

Welcome to



**University of Lincoln
School of Computer Science
Applicant Workshop**

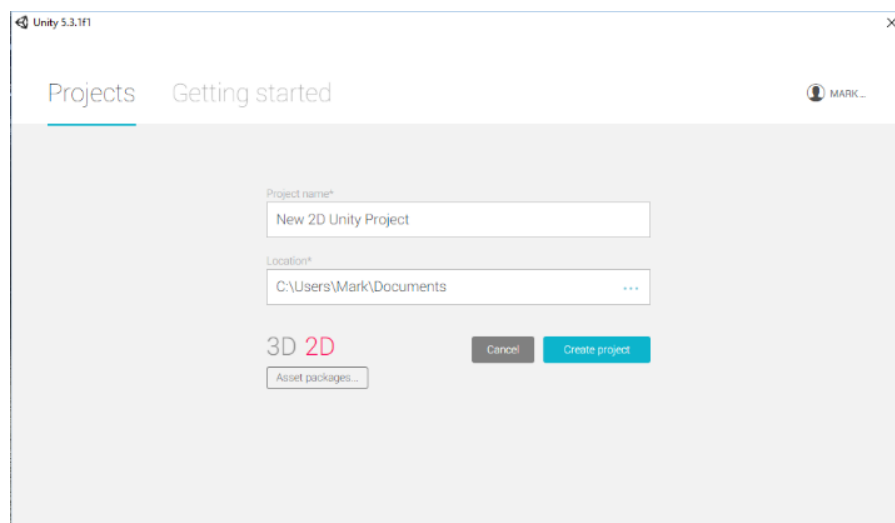
In this session we are going to look at recreating the popular phone app 'Flappy Bird' using the game development engine Unity3D.

This taster session will illustrate some of the features of 2D development in Unity3D. It will also take you through developing a game in 2D.

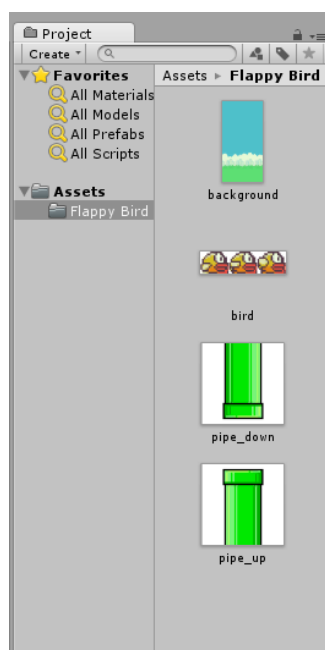
There are many people to help you if you have question about any aspects of this games development.

Task 1 – Adding a Background

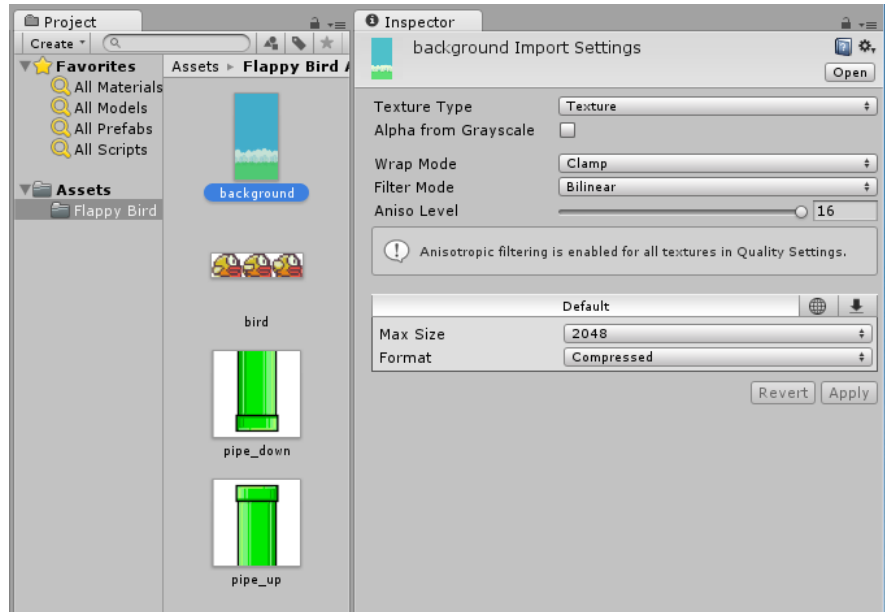
1. Open Unity, Set up a New Project, and set the default behaviours to 2D:



2. In the folder 'Applicant Day' you will find 4 assets or images, Drag these Assets into the Asset window – you should see the four sprites listed.



