Unity User Guide

About

This is a Unity User guide

I will start at the beginning and work my way through the variations avenues you could have about your game. I will go through 2D and 3D games along with general tips and tricks for solo and team-based projects. Without future ado, let's begin!

Index

1. Installation

Before installing, note that Unity is completely free forever, if you or your company makes less than $100 000 (USD).

Follow the link here and download the free installer (the plus version is definitely not necessary). Follow the instructions and download the installer. The installer is simple and easy however, if you run into trouble, go to this video for Windows. Also note that you will may need a Unity account or just use your Google/ Facebook account to sign in.

If you want a new version of Unity, go to the Unity Hub and click the **Installs** tab on the left and **add** (top right) the desired version. Afterwards, the installation will take quite a while, even with a fast internet connection.

When installing Unity, it might ask you to install **Visual Studio** along-side Unity. If you have Visual Studio, do not install again, it should automatically detect the current VS on the system. Otherwise, it is optional and I've heard that MonoDevelop (comes with Unity) is good enough for handling the task Unity puts forth.

2. General

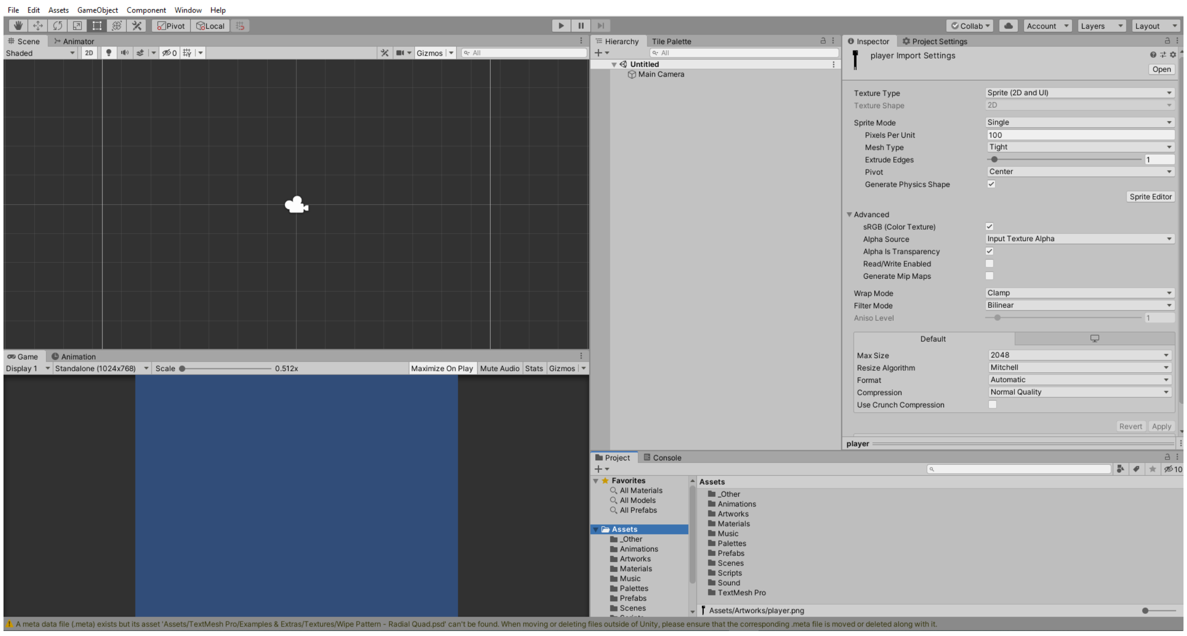
Unity is a 3D game engine built in **C#** but can be used for 2D. Before you freak out about programming, let me assure, you it is quite trivial. If you have programmed before especially in C++, it is a huge plus. If you have not, don't worry, it's just a bunch of copy and pasting. The most important thing is don't be frustrated and

```
Debug.Log() // logs to Unity console
```

is your friend. Also organize your scenes by naming them or using empties as folders (+1 organization). Now, it is time to pick, 2D or 3D. (As a beginner, start with 2D)

3. Unity Editor Overview

Unity is a panel based application. These panels can be rearranged however you like and saved as default.



