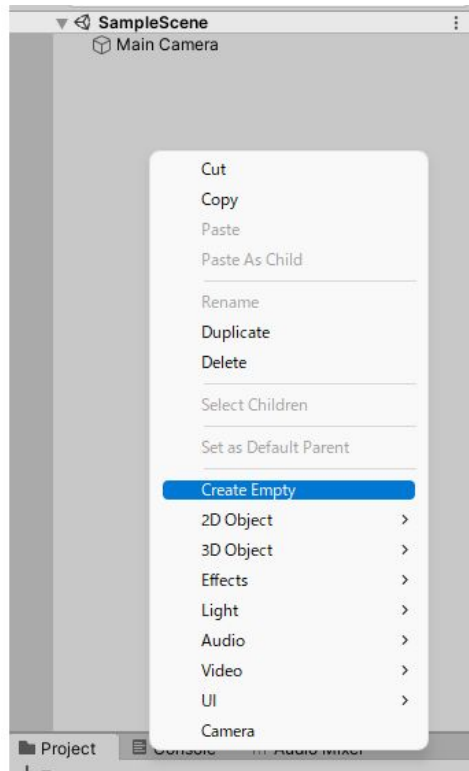
右クリックで要素を追加するメニューが表示する

空のGameObjectを追加する場合はCreate Emptyを選択する

GameObjectとは

箱だったり、テクスチャだったりの表示物を表現するもので、空の場合は何も表示されない。

座標情報やサイズ情報を持っている。



インスペクタ

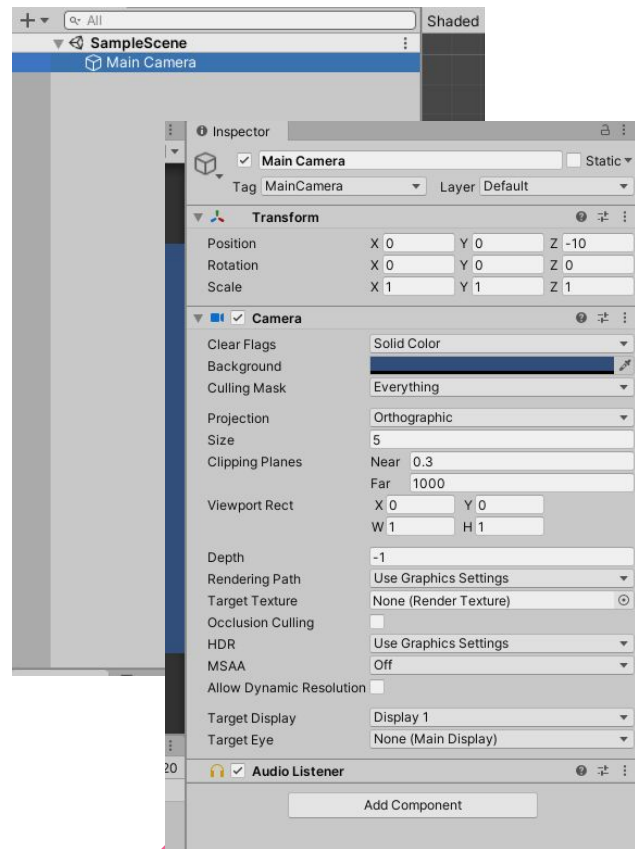
ヒエラルキーで選択したオブジェクトの情報が表示されている

Componentと呼ばれるものが羅列している。

右だと**Transform・Camera・AudioListener**が表示されている。

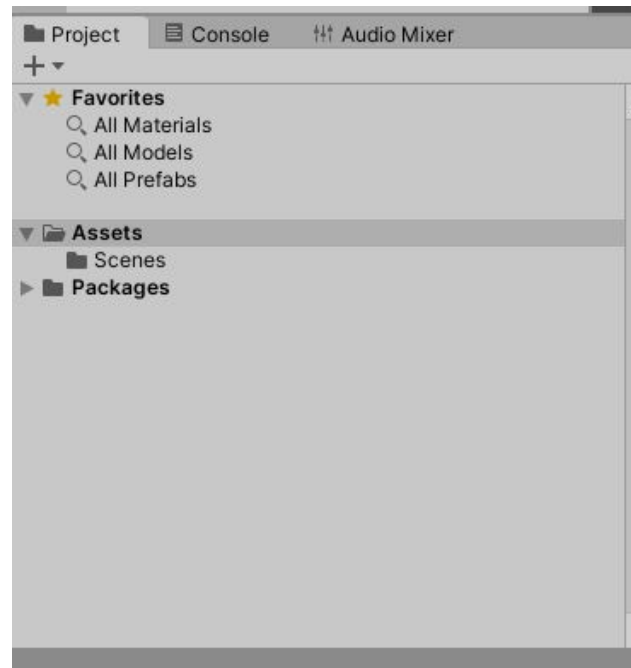
Transformとは

座標・回転・サイズの情報がある。全てのGameObjectが所持している情報。



プロジェクト

Projectは実際の素材類などがある場所で、実際のフォルダにあるコンテンツが表示している。