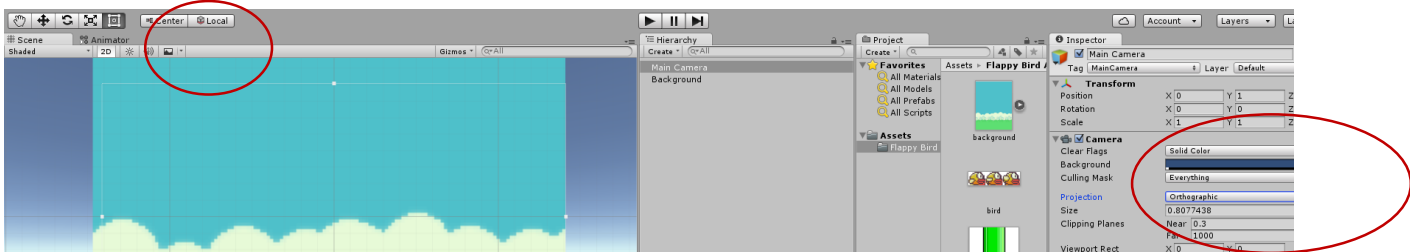
3. Select the 'background' sprite, and see its properties appear in the 'Inspector' window.



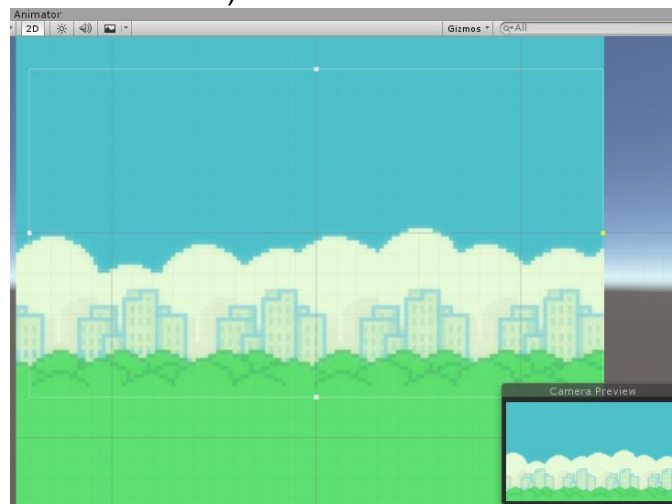
3a. Change the 'Texture Type' from 'Texture' to 'Sprite (2d and UI)', and click 'apply'.

3b. Drag the 'Background' sprite into the hierarchy window, you will see the background appear in your Scene window and the Game window.

Note: Make sure that the '2D' button is pressed in the Scene window bar, and that, with 'Main Camera' selected in the Hierarchy window, the projection setting is set to 'Orthographic'.

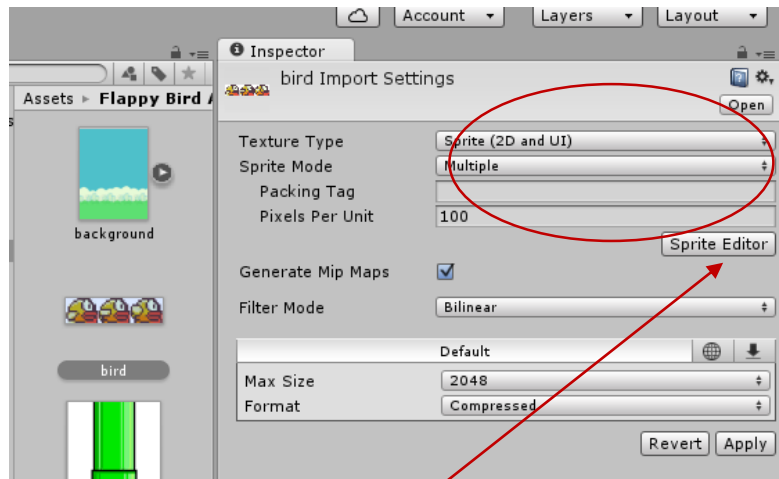


Resize the background sprite in the main window so it fills the game frame (zoom out with the mouse wheel to see what I mean).



Task 2 - Let's add the bird!

