

Tutorial : Membuat Permainan Flappy Bird menggunakan Unity3D

“Tutorial ini mengajarkan pembuatan permainan Flappy Bird menggunakan Unity3D namun Unity dapat digunakan untuk membuat permainan 2D”

Langkah 1 : Install Unity3D

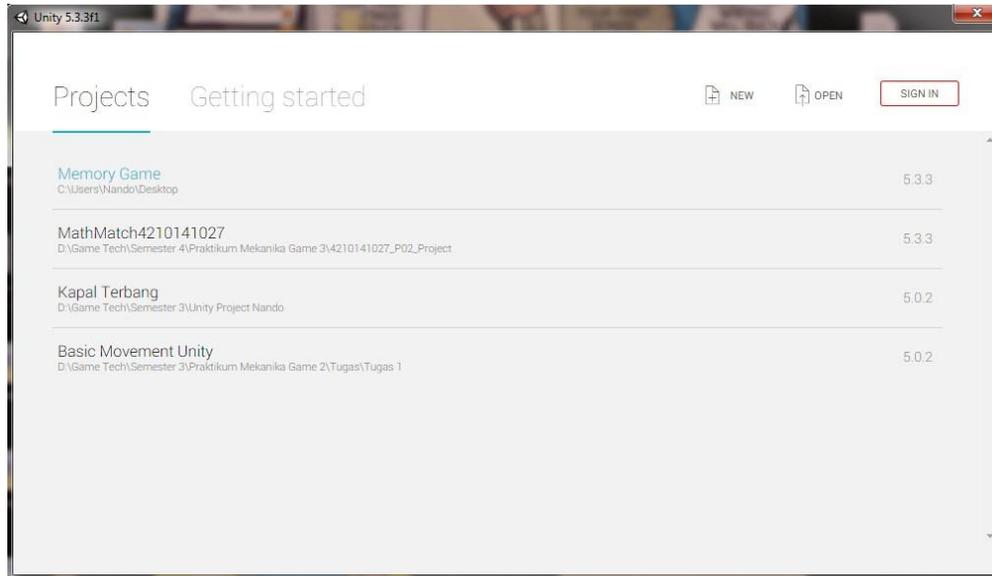
“Kalau sudah diinstall langsung ke langkah 2 saja”



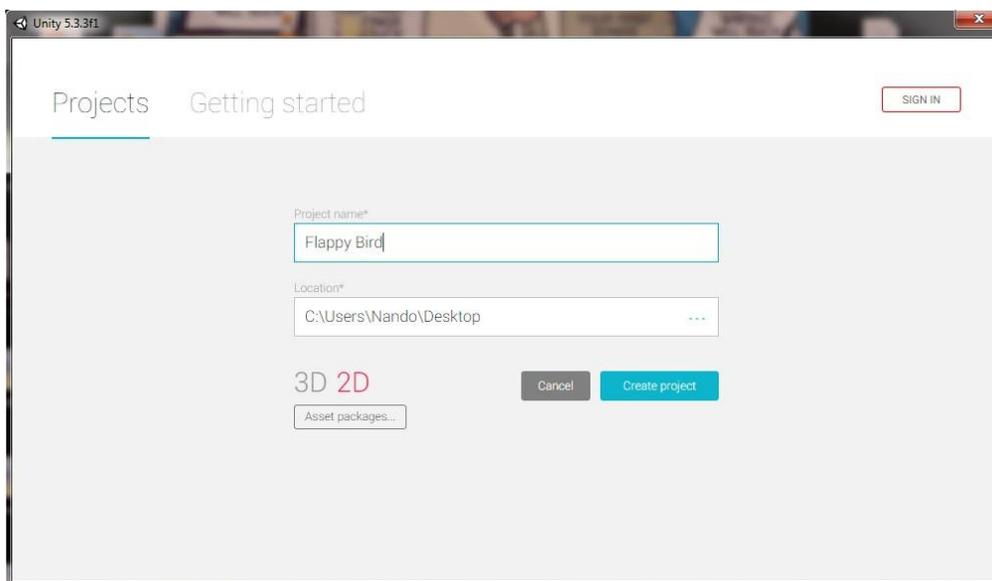
Unity3D adalah sebuah game engine free. Meskipun terdapat tulisan “3D” namun versi – versi terbaru dari Unity mampu membuat permainan 2D. Ketika memuat permainan menggunakan Unity, kita dapat mendeploynya dalam berbagai platform seperti Android, iOS, Windows Phone, dll.

Langkah 2 : Membuat Project Baru

Ini adalah tampilan pertama yang akan kalian temui saat membuka Unity :



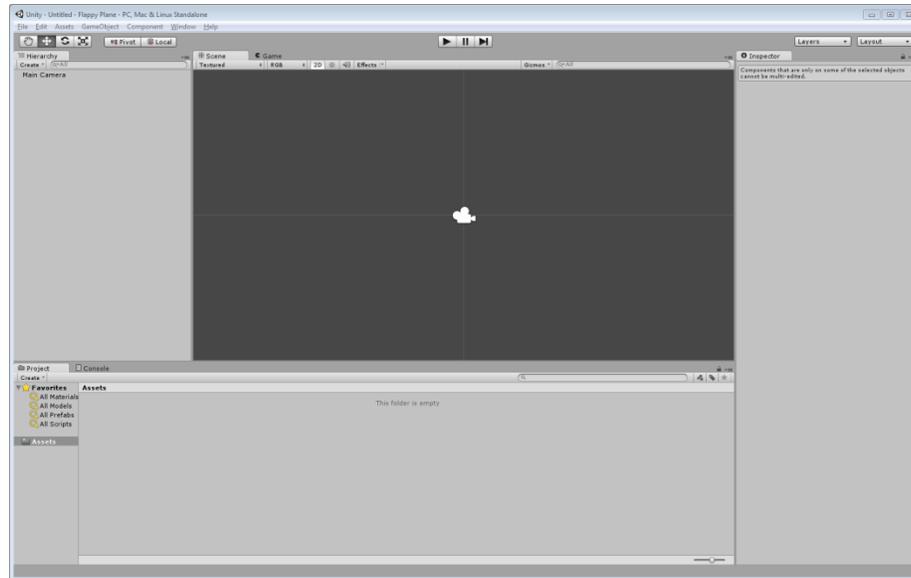
Kemudian pilihlah “+New” yang berada pada kanan atas tepat disebelah tombol “+Open”. Lalu akan terdapat tampilan seperti dibawah ini :