実際に覚えるもの

2Dゲームを作るための最低限の要素から学ぶ



Canvas

Unity上に2Dのコンテンツを表示するためのもの。

イメージ的にはキャンバスに対して描画される

文字だったり、画像だったりを表示させる

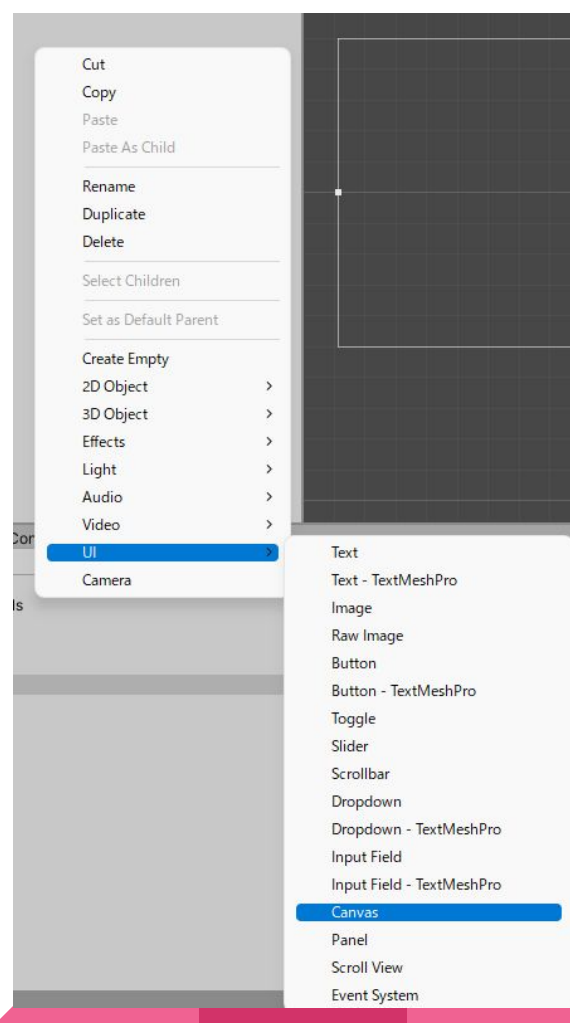


Image / RawImage

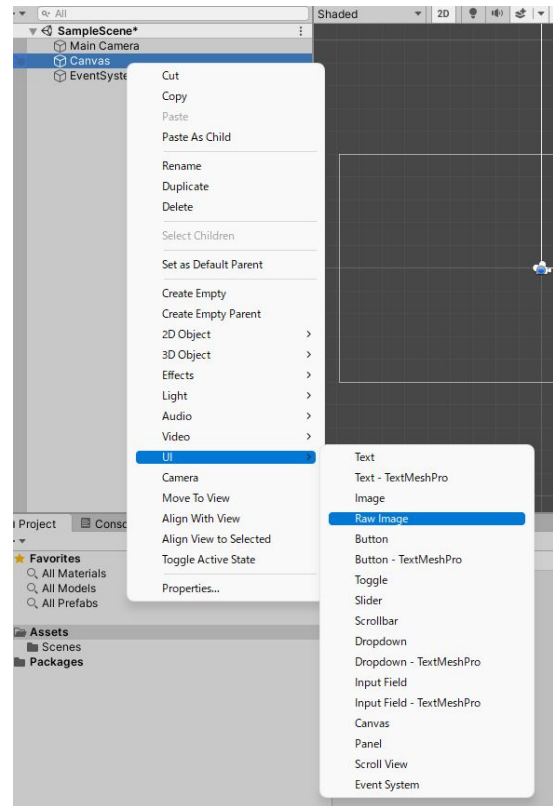
画像を表示されるもの。

Imageはsprite、RawImageはテクスチャで表示する。

Spriteとは

画像から部分的にコンテンツを表示させることができる。

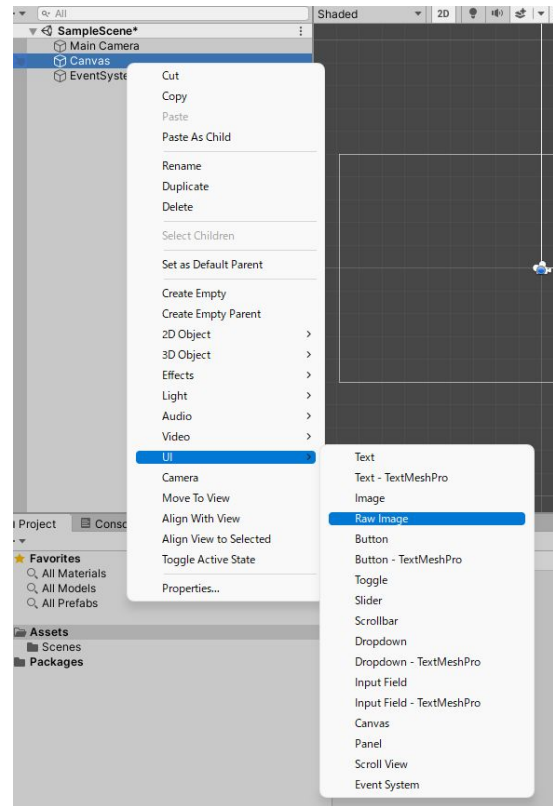
現時点では意識しなくてOK



Text

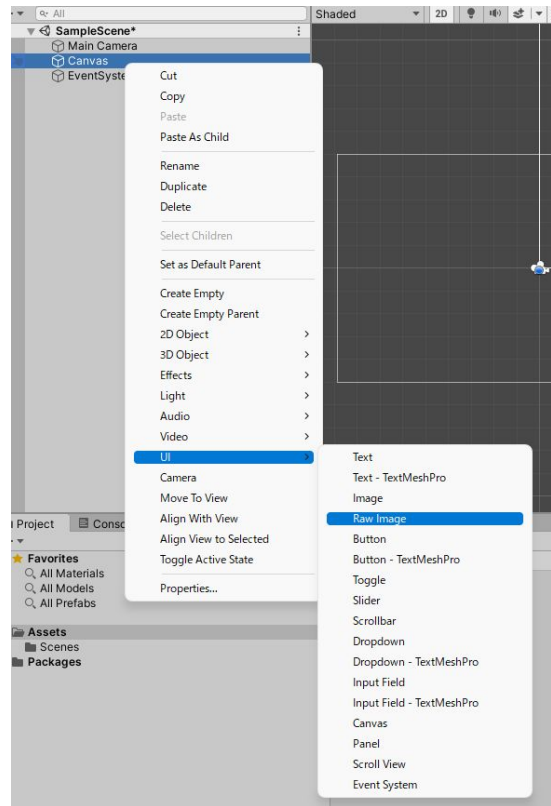
Textは文字を表示させる。

※Text - TextMeshProは一旦無視する



Button