4. Select the 'bird' sprite sheet in the Asset window. Set it to be a Sprite (2D and UI), but also set the Sprite Mode to 'multiple':



5. Open the Sprite Editor by pressing the button:



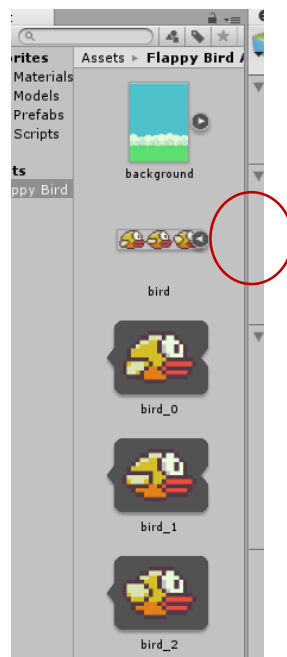
5a. You will see the bird sprites appear as above. We can use the Sprite Editor to 'slice' this sheet up into animation frames.

5b. Select the 'Slice' button, and see the bird automatically split into frames.



5c. Click 'Apply' in the Sprite Editor,

6. Clicking the arrow on the sprite in the asset window will show all of the individual sprites that have been sliced up:



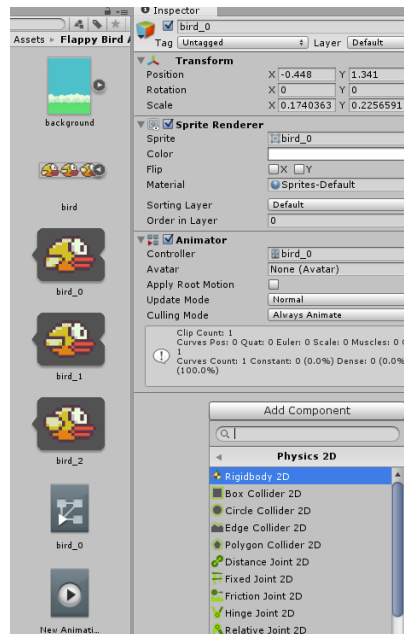
6a. Select all three frames of animation (use the CTRL key to multiple select), and drag them onto the scene. You will be prompted to save a New Animation – just agree.

6b. Playing the game now, you should see the background and the bird sprite (stationary) but flapping its wings...!

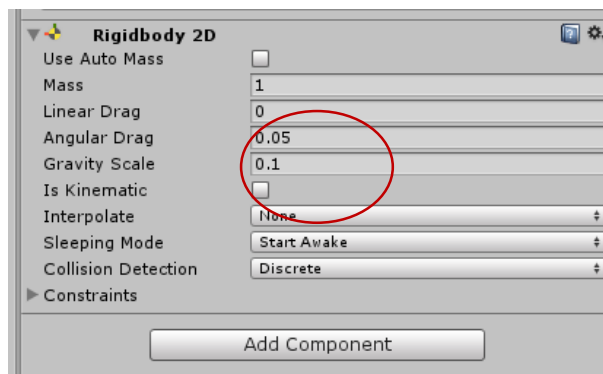
Task 3 – Movement!

Let's give the bird some movement and respond to gravity.

7. To make it act under gravity is easy in Unity. With the bird selected in the Hierarchy, add a 'component', and select the 'Rigidbody2D' component, within the Physics2D collection.



8. The 'Rigidbody2D' component will now be showing in the Inspector for the bird. Change the 'Gravity Scale' setting, from 1 to something smaller – say 0.1. Run the game and see what happens!