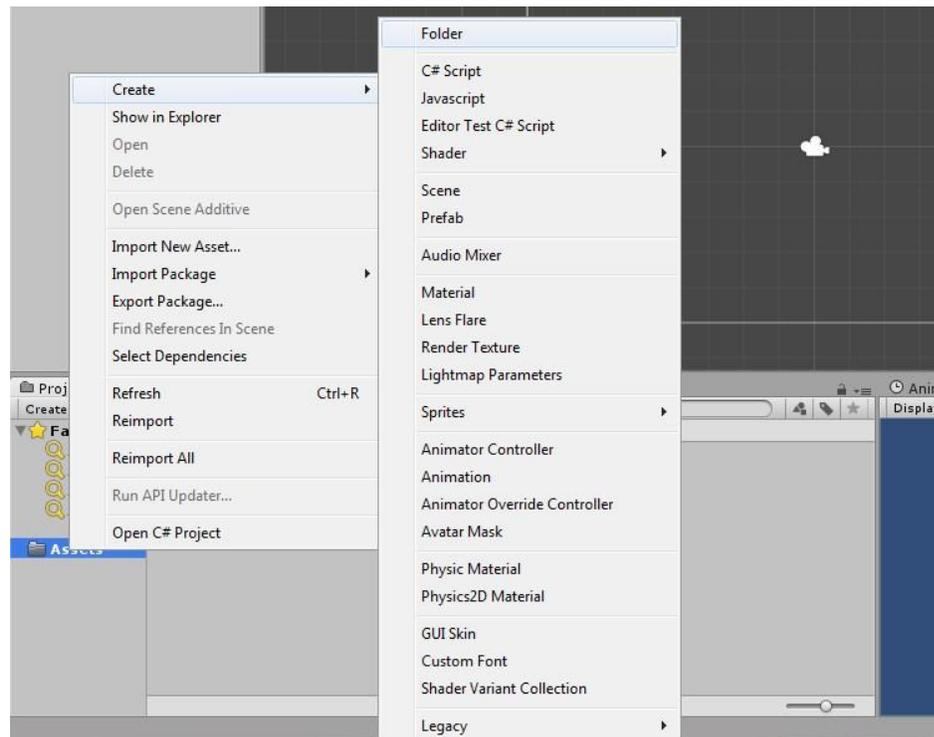
Berilah Project Name sesuai dengan yang diinginkan namun Project Name yang disarankan adalah “Flappy Bird”/ “Flappy Plane”. Kemudian pilihlah “2D” pada kiri bawah dan klik “Create Project”.

Langkah 3 : Mengorganisir Project

Setelah membuat project baru kalian akan menemui tampilan seperti ini :



Kemudian fokuslah pada “Project Window” yang tepat berada dipojok kiri bawah dan buatlah 4 folder bernama “Prefab”, “Scenes”, “Scripts”, “Sprites”. Untuk membuat 4 folder tersebut dapat dilakukan dengan cara Klik kanan pada “Assets”> Create > Folder.



Fungsi dari keempat folder tersebut dan kegunaanya adalah :

Prefab (Menyimpan Prefab)

Dalam Unity, Prefab adalah sebuah object yang dapat digunakan berulang – ulang atau di generate berulang – ulang seperti peluru, lawan, atau tembok. Namun kita tidak perlu menembahkannya pada “Hierarchy” Window.

Scenes (Menyimpan Scenes)

Dalam Unity, Scenes adalah sebuah tempat kita membuat sebuah level dalam sebuah permainan atau main menu. Jadi setiap scenes memiliki isi yang berbeda – beda entah itu level atau main menu atau informasi dari permainan tersebut.

Scripts (Menyimpan Scripts)

Dalam Unity, Scripts adalah barisan code - code yang menggerakkan keseluruhan permainan.

Sprites (Menyimpan Sprite)

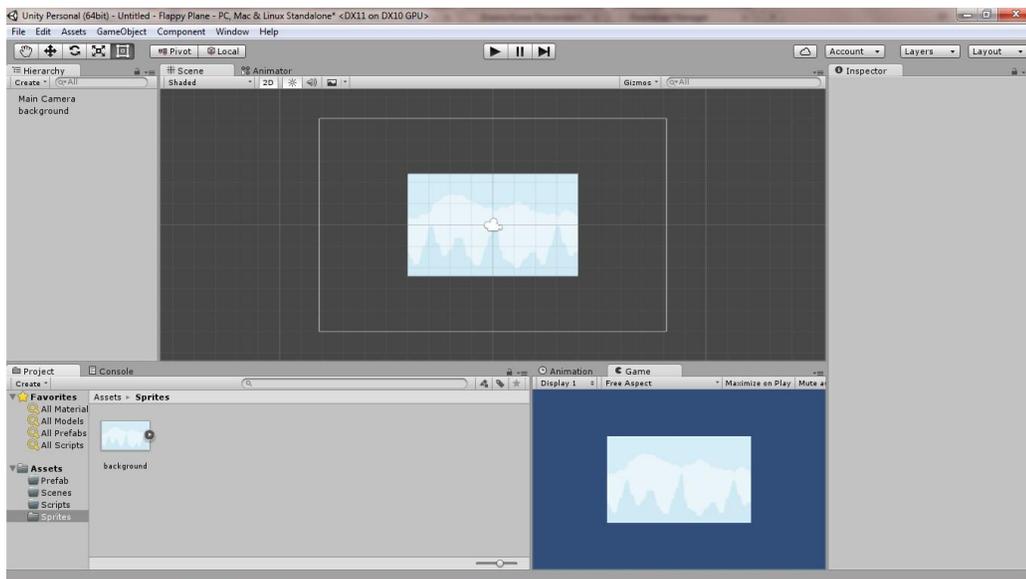
Dalam Unity, Sprite adalah gambar atau semua object yang memperindah permainan tersebut.