ボタンは押したら何かしらを処理させるものを生成する。



課題

右のような画面を作ること

上級課題

ゲームを開始 と押したら

タイトル

というテキストを

ボタンを押した

というテキストに書き換える



第二章 初めてのプログラミング

概要

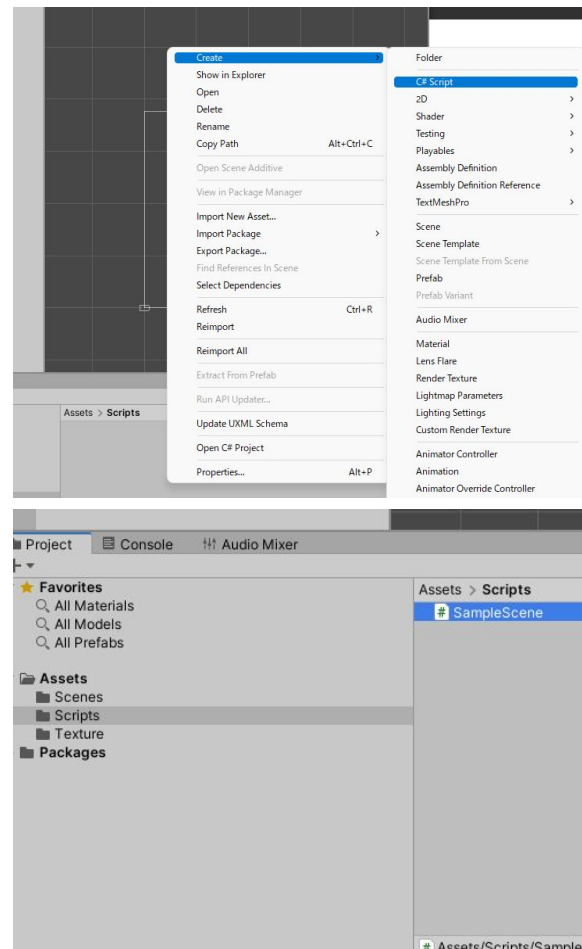
初歩的なプログラミングを学ぶ



Scriptを生成する

ProjectでCreate > C# Scriptを選んで生成する。

今回は「SampleScene.cs」と命名して生成する。



IDE (Visual Studio等)でScriptを開く

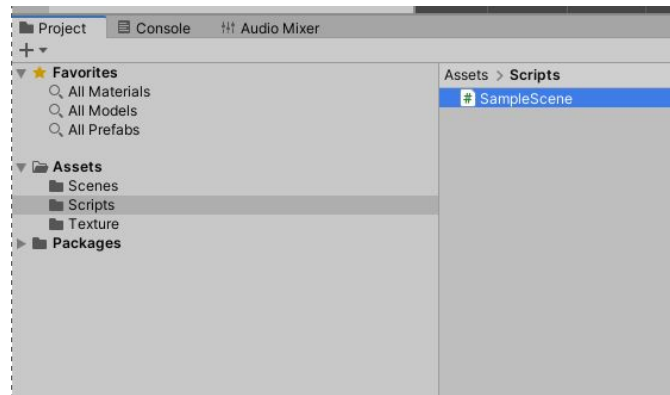
SampleSceneをダブルクリックして開くことができる

開かない場合は

Preference > External Tools で設定すること

※ソフトウェアインストールも必要

Rider, Visual Studio Codeなど
いろいろツールは存在する



今回したいこと

ボタンを押したら

タイトルの文字が別の何かに変わるように
する



実装

生成した直後はこんな感じ

ボタンに処理登録が必要なので登録の
処理を記述する

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

// No asset usages
public class SampleScene : MonoBehaviour
{
    // Start is called before the first frame update
    // Event function
    void Start()
    {
    }

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

実装

変数の設定をclassの中にします。

classの中に変数を書く(メンバ変数という)

変数の型はButtonになります。

Buttonのクラスを使う場合、

using UnityEngine.UI;