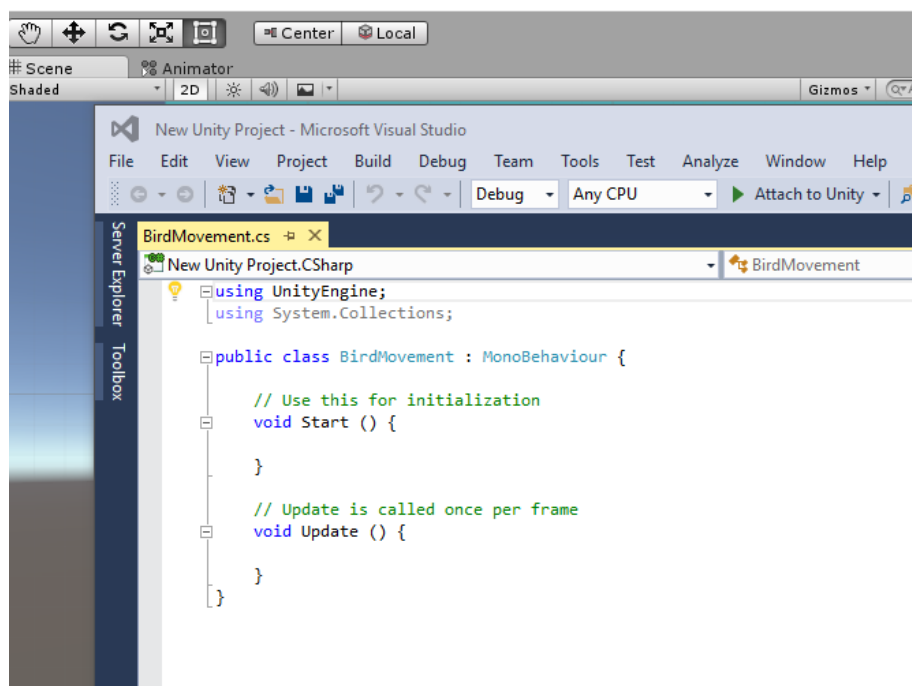
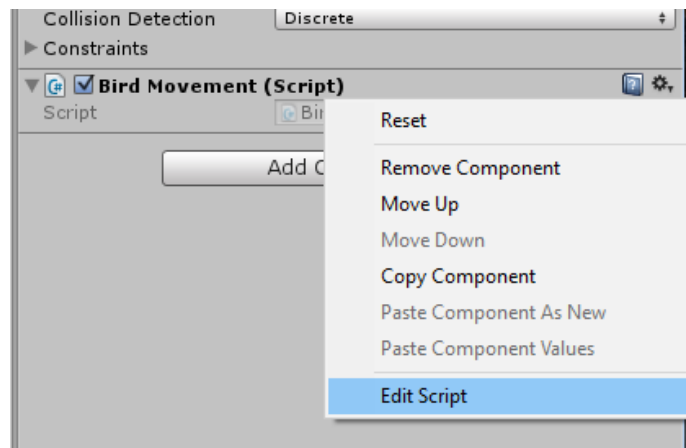


9. To make the bird move sideways, we have to add a '**script**'. This is a developers way of telling Unity what exactly to do with an asset – by using programming. We will be using C#

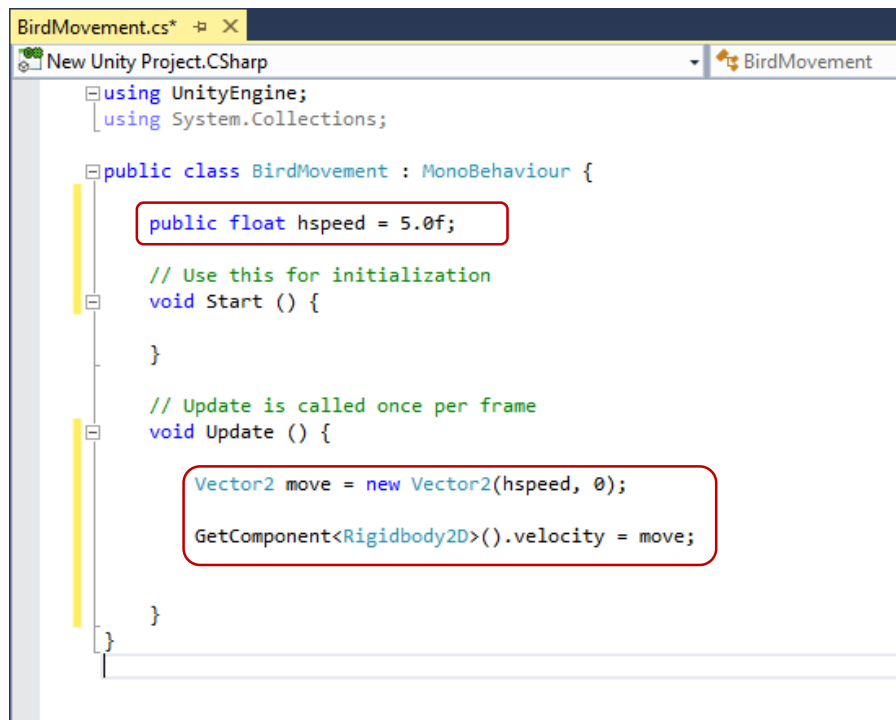
10. With the bird selected, press 'Add Component' and select 'New Script'. There is a space for naming the script – call it 'BirdMovement'. Then press 'Create and Add'.