player Import Settings

Texture Type	Sprite (2D and UI)
Texture Shape	2D
Sprite Mode	Single
Pixels Per Unit	100
Mesh Type	Tight
Extrude Edges	1
Pivot	Center
Generate Physics Shape	✓
Advanced	
sRGB (Color Texture)	✓
Alpha Source	Input Texture Alpha
Alpha Is Transparency	✓
Read/Write Enabled	
Generate Mip Maps	
Wrap Mode	Clamp
Filter Mode	Bilinear
Aniso Level	1
Default	
Max Size	2048
Resize Algorithm	Mitchell
Format	Automatic
Compression	Normal Quality
Use Crunch Compression	

```
A meta data file (.meta) exists but its asset 'Assets/TextMesh Pro/Examples & Extras/Textures/Wipe Pattern - Radial Quad.psd' can't be found. When moving or deleting files outside of Unity, please ensure that the corresponding .meta file is moved or deleted along with it.
```

This is my default setup. The **Scene** panel is the most important viewport. This is where you modify the scene. The **Game** scene only turns on if you hit play. It is a preview of what the user will see if you build the current scene.

On the right hand side, we have the **Project** directory (folder) panel on the bottom. This is where all your assets are (ie art, music, scripts, etc.). The **Hierarchy** is like a layers panel in Photoshop. It is an arrangement of all elements in the scene. For every element