Langkah 4 : Set Up Background

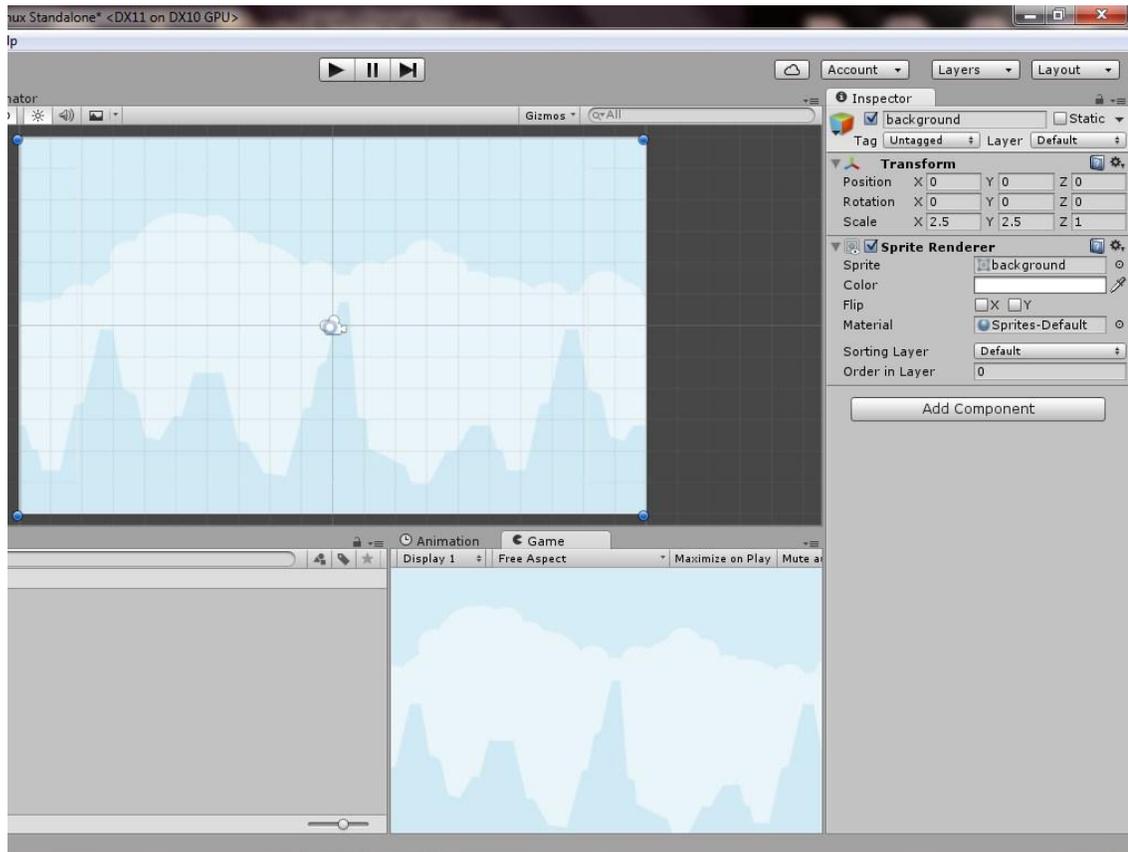
Masukkan gambar background dibawah ini kedalam folder “Sprites”/
Gambar dapat kita temukan pada folder “Flappy Plane Bundle”



Kemudian drag gambar tersebut kedalam “Scene” window dan kalian akan menemui seperti pada gambar dibawah.



Lalu aturlah scalenya menjadi X : 2,5 dan Y : 2,5. Scale tersebut dapat ditemui pada “Inspector” Window seperti gambar dibawah ini :

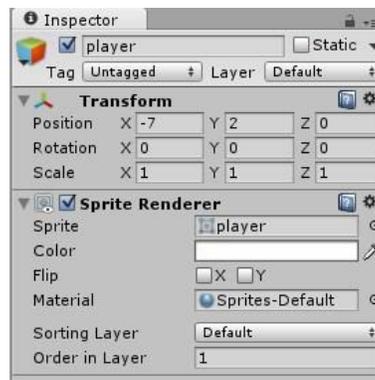


Langkah 5 : Menambahkan Player

Masukkan gambar player dibawah ini kedalam folder “Sprites”/ Gambar dapat kita temukan pada folder “Flappy Plane Bundle”. Kemudian drag and drop pada “Scene” window.



Kemudian ubahlah nilai dari “Order in Layer” pada “Inspector” window menjadi 1. Hal ini untuk memastikan bahwa player selalu berada dipaling atas.



Pada Inspector juga tambahkan “Rigidbody 2D” ini dapat ditemukan dengan cara klik Add Component > Ketikkan “Rigidbody2D” > Klik “Rigidbody2D”.