11. You will see the script component appear in the bird's Inspector window. Selecting the 'cog' on the right hand side of the component will bring up a number of options – select 'edit script' as we want to start to add to it. You should start to see Visual Studio start up and show the component.



The 'Start' function runs once when the script is started. The 'Update' function is run each frame of the game.



```
using UnityEngine;
using System.Collections;

public class BirdMovement : MonoBehaviour {

    public float hspeed = 5.0f;

    // Use this for initialization
    void Start () {

    }

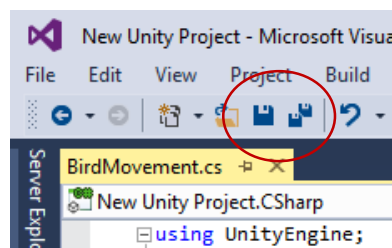
    // Update is called once per frame
    void Update () {

        Vector2 move = new Vector2(hspeed, 0);
        GetComponent<Rigidbody2D>().velocity = move;

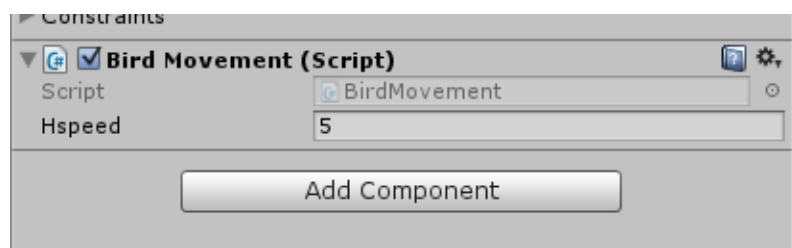
    }

}
```

12. Add the lines of code shown above. When you have finished, leave the window open and just save the script file:



The variable '**hspeed**' will now be visible in Unity in the script component – allowing it to be changed! We don't have to keep going to Visual Studio to change this number, we can do it within Unity!