of the scene can be hidden from the view by hitting eye button when hovering on the element. The lock button is the pointer icon. The **Inspector** contains all modifiable aspect of an element along with info. This is where you drag scripts in the element.

Other panels include a console, which will output when the game is run. The **Animator** and **Animation** go hand in hand and creates animation for sprites.

4. 2D

4a General

So you've decide to make a 2D game. Great! Who needs modelling and lighting (It's a lot of work anyways)?

2D games are secretly 3D, what does that mean? Think of Unity 2D as a bunch of layers like in Photoshop, Gimp, After Effects, or Illustrator. The closest to the camera is picked up first and blocks the ones behind it. As a 2D world, lighting is global (unless you want enable an experimental local lighting feature). To start a new 2D game, click **New** and select 2D on the pop-up screen and use the desired directory (or follow my cloning guide so to not start from starch). This may take some time, but after Unity finishes installing itself, we can get started. Now let's make a player and make it move!

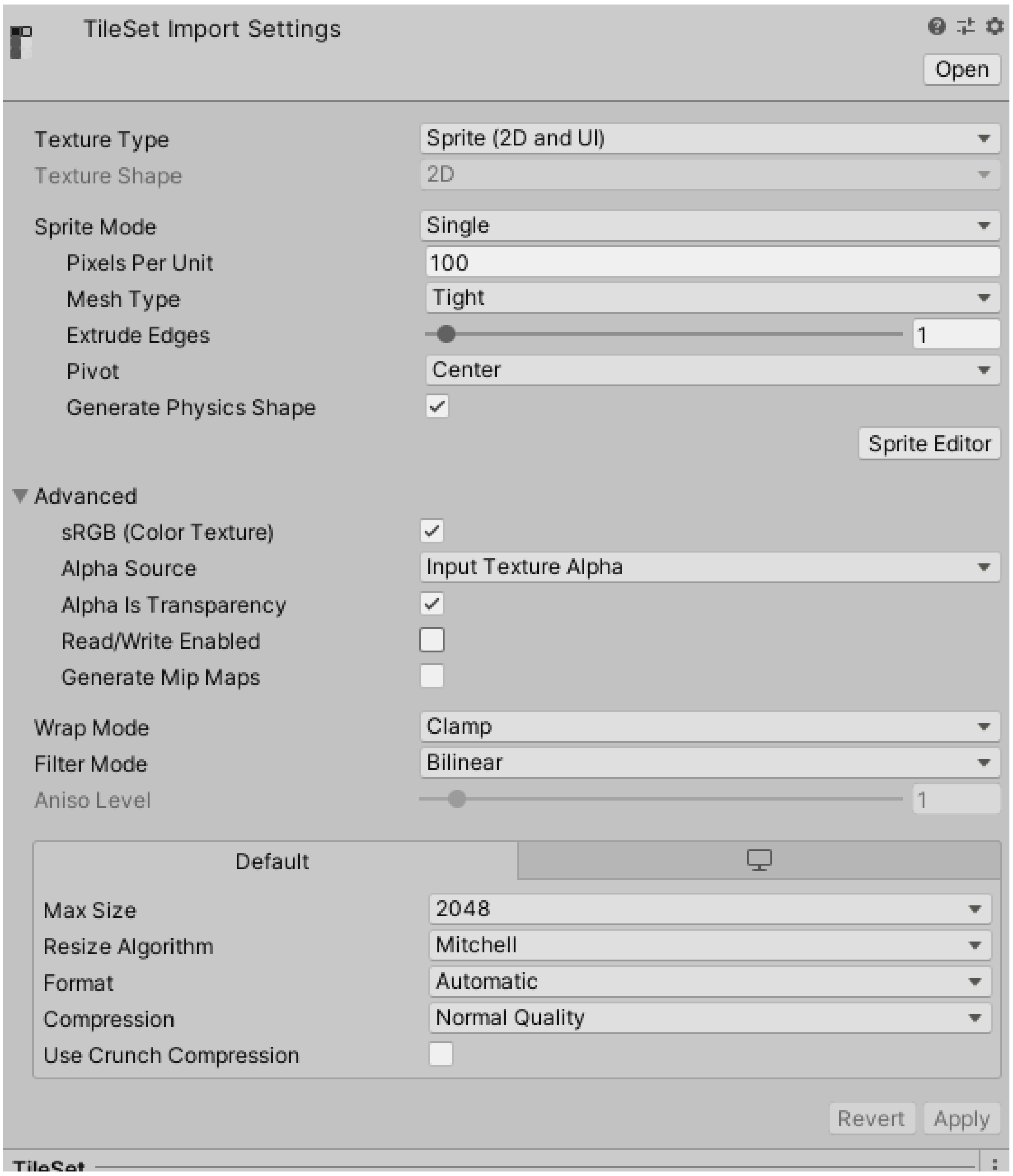
*Note, part of the tutorial (2D) follows this repo: Unity-Template-2D-2019_3

