

An Introduction to 2D in Unity 5

Introduction

This will give you a brief introduction to starting a 2D project in Unity. Creating a project, game objects and controlling them using the Unity interface and also with C#.

Preparation

Install Unity

You will need to have **Unity** downloaded and installed. If you don't already then visit this link and follow the instructions: <https://unity3d.com/get-unity/download>

Resources

- [The difference between 2D & 3D in Unity](#)
- [Overview, terminology and gameplay in 2D](#)

Tutorial

Create a New Project

- Open Unity 5
- Create a new project: `File > New Project`
- Give your project a **name**, and a **location**, and select the **2D** option

TwxYDBh8gX.gif

Your screen should look like this:

Unity 2D - New Project

Change layout

If you're layout doesn't look like this then select **layout** in the top right corner, and select **2 by 3**:

Change Layout 2x3

Creating a Main file

First we need an Empty Game Object. You can do this one of two ways:

- Option 1: On the menu bar select: `GameObject > Create Empty`
- Option 2: In the Hierarchy tab, click: `Create > Create Empty`

Create Object

Once you have done this you will see a GameObject in your Hierarchy window. Select this, and then in the Inspector tab, change it's Name from GameObject to Main:

Change GameObject name

Adding C# script to the GameObject

Now we are going to add some code to the Main GameObject.

- In the Project tab, select: `Create / C# Script`
- Rename the script to 'Main'
- Drag the C# Main file from the Project tab onto the Main GameObject in the Hierarchy tab
- With Main in the Hierarchy tab select, check the Inspector tab to see if the script has been added

Create C# script and add to Main object

Adding code to C# script using MonoDevelop

Double click on the Main script in the light grey box in the Inspector tab, this will open MonoDevelop and we can start adding some code:

Open script in MonoDevelop

The function `Start()` will run once when the GameObject is created, and then `Update()` will run once per frame. We can test this using `Debug.Log()` for now:

See the Anatomy of a Script section on [this page of the Unity documentation](#) for more information about the `Start()` & `Update()` functions.

Edit the Main script in MonoDevelop to appear as follows:

```
using UnityEngine;
using System.Collections;

public class Main : MonoBehaviour {

    // Use this for initialization
```

```
void Start () {  
    Debug.Log ("This is the Start function");  
}  
  
// Update is called once per frame  
void Update () {  
    Debug.Log ("This is the Update function");  
}  
}
```

Select save and go back to the Unity window. In order to see the text outputted the console window needs to be opened. In the menu bar select: `Window > Console`. It will open as a separate window, but by dragging the tab we can move it underneath the Inspector tab, like this:

Snap console tab to interface

Now click the play button to see the `Debug.Log()` text in the console:

See logs in console

Save your Scene

Now is perhaps a good point to save the Scene we are currently working on:

- `File > Save Scene`
- Rename the Scene to 'Scene1' and click **Save**

Save Scene

Adding a Sprite

In this example our sprite will contain a white square, with equal width and height. Either create one yourself using graphics editing software (e.g. Photoshop or Gimp) or you can use the link below which generates a square white image of 400 x 400 pixel:

<http://placeholder.it/400x400/f5f5f5/ff00ff/>

Save your white square to the Desktop or somewhere you can easily access it.

Go back to the Unity Editor and either:

- Option 1: On the menu bar select: `Assets > Import Asset`. Then find the white square image/asset.
- Option 2: Drag the file directly into the assets folder in the Project tab.

In either case the Project tab should now contain the square image:

Import Asset - square image

Click on Square in the Projects tab and in the Inspector you can set how many pixels there are per unit. If we set this the same as the size of the square image then this square is one unit in the game system.

If this window pops up, click Apply.

Pixels per unit

Now drag Square from the Project tab to the Hierarchy tab and rename it 'RedSquare' in the Inspector tab.

Then in the Inspector tab use the color option in Sprite Renderer to change it to red:

Rename asset and change colour

Both in the Scene and in the Inspector tab it is possible to manipulate the **Position**, **Scale** and **Rotation** of GameObjects.

With RedSquare selected in the Hierarchy, play with both these methods to see what you can do:

Transform GameObject using SceneTransform GameObject using Inspector

You can always reset the position, rotation and scale, by clicking the cog in the top right of the Transform box and selecting reset.

Revision #20

Created 3 August 2016 14:48:04

Updated 19 January 2020 17:58:32