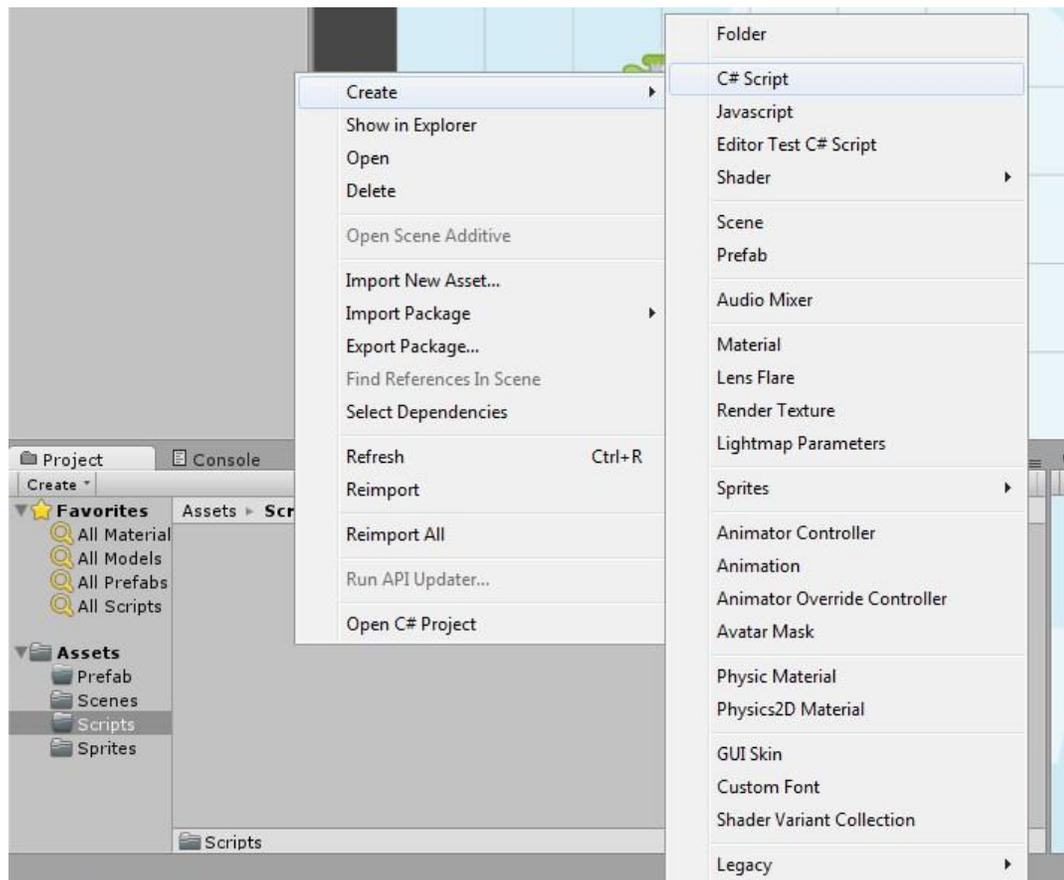


Setelah menambahkan “Rigidbody2D” maka ketika kita memplay Unity kita maka player kita akan langsung jatuh karena efek gravitasi yang diberikan oleh “Rigidbody2D”

Langkah 6 : Controlling Player



Buatlah sebuah script dengan nama “Player” pada folder “Scripts”. Untuk membuat script dapat dilakukan dengan mengklik folder “Scripts” > Klik kanan > Create > C# Script > Beri nama “Player”.



Kemudian double – klik script tersebut maka kita akan terhubung dengan MonoDevelop dan disinilah kita akan mulai memberikan coding pada permainan kita. Kemudian tambahkan code seperti dibawah ini yang berfungsi agar saat kita menekan tombol spasi player dapat naik kembali.

```

1 using UnityEngine;
2 using System.Collections;
3
4 public class Player : MonoBehaviour {
5
6     public Vector2 jumpForce = new Vector2 (0, 300);
7     void Update () {
8         if (Input.GetKeyDown("space")) {
9             GetComponent<Rigidbody2D>().velocity = Vector2.zero;
10            GetComponent<Rigidbody2D>().AddForce (jumpForce);
11        }
12    }
13 }
14

```

Pada code diatas terdapat 2 bagian utama yaitu “jumpForce” dan fungsi/method “Update”

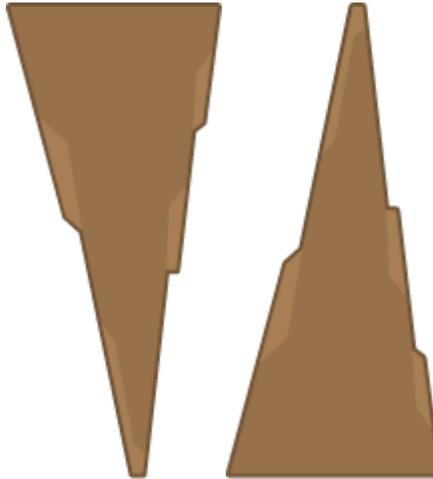
jumpForce : Mengatur tinggi dan jarak loncat kita melalui x dan y karena jumpForce di set public maka dapat diganti pada “Inspector” window.

Update : Mengatur tiap frame pada permainan tersebut dan didalam update tersebut kita telah mengatur apabila kita menekan space maka sebuah force ditambahkan pada rigidbody.