4b Sprites

Sprites are what make up the visuals of the game. These could be png or jpeg (recommended because of small size) file. You can make your own and drag them into an **Artworks folder**.

If you have multiple sprites on a picture file, you have what is called a sprite sheet. These could be useful because it saves space. Unity comes with a sprite editor that can cut the sprite sheet into multiple sprites.

However, before we get to that, let us take a look at the Sprite setting



TileSet Import Settings

Texture Type	Sprite (2D and UI)
Texture Shape	2D
Sprite Mode	Single
Pixels Per Unit	100
Mesh Type	Tight
Extrude Edges	1
Pivot	Center
Generate Physics Shape	✓
Advanced	
sRGB (Color Texture)	✓
Alpha Source	Input Texture Alpha
Alpha Is Transparency	✓
Read/Write Enabled	
Generate Mip Maps	
Wrap Mode	Clamp
Filter Mode	Bilinear
Aniso Level	1

Default

Max Size	2048
Resize Algorithm	Mitchell
Format	Automatic
Compression	Normal Quality
Use Crunch Compression	

From the top header, we can see some info on the item

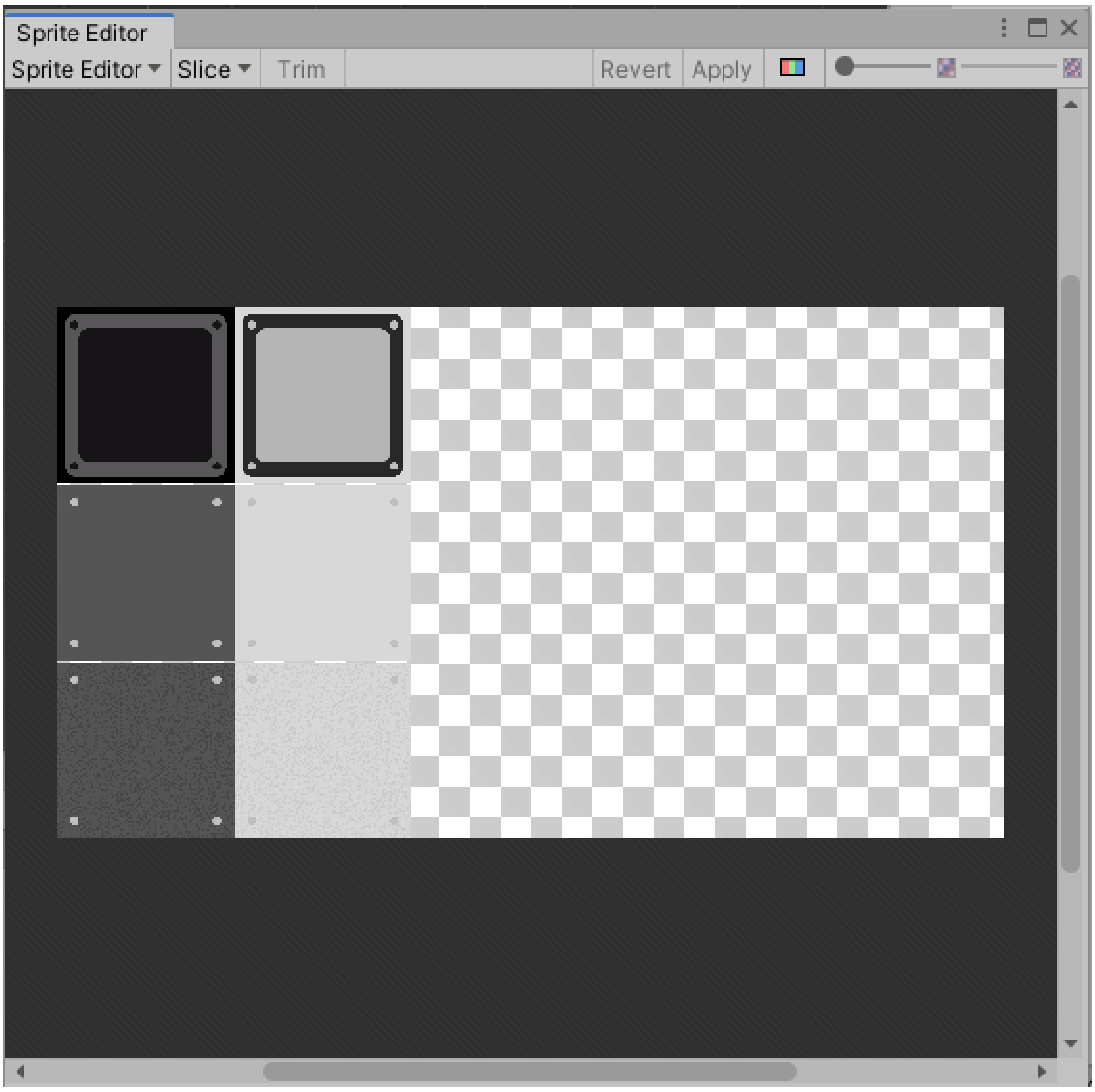
From the header we also see:

- Texture Type - Usually will only use sprite (2D image) or normal map (for 3D depth)
- Sprite mode - as single texture or sprite sheet
  - Adjust the pixels per unit (Pixels per unit should be 1-1) (In my case, it is 360 px per tile)
  - Pivot - where the center of object is
- Wrap Mode - how the image is displayed (ie repeated or clamped(/cut) at the board)
- Filter Mode - Point (equivalent to pixel perfect or nearest neighbour in Photoshop) or Bilinear (natural scaling with edge softening)
- Format - Format of image (8bit, 16 bit, 32 bit colour with or without alpha)

*Note, if a setting was not mentioned, it is not that important at the beginner level.