と記述が必要。

これはUIのシステムを使います、という宣言を意味します

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

// No asset usages
public class SampleScene : MonoBehaviour
{
    public Button startButton; // Unchanged

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

    }

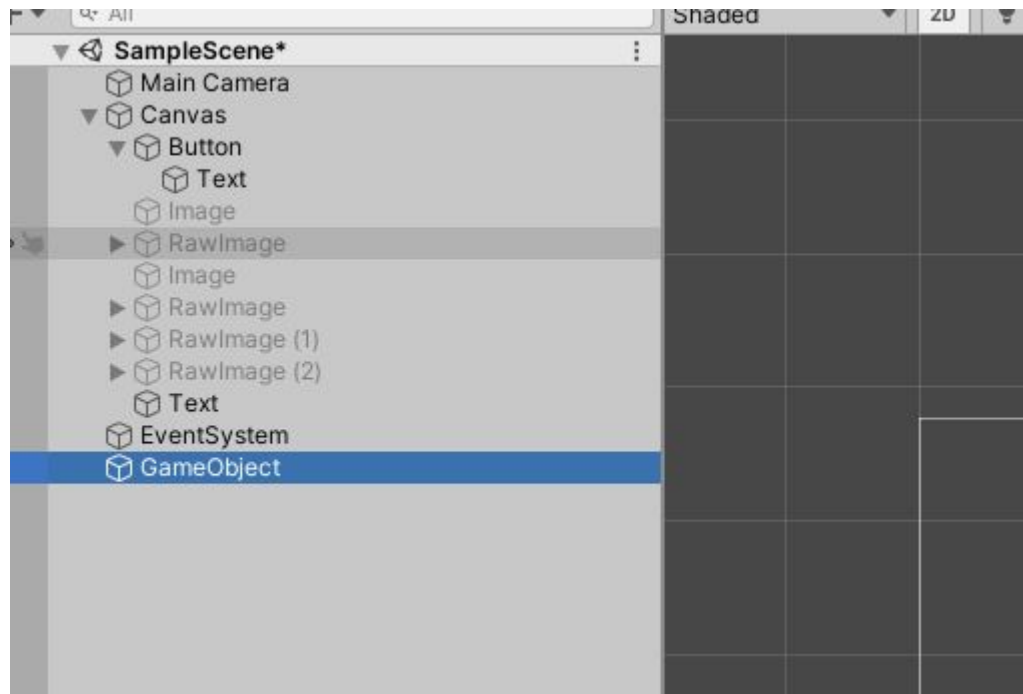
    // Update is called once per frame
    // Event function
    void Update()
    {