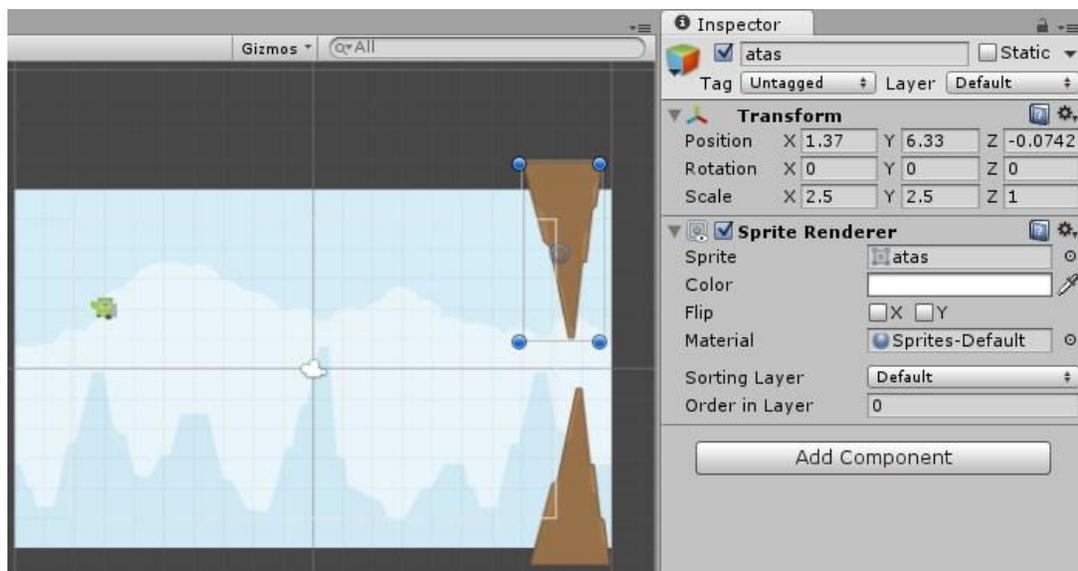
Untuk menambahkan script tersebut kita bisa melakukan dengan drag and drop pada “Inspector” window player atau melalui Add Component.

Langkah 7 : Membuat Obstacle

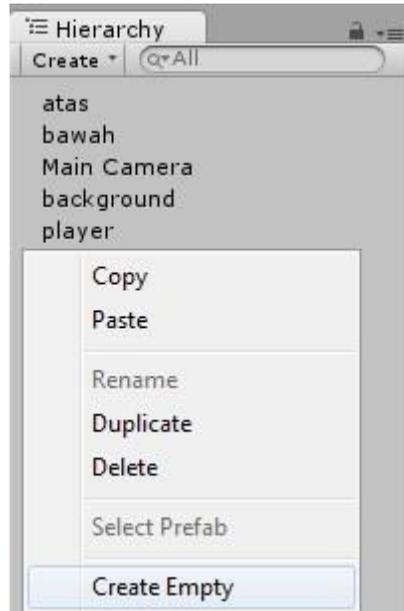
Masukkan gambar batu atas dan bawah dibawah ini kedalam folder “Sprites”/ Gambar dapat kita temukan pada folder “Flappy Plane Bundle”. Kemudian drag and drop pada “Scene” window.



Setelah dimasukkan kedalam “Scene” window kemudian aturlah scale dari kedua batu tersebut menjadi X : 2.5 dan Y : 2.5. Lalu posisikan batu – batu tersebut saling berhadapan atas bawah dan berikan sedikit jarak agar player mampu melwatinya.. Kemudian ubahlah nilai dari “Order in Layer” pada “Inspector” window menjadi 1. Hal ini untuk memastikan bahwa obstacle selalu berada dipaling atas.

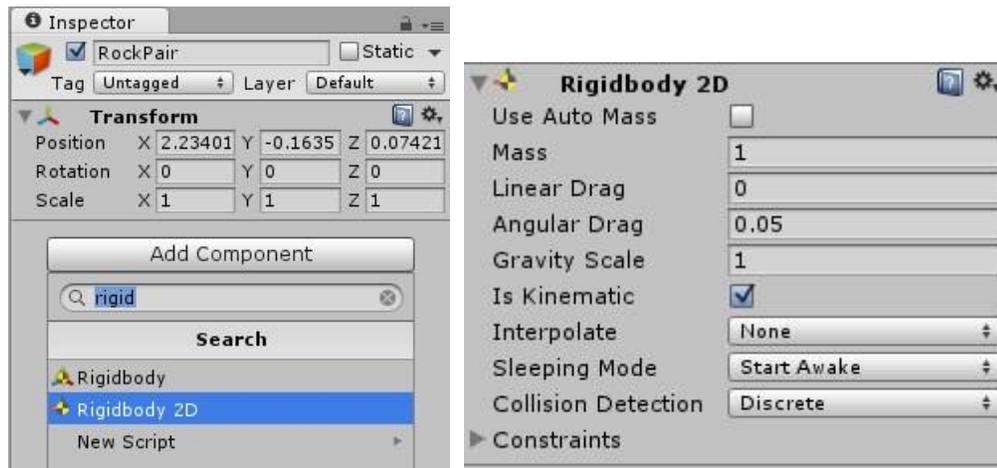


Kemudian pada “Hierarchy” window tambahkan sebuah Game Object bernama “RockPair” untuk membuat Game Object dapat dilakukan dengan mengklik kanan pada “Hierarchy” window > pilihlah Creat Empty > ubah nama menjadi “RockPair”. Dan masukkan kedua batu kedalam Game Object tersebut



Langkah 8 : Menggerakkan Obstacle

Tambahkan “Rigidbody2D” pada “RockPair” dan pastikan “Is Kinematic” tercentang agar obstacle tersebut tidak terpengaruh oleh gravitasi.