Multiple Sprites

1. Select Multiple in **Sprite Mode**
2. Click **Spirit Editor** button



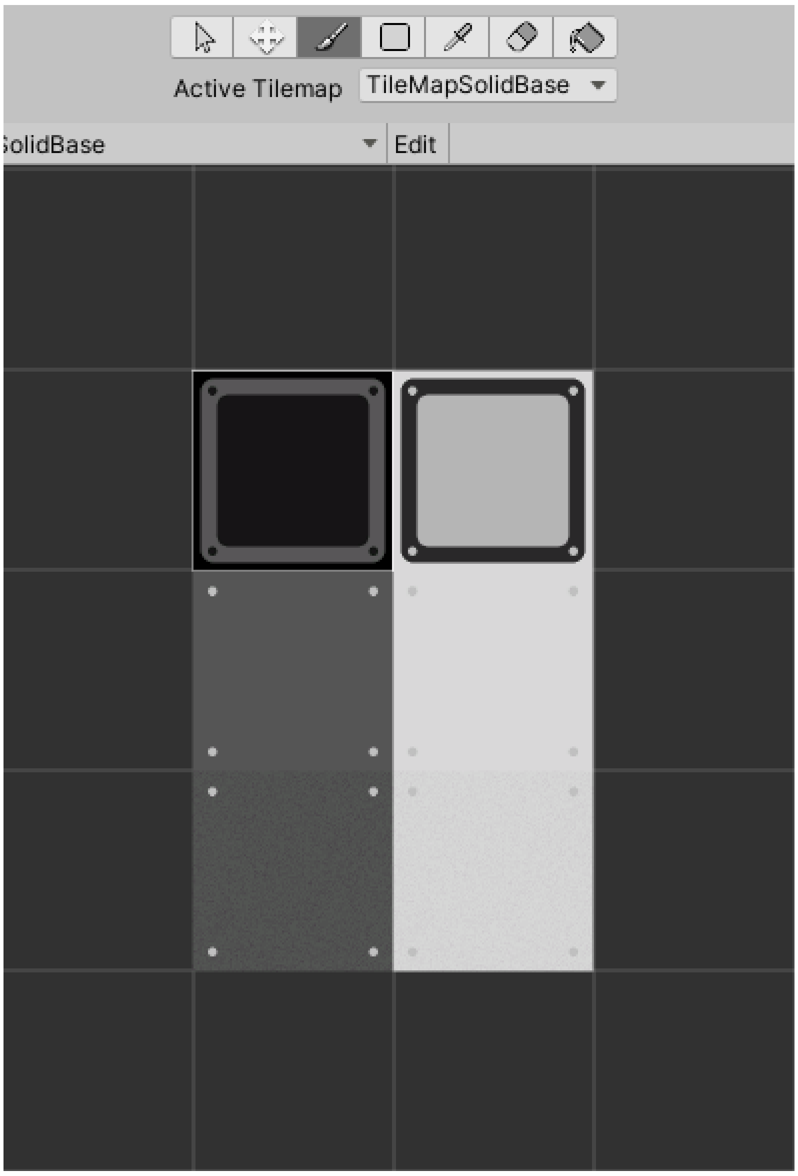
3. In sprite editor, click Slice -> Slice (This is an automatic slicer and is usually good enough, if not, manually move the transform bound boxes to fit or define split at distance apart (must change to **Grid by Size Count**, not **Automatic**) with the size of tile in **Pixel Size**.
4. Click apply
5. Close window or drag to some widget
6. You can inspect every tile by expanding the original image and a bunch of tiles with name pictureName_0 to # of tiles with be a child of it

Now, with a sprite, you can drag it into the scene

To create a tile map, follow the following:

1. GameObject -> 2D Object -> Tile Map
2. Open Tile Palette, by going to Window -> 2D -> Tile Palette
3. Create New Palette and save it
4. Drag all individual tiles into widget or drag parent spritesheet (and save)

Now with the tile palette, you can draw on the scene, reorganize everything and many other things



From the top, there are many icons, we will go through each of them

- Curser - selects tiles from scene
- Move - Moves tiles in palette (Only if you click **Edit**)
- Brush - paints on scene
- Square/Rect - Selects multiple tiles or one to paint from
- Eye dropper - Selects tile from scene
- Eraser - Erases
- Paint Bucket - Floods with active brush (Like Photoshop)

To layer tiles,

- Right click on the tile map -> 2D -> Tilemap for however many more layers you need

To set a layer as a solid with collision,

1. go to the layer you want to make solid
2. Add Component -> search add Tilemap Collider 2D

Resources: [Brackeys]

4c Movement

Movement is critical in all games, whether the movement is limited to left or right, or games that in 3D. On the internet there are may sources that claim the perfect jump, but only you can decide that based on the gameplay.

To start off with;

1. make a new folder called **scripts**
2. right click -> Create -> C# Script

Next, you want to open the script by double clicking on it

This will either open the script in MonoDevelop or Visual Studio (to find out more about scripting, go to the scripting section)

https://www.youtube.com/watch?v=dwcT-Dch0bA

4d Camera

When implementing a camera, there are two options, however, we will through the harder option first to get use to how cameras work

1. We will create a camera follow script

   OR

2. Use the Cinemachine Extension

Resource: https://www.youtube.com/watch?v=MFQhpwc6cKE or https://www.youtube.com/watch?v=2jTY11Am0Ig

4e Prefab

A prefab is simply a clone that can be dragged to the scene

4f Particle System

A particle system adds an extra layer of immersion that is quite simple in Unity.