    }
}
```

実装

Unityに戻って、

GameObjectをCreateします。

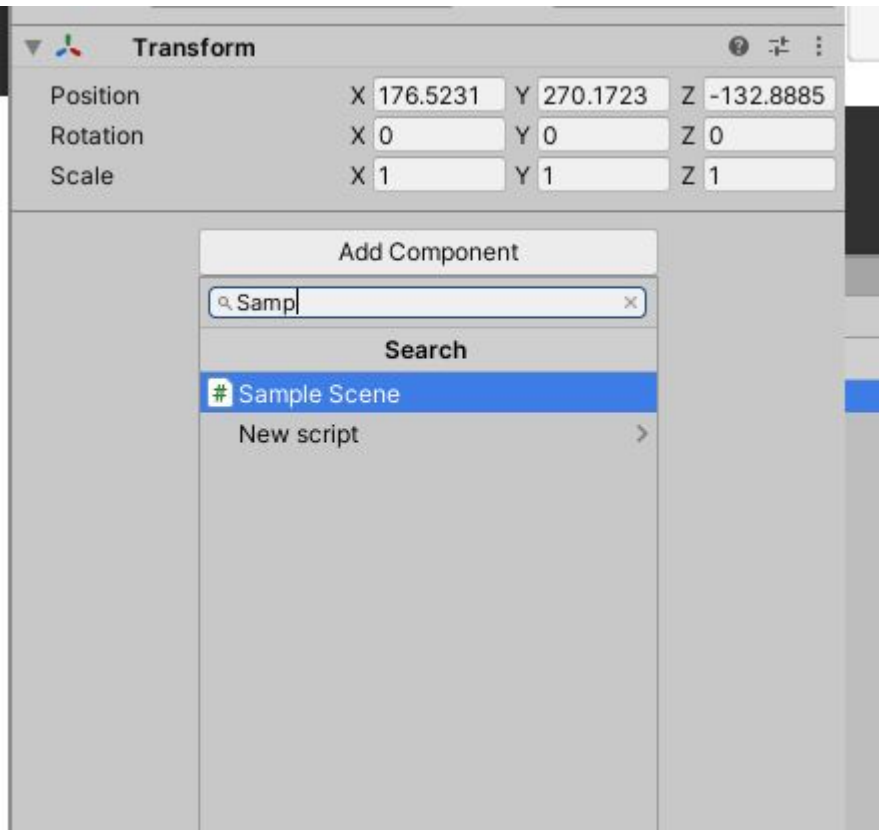


実装

Inspectorに移動し、

AddComponentを使って

SampleSceneをアタッチします。

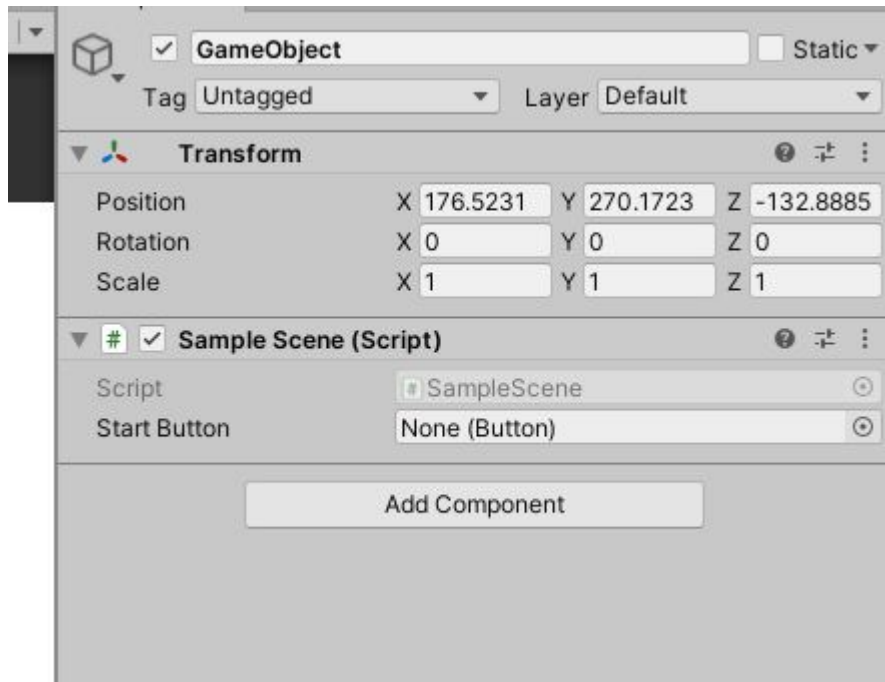


実装

するとSampleSceneが表示されます。

さっきソースコードに記述したButtonが表示されています。

この要領でタイトル文字も記述します。



実装

右記のように**titleText**を追記する。

このようにするとInspectorに

TitleText

が追加されています。

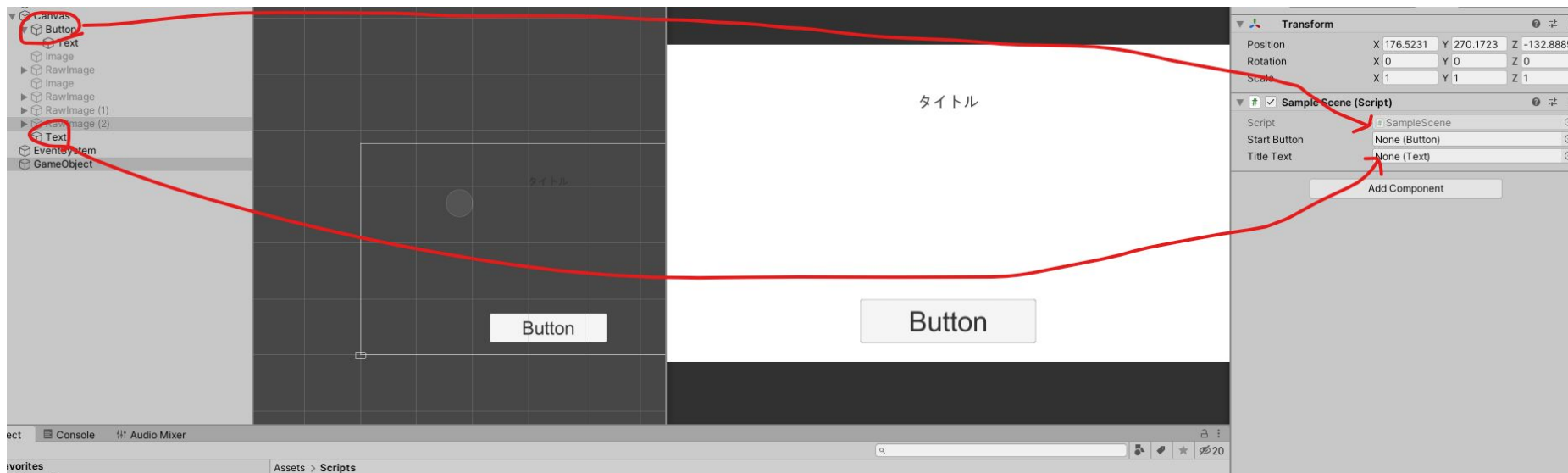
```
No asset usages
public class SampleScene : MonoBehaviour
{
    public Button startButton;  ⚙️ Unchanged
    public Text titleText;  ⚙️ Unchanged