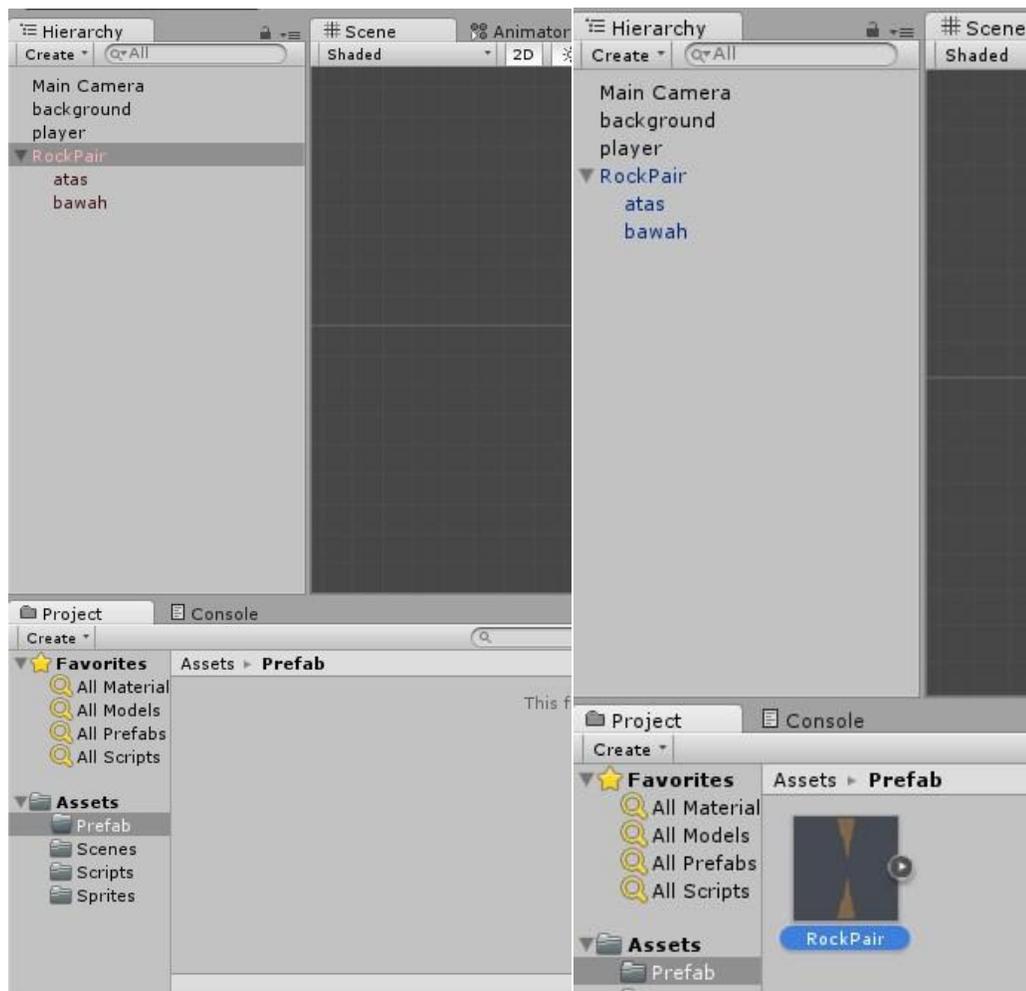
Kemudian buatlah script baru dengan nama “Obstacle” dan pastikan berada pada folder “Scripts”. Dan script tersebut akan berfungsi untuk menggerakkan batu dari kanan ke kiri. Kemudian dalam script tersebut isilah seperti dibawah ini.

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class Obstacle : MonoBehaviour {
5
6     public Vector2 velocity = new Vector2(-4, 0);
7     void Start () {
8         GetComponent<Rigidbody2D>().velocity = velocity;
9     }
10 }
11 |
```

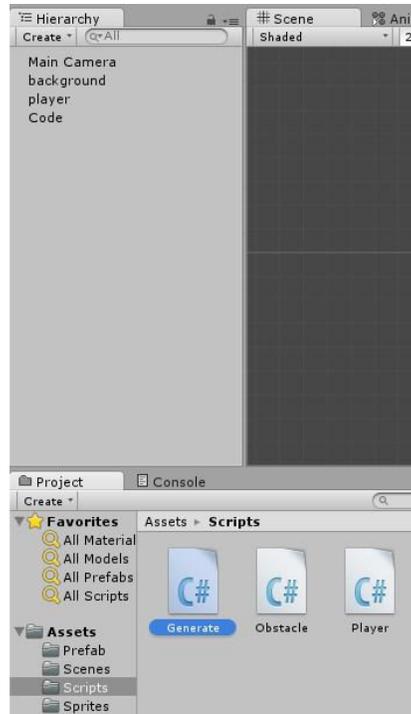
Method “Start” diatas hanya berjalan satu kali saat kita memulai permainan dan dengan code diatas maka seharusnya batu berjalan dari kanan ke kiri dengan kecepatan sebesar X: -4 dan Y: 0. Tambahkan script ini pada “RockPair” dan klik play untuk melihat hasilnya.

Langkah 9 : Mengenerate Obstacle

Kita harus membuat obstacle tersebut terus tergenerate setiap beberapa detik. Untuk memastikan hal tersebut kita harus membuat object “RockPair” menjadi prefab. Untuk membuat prefab kita hanya perlu mendrag object “RockPair” yang berada pada “Hierarchy” window ke dalam “Prefab” folder. Setelah berhasil menjadi prefab maka kita dapat melihat pada “Hierarchy” window warna dari object menjadi biru. Seperti pada kedua gambar dibawah ini :



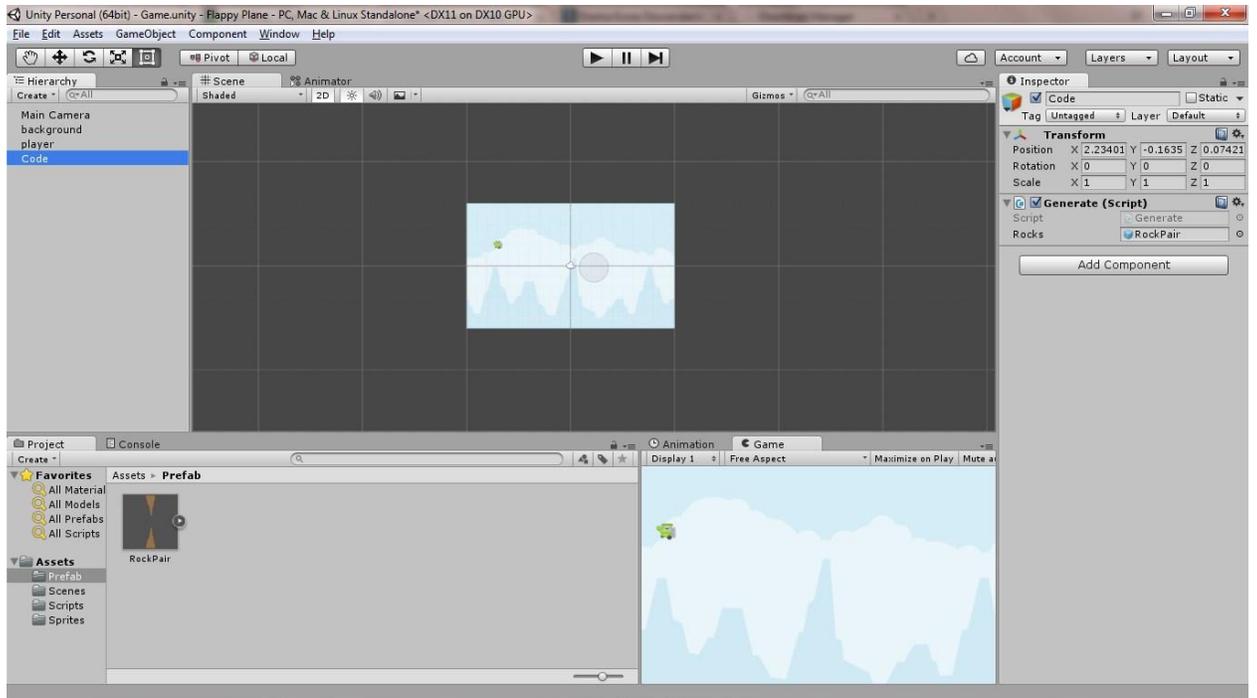
Kemudian buatlah Empty Game Object lainnya dan beri nama “Code”. Lalu buatlah sebuah script baru dan beri nama “Generate”.



Kemudian dalam script tersebut isilah seperti code dibawah ini. “Generate” berfungsi untuk mengenerate obstacle agar terus ada pada lintasan permainan.

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class Generate : MonoBehaviour {
5
6     public GameObject rocks;
7     void Start () {
8         InvokeRepeating ("CreateObstacle", 1f, 1.5f);
9     }
10
11     void CreateObstacle(){
12         Instantiate (rocks);
13     }
14 }
15
```