4g Collectibles

There are two parts to a collectible,

a. collection of the item

b. the storage of the item in inventory

4h Animation

Animation is critical in creating any game that does not look static. In Unity, animation is handled by

4i Events

4j Pixelated

5. 3D

5a General

3D, it's where we live. Good for you for deciding to go 3D. Before we go any further, it is not recommended that you start off with a 3D project if you just started game design.

5b

6. Title Screen

One of the last things to do is to create a title screen

7. Scripting

Unity is based on C# and is very much a c-styled programming language. That means it resembles programs that are written in C, C++, F#, or Java.

The main thing to know though is Unity's implementation of C#

If you create a script, automatically, it will be drawn from a template that looks similar to the following:

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

public class movement : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {

    }

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

You will notice that at the top there are imports from Unity's other libraries using the **using** keyword. Afterwards, Unity defines the script as a class*. The class is defined as public so other scripts and Unity can assess and utilize this class. The name you gave the script will follow the **class** keyword. Afterwards, **MonoBehaviour** is defined as what movement inherits from. Think of this as your script **extending** from class MonoBehaviour.

Below that and indented, we see void Start() and void Update(). As the comments suggest, void Start() occurs before the first frame update and void Update() is called once per frame. Note **void** means there will not be a return value (ie no return statement).

Also, there are different methods of MonoBehaviour such as **FixedUpdate()** that may update more or less than once per frame. This will be important for any physics related functions

* A class is a data structure that holds a collection of information on the class (ie, its attributes and methods).

A list of other functions and methods that can be used can be found on the Unity Documentation site, however, the most important thing about learning and debugging Unity is to use

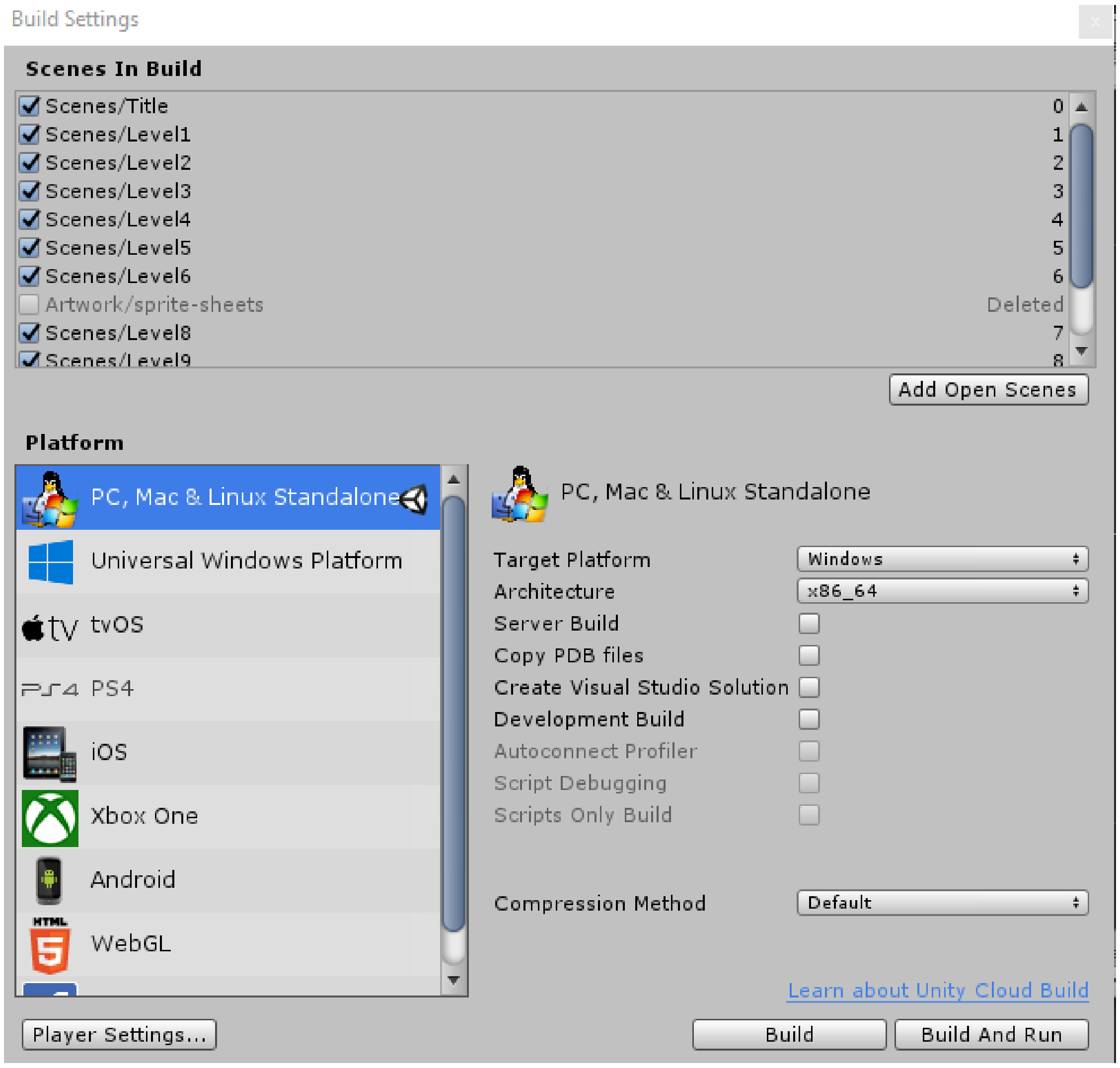
```
Debug.Log() // logs to Unity console
```