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

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


実装

ヒエラルキーのオブジェクトからドラッグアンドドロップでStartButtonにボタンを、TitleTextにテキストを設定します。



実装

右記のようにテキストを書き換えて
実行すると、以下のようにタイトルが
変わります。

今回の目的はボタンを押して処理を変える
なのでボタンに処理を次に追加します。

```
No asset usages
public class SampleScene : MonoBehaviour
{
    public Button startButton;
    public Text titleText;

    // Start is called before the first frame update
    void Start()
    {
        titleText.text = "竜がガバ";
    }

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

竜がガバ

Button

実装

右記のようにボタン処理を追加します。

中に書いている処理は**無名関数**という。

詳細は今回省くが、ボタンで押されたときに実行した処理を登録するソースコードです。

```
1 asset usage
public class SampleScene : MonoBehaviour
{
    public Button startButton; 1 Button (MonoBehaviour)
    public Text titleText; 1 Text (MonoBehaviour)

    // Start is called before the first frame update
    1 Event function
    void Start()
    {
        titleText.text = "竜がガバ";

        startButton.onClick.AddListener(() =>
        {
            titleText.text = "竜がガバガバ";
        });

        // startButton.onClick.AddListener(() =>
        // {
        //     titleText.text = "竜がガバガバガバ";
        // });
        // 上記の処理は無名関数という。
        // 処理の働きとしては
        //
        // startButton.onClick.AddListener(OnStart);
        // void OnStart()
        // {
        //     titleText.text = "竜がガバガバガバ";
        // }
        // 同じ処理
    }