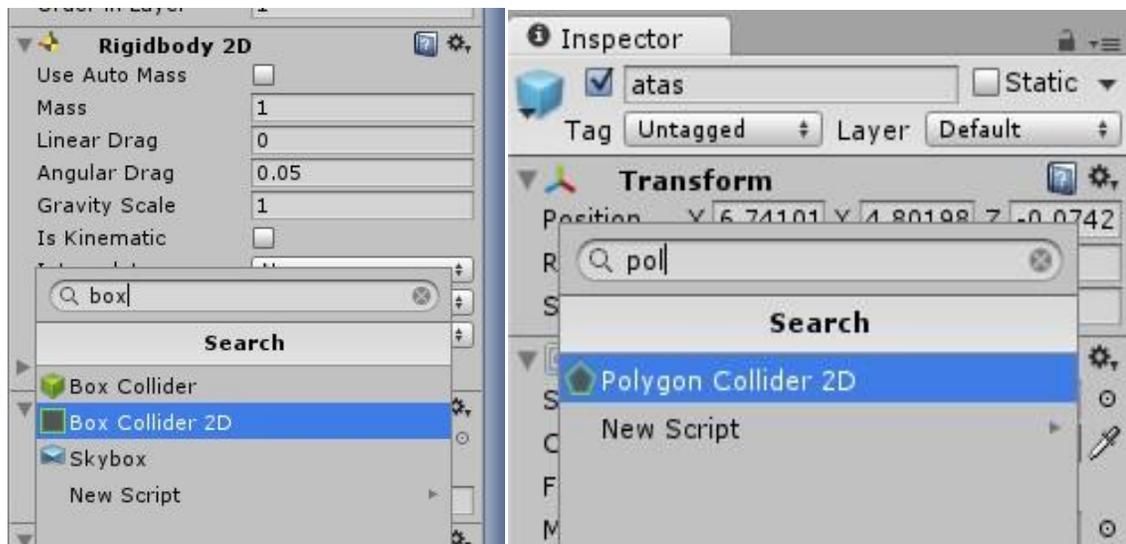
Dalam baris code tersebut kita menggunakan method “InvokeRepeating” yang berfungsi untuk memanggil method “CreateObstacle” setiap 1.5 detik. Kemudian dalam method “CreateObstacle” terdapat method “Instantiate” yang berfungsi untuk mengenerate rocks. Kemudian drag and drop script “Generate” kedalam object “Code”.



Kemudian drag and drop prefab “RockPair” kedalam Rocks yang berada pada “Inspector” window object “Code”. Setelah dimasukkan cobalah bermain untuk mencheck apakah sudah berhasil atau belum.

Langkah 10: Membuat Player Bisa Mati

Setelah kita menambahkan generate obstacle pada langkah sebelumnya. Dan sudah pasti kita dapatkan bahwa tidak ada sentuhan antara player dan obstacle. Yang kita lihat hanya player yang menembus obstacle. Untuk memastikan bahwa obstacle tersebut menghalangi pesawat kita hanya perlu menambahkan “Box Collider 2D” pada Player dan “Polygon Collider 2D” pada Obstacle. Untuk menambahkannya dapat melalui Add Component > Search Box/Polygon Collider2D. Untuk Obstaclenya kita dapat menambahkan melalui prefab kemudian klik tanda kanan yang berada pada prefabnya dan tambahkan di keduanya baik object “Atas” maupun “Bawah”.



