【目標】タイトル、ゲームオーバー、リザルトのシーン切り替え。

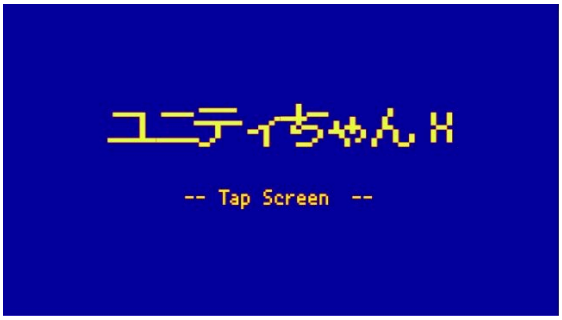
タイトル画面からステージ紹介画面、ステージ画面、ゲームオーバー/次ステージへのシーン切り替えの実装を行います。

ここでの重要キーワード： 、

1. 画面作成 1 タイトル画面

現在、ゲーム本体になるシーンは現在作成しているため、まずはタイトルを作成することを考えます。
メニューバーから File > New Scene と選択し、生成されたシーンを ”Title” という名前で保存しておきます。

最初に適当なタイトルを uGUI を使って作成します。



シーン紹介用の “Stage1” シーンへ遷移するためのスクリプト “CallSatge1Script” を作成し、シーンが読み込まれた際、ステージ上に配置されている適当なゲームオブジェクトに、先ほど作成した”CallStage1Script”スクリプトを適用しておきます。

CallStage1Script.cs

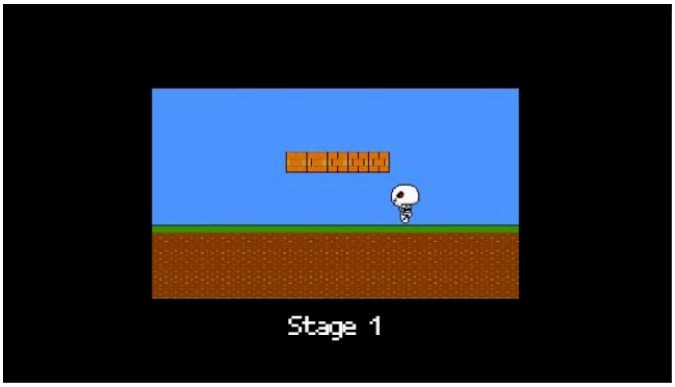
```
1   using UnityEngine;
2   using UnityEngine.SceneManagement;
3   using System.Collections;
4
5   public class CallStage1Script : MonoBehaviour {
6
7       // Use this for initialization
8       void Start () {
9
10      }
11
12      void Update()
13      {
14          if (Input.GetMouseButtonDown(0))
15          {
16              SceneManager.LoadScene("Stage1");
17          }
18      }
19  }
```

2. Stage1 作成

続いて Stage1 画面を作成します。いきなりゲーム画面に遷移するのではなく、ゲームの紹介画面を表示して、一定時間経過するとゲーム画面に切り替えるようにしてみましょう。

“Stage1”という名前の新規シーンを Title と同じように作成保存後、そのシーンに uGUI を使って適当にゲーム紹介画面を作成します。

作成例



画面キャプチャし、ゲームの雰囲気が分かるようにしてあります。

作成後、2 秒経過で”Main”シーンへ遷移するよう “CallMainScript” を作成し、適当なゲームオブジェクトに適用しておきます。時間経過の計測にはいくつか方法がありますが。この場合大きく次の 2 つの方法があります。

〇Time.deltaTime を利用する。

〇コルーチン＋WaitForSeconds()を利用する。

Time.deltaTime を利用するスクリプトを、下記のテンプレートを元に考えて下さい。Update ごとに Time.deltaTime を加算し、比較すれば大丈夫です。可能であればコルーチンの利用もチャレンジして下さい。

CallMainScript.cs テンプレート

```
1   using UnityEngine;
2   using UnityEngine.SceneManagement;
3   using System.Collections;
4
5   public class CallMainScript : MonoBehaviour {
6       // 管理用変数の定義
7
8       // Use this for initialization
9       void Start () {
10
11      }
12
13      void Update()
14      {
15          // Time.deltaTime の加算
16          // Time.deltaTime 加算後、時間比較 ＞ 時間経過後シーンの切り替え
17
18      }
19  }
```

3. ゲームオーバー機能の実装：ライフ “0”