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

実装

実装するとこのようになる。

上級課題

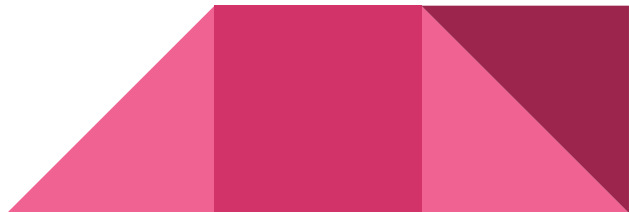
背景にキャラクターイラストを乗せる。

※いらすとやや自分の絵を使う

初期カラーは黒で、ボタンを押したら
白色にし、絵そのものの見た目が表示
されるようにすること



竜がガバガバ



第三章 イメージ

今回は背景にキャラクターを追加します

RawImageを使ってキャラクターを表示します

RawImageとはUGUIの機能で、UIから生成できます。



ImageとRawImageの違い

ImageとRawImageの違い

Imageとは

Spriteで表現される機能で、Spriteは画像から生成した情報になる

RawImageとは

画像そのものを表示する



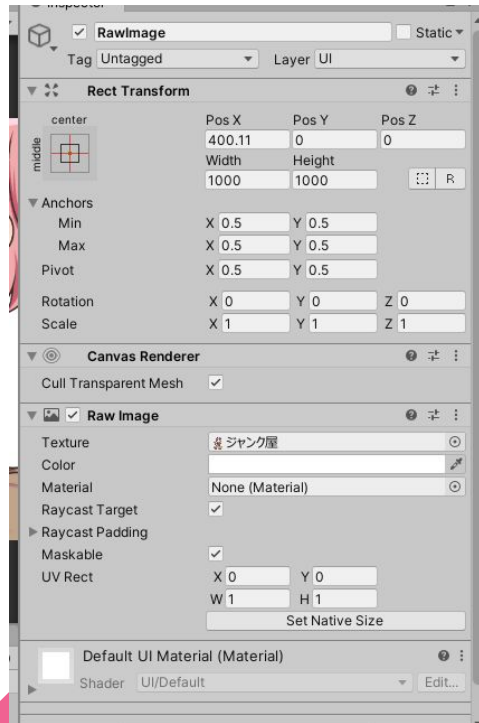
RawImage

Width・Heightでサイズ表現がされる。

Set Native Size を押すと画像サイズに合う。

Rawcast TargetはButtonなどで処理するようにするための設定になります

Colorは画像に対して**加算**で値に情報を与えます



Image

ImageはSpriteという情報で表現されているため、
RawImageよりも機能が多いです



Image

ImageはSpriteという情報で表現されているため、
RawImageよりも機能が多いです

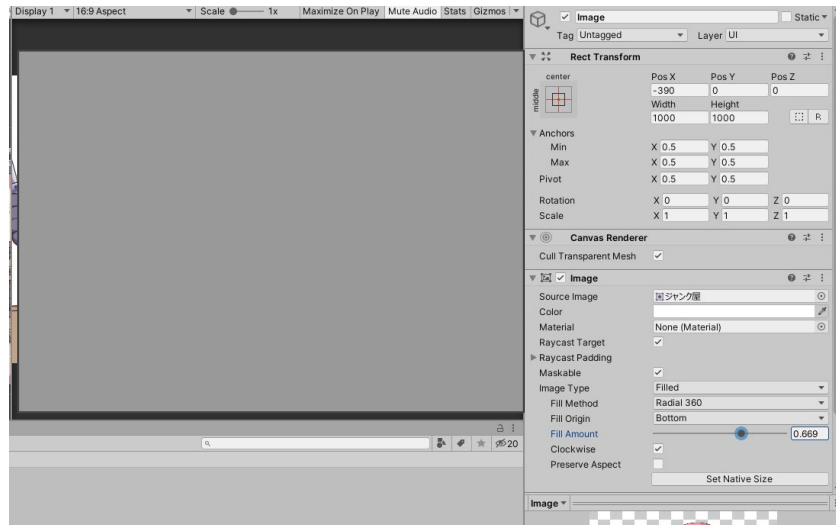


Imageの機能

例えば「ImageType」をFilledにします

その後右記のように設定すると画像中心に
画像を非表示にできます

Spriteはこういった表現ができます



課題

タイトルの背景にImageを使ってタイル表示をする
ボタンを押して画像を白色から黒色へとしましょう