Setelah ditambahkan kemudian coba mainkan. Akan terjadi tubrukan apabila Player menabrak Obstacle. Namun yang terjadi sekarang adalah Player tidak mati dan permainan tidak mengulang. Untuk mendapatkan hal tersebut kita harus membuka script “Player” dan tambahkan script dibawah ini :

```

1 using UnityEngine;
2 using System.Collections;
3 using UnityEngine.SceneManagement;
4
5 public class Player : MonoBehaviour {
6
7     public Vector2 jumpForce = new Vector2 (0, 300);
8     void Update () {
9         if (Input.GetKeyDown("space")) {
10             GetComponent<Rigidbody2D>().velocity = Vector2.zero;
11             GetComponent<Rigidbody2D>().AddForce (jumpForce);
12         }
13
14         Vector2 screenPosition = Camera.main.WorldToScreenPoint (transform.position);
15         if (screenPosition.y > Screen.height || screenPosition.y < 0) {
16             Die();
17         }
18     }
19
20     void OnCollisionEnter2D(Collision2D other){
21         Die();
22     }
23
24     void Die(){
25         Application.LoadLevel (Application.loadedLevel);
26     }
27 }

```

Dalam method “Update” kita menambahkan sedikit code yang berfungsi untuk mencegah agar Player tidak melebihi batas atas dan batas bawah layar. Kemudian kita menambahkan dua buah method baru yaitu “OnCollisionEnter2D” dan “Die”. Untuk method “OnCollisionEnter2D” berfungsi untuk mendeteksi collision atau tubrukan antar object sehingga ketika Player menabrak Obstacle maka langsung mati. Kemudian method “Die” berfungsi untuk membuat Player mati dengan cara merestart permainan. Kemudian cobalah bermain dan lihat hasilnya apabila masih ada kesalah ulangi langkah ini.

Langkah 11: Randomness

Sepertinya permainan akan menjadi bosan apabila Obstacle tidak berubah – ubah sama sekali untuk memastikan Obstacle berubah – ubah posisi maka kita menambahkan code pada script “Generate” seperti dibawah ini :

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class Obstacle : MonoBehaviour {
5
6     public Vector2 velocity = new Vector2(-4, 0);
7     public float range = 4;
8     void Start () {
9         GetComponent<Rigidbody2D>().velocity = velocity;
10        transform.position = new Vector3 (transform.position.x,
11                                           transform.position.y - range * Random.value,
12                                           transform.position.z);
13    }
14 }
```

Kita menambahkan sebuah variable baru yaitu “range” yang berfungsi untuk mengatur kerandoman dari Obstacle tersebut. Mungkin untuk memfitkan dengan layar laptop kalian range bisa diubah sesuka kalian. Kemudian lihatlah hasilnya sekarang Obstacle – Obstacle tersebut menjadi acak posisinya.

Langkah 12: Score

Rasanya hambar jika sebuah permainan tidak ada score. Untuk memenuhi hal tersebut untuk menambahkan score kita harus membuka script “Generate” dan tambahkan beberapa code seperti dibawah ini.

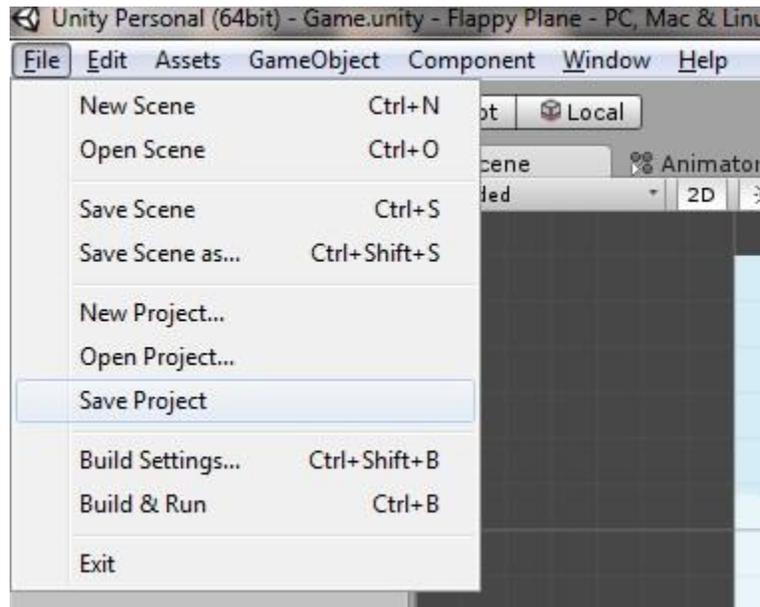
```
1 using UnityEngine;
2 using System.Collections;
3
4 public class Generate : MonoBehaviour {
5
6     public GameObject rocks;
7     int score = 0;
8     void Start () {
9         InvokeRepeating ("CreateObstacle", 1f, 1.5f);
10    }
11
12    void OnGUI(){
13        GUI.color = Color.black;
14        GUILayout.Label ("Score: " + score.ToString ());
15    }
16
17    void CreateObstacle(){
18        Instantiate (rocks);
19        score++;
20    }
21 }
```

Setelah menambahkan project tersebut maka akan tergenerate score pada pojok kiri atas. Kita menambahkan variable “score” dengan tipe int yang berfungsi untuk menyimpan angka score. Kemudian terdapat method “OnGUI” yang berfungsi untuk menampilkan tulisan “Score : x” dan menambahkan “score++” pada method “CreateObstacle” agar setiap kali sebuah Obstacle muncul maka score akan bertambah. Dan seperti inilah hasilnya. Apabila gagal maka coba ulangi langkah ini.



Langkah 13: SAVE PROJECT!

Langkah paling terakhir dan terpenting adalah save project! Hal ini dapat dilakukan dengan menggunakan shortcut “Ctrl + S” atau bisa langsung diakses melalui File > Save Project.



— Selamat Mencoba —