続いてゲームオーバー処理を実装します。ゲームオーバーになる条件は

〇ライフが ”0” になる。

〇穴に落ちる。

の 2 つになります。

ゲームオーバーのメッセージ処理のために、”Main” シーンの画面中央に ”GAME OVER” の文字を配置し、インスペクター上で非表示にしておきます。

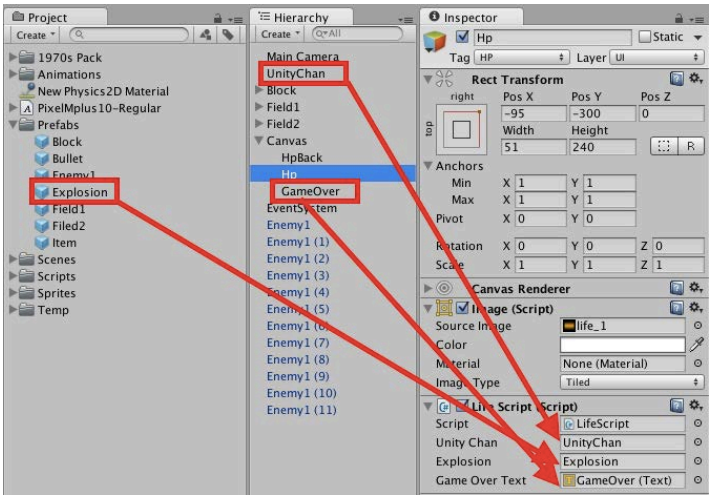
次に、Unity ちゃんのライフが ”0” になったら、先ほどの ”GAME OVER” を画面上に表示するよう、Life スクリプトに修正を加えます。

Life.cs

```
1   using UnityEngine;
2   using UnityEngine.SceneManagement;
3   using System.Collections;
4
5   using UnityEngine.UI;
6
7   public class Life : MonoBehaviour {
8       RectTransform rt;
9
10      public  GameObject  unityChan;    //ユニティちゃん
11      public  GameObject  explosion;    //爆発アニメーション
12      public  UnityEngine.UI.Text    gameOverText;    //ゲームオーバーの文字
13      private bool        gameOver = false; //ゲームオーバー判定
14
15      void Start ()
16      {
17          rt = GetComponent<RectTransform>();
18      }
19
20      void Update ()
21      {
22          //ライフが 0 以下になった時、
23          if (rt.sizeDelta.y <= 0) {
24              //ゲームオーバー判定が false なら爆発アニメーションを生成
25              //GameOver メソッドで true になるので、1 回のみ実行
26              if (gameOver == false) {
27                  Instantiate (explosion,
28                      unityChan.transform.position + new Vector3 (0, 1, 0),
29                      unityChan.transform.rotation);
30              }
31              //ゲームオーバー判定を true にし、ユニティちゃんを消去
32              GameOver ();
33          }
34          //ゲームオーバー判定が true の時、
35          if (gameOver) {
```

```
36                      //ゲームオーバーの文字を表示
37                      gameOverText.enabled = true;
38                      //画面をクリックすると
39                      if (Input.GetMouseButtonDown (0)) {
40                          //タイトルへ戻る
41                          SceneManager.LoadScene("Title");
42                      }
43                  }
44              }
45
46              public void LifeDown (int ap)
47              {
48                  rt.sizeDelta -= new Vector2 (0,ap);
49              }
50
51              public void LifeUp (int hp)
52              {
53                  rt.sizeDelta += new Vector2 (0,hp);
54
55                  if (rt.sizeDelta.y > 240f) {
56                      rt.sizeDelta = new Vector2 (51f, 240f);
57                  }
58              }
59
60              public void GameOver ()
61              {
62                  gameOver = true;
63                  Destroy(unityChan);
64              }
65          }
```

public 変数である unityChan と Explosion、gameOverText にオブジェクトを設定します。



4. ゲームオーバー機能の実装：穴に落ちる。

続いて穴に落ちたときの処理を実装します。条件として

”穴に落ちる” ＝ “プレイヤーの位置が規定の位置よりも下になる”

ことで、判断できます。そこで、メインカメラの位置よりも “8” 低くなった時点で、ゲームオーバーとします。Player スクリプトで、HP オブジェクトを取得し、Update にて現在のカメラの高さと(y)を比較し、HP オブジェクトに設定されている Life スクリプトの GameOver()メソッドを呼び出せばよい。

実際に実装してみよう。