When using this, after running the game, it will output to Unity's debug console

8. Build

So you are finished your game. You need to disturbed the game. This is done through building the game.

1. Go to **File**, **Build Settings...**
2. Select the scenes that you want to build


Build Settings

Scenes In Build

✓ Scenes/Title	0
✓ Scenes/Level1	1
✓ Scenes/Level2	2
✓ Scenes/Level3	3
✓ Scenes/Level4	4
✓ Scenes/Level5	5
✓ Scenes/Level6	6
Artwork/sprite-sheets	Deleted
✓ Scenes/Level8	7
✓ Scenes/Level9	8

Platform: PC, Mac & Linux Standalone

Target Platform	Windows
Architecture	x86_64
Server Build	☐
Copy PDB files	☐
Create Visual Studio Solution	☐
Development Build	☐
Autoconnect Profiler	☐
Script Debugging	☐
Scripts Only Build	☐
Compression Method	Default

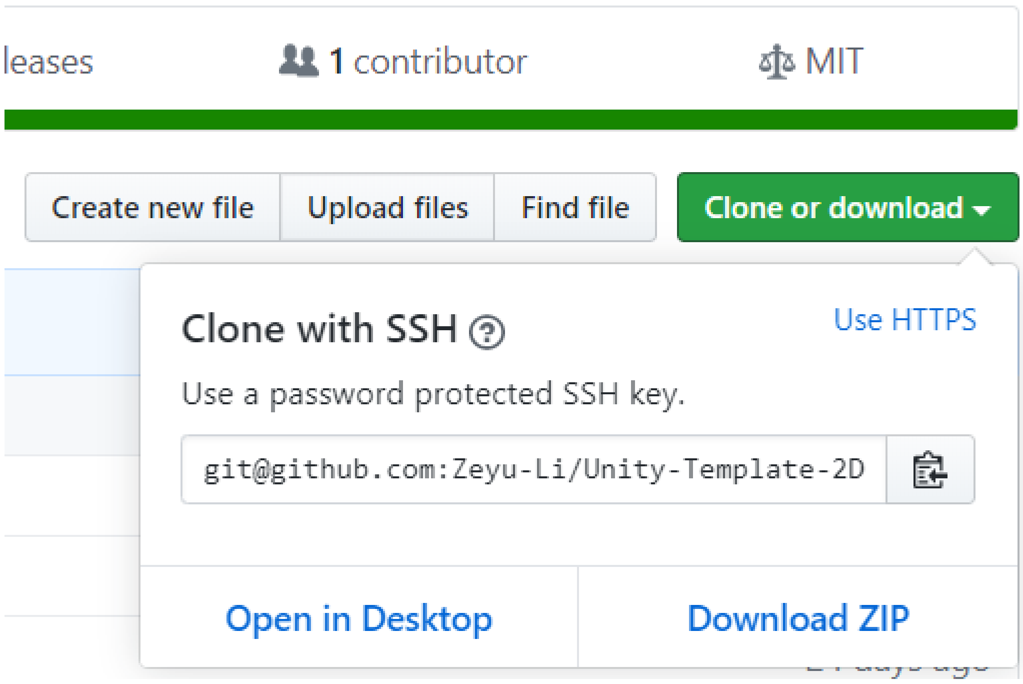
Add Open Scenes
PC, Mac & Linux Standalone
Universal Windows Platform
tvOS
PS4
iOS
Xbox One
Android
WebGL
Learn about Unity Cloud Build
Player Settings...
Build
Build And Run