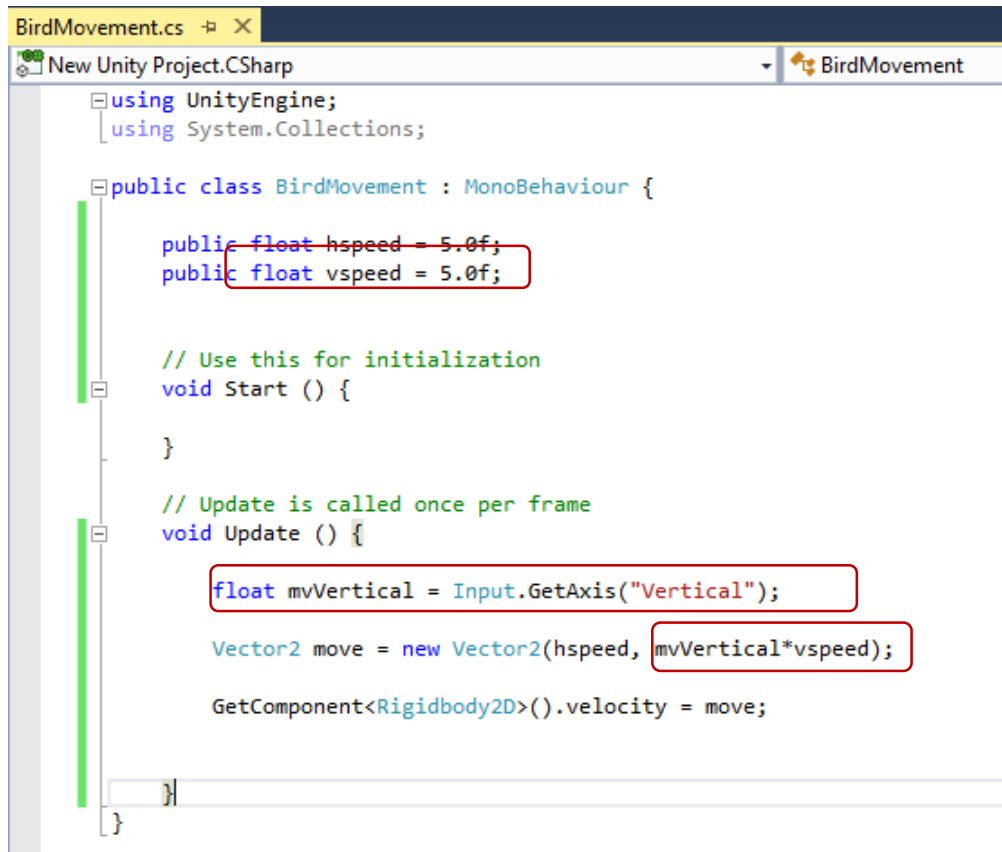


5 is too fast (try it!). Set it to something like 0.1 and see the difference when you run the game.

Task 4 – User Input

Let's make the bird respond to some user input.

13. In your script file (in Visual Studio, which you should have still open), we need to add some more lines of code:



```
BirdMovement.cs
using UnityEngine;
using System.Collections;

public class BirdMovement : MonoBehaviour {

    public float hspeed = 5.0f;
    public float vspeed = 5.0f;

    // Use this for initialization
    void Start () {

    }

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

        float mvVertical = Input.GetAxis("Vertical");

        Vector2 move = new Vector2(hspeed, mvVertical*vspeed);

        GetComponent<Rigidbody2D>().velocity = move;
    }
}
```

The '**GetAxis**' function gets the up/down/left/right key which is input by the user.

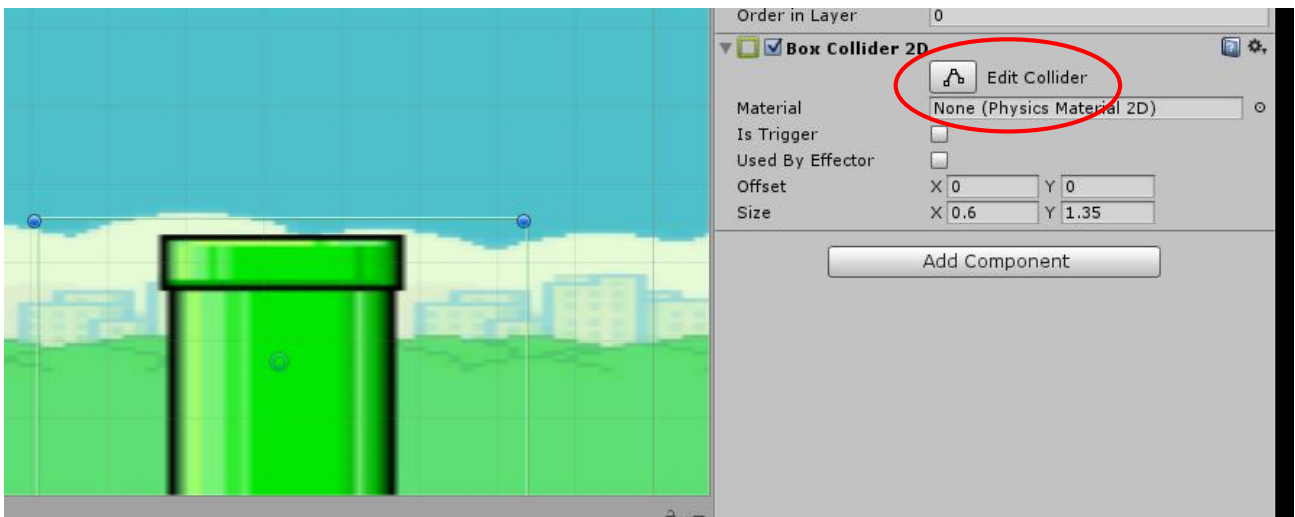
14. Now play the game and see how the arrow keys control the bird. Modify the Gravity value and the **hspeed** and **vspeed** values to see how this changes the game.

Task 5 – Additional Task

To make the game more fun, we need to add obstacles to avoid, try adding some of the Pipe sprite to the game.

Try adding some collision detection, this way we can start interacting in our game.

After adding some of the Pipes, click on one of the pipes and select 'Add Component', from the list select Physics 2D -> Box Collider 2D



Click Edit Collider and Resize the Collision Box to fit the Pipe.

Now repeat the process but this time adding a Sphere Collider 2D to the Bird object.

Once this is done you will need to add some final code to the Bird Script, add the code below and play your game.



Now when you crash into the pipe, it should vanish!

Try adding lots of pipes and playing about!

End of Workshop!