3. Select the desired **Platform**
4. You can customize the company name, product name, icon, and cursor
5. **Build** (**And Run** if you want to run it)
6. Select the folder and wait for it to build

9. Cloning Guide

From my GitHub:

1. Find the right Unity Repo to clone from my account (ie Unity-Template-2D-2019_3, Unity-Template-3D-2019)
2. Download ZIP after clicking Clone or download



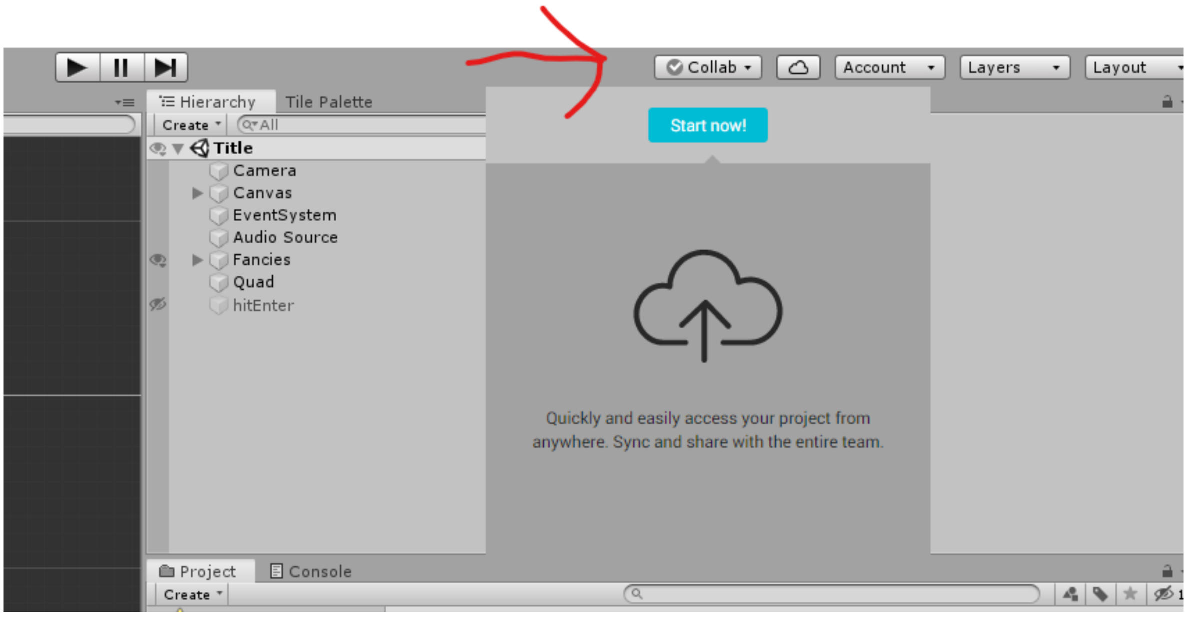
3. Unzip and move to desired directory
4. Go on Unity Hub and click **Add** and located to directory
5. The project should appear in Projects and it is done
6. Click on the project to open it

10. Collaboration

Unity's Collab

Collaborating will friends can be tough will Unity, especially with git/GitHub, but it you are careful, things could turn out fine.

If you need to collaborate, the best way is to use Unity's Collab feature



However, note that this only supports 4 people (which is usually enough because only people that will contribute to the codebase should need it)

GitHub

Otherwise if you are a masochistic and decide to GitHub, here are some general practices:

1. Everyone should work on there own branches and work on there own scenes working towards **Prefabs** (ie one person on movement, one on moving platforms, etc.)
2. Optionally, have one person on the master (final) level to peace it together
3. Use this gitignore or clone from my projects on GitHub
4. Have the core gameplay done first (especially at a Game Jam)
5. Get to know Git/GitHub and how version control works
6. Remember you can go back to a previous working version
7. Have fun and don't fight with your team members

Back to Top

11. Resources

- The Unity User Manual provides some great documentation with code that can for the most part be copied and pasted
- Brackeys YouTube channel
- Lynda.com - If you have a library card, there is a high probability that you have access to Lynda with their organization deal
- Google is your best friend
- Ask on Reddit or Stack Exchange, don't worry they won't bite