Slice IT !

By Joseph Grados, Ashley Mendez, Anaya Yorke, and Gregory Smith
Georgia Gwinnett College
TAP
Fall 2022| STEC 4800

Technology Ambassadors Program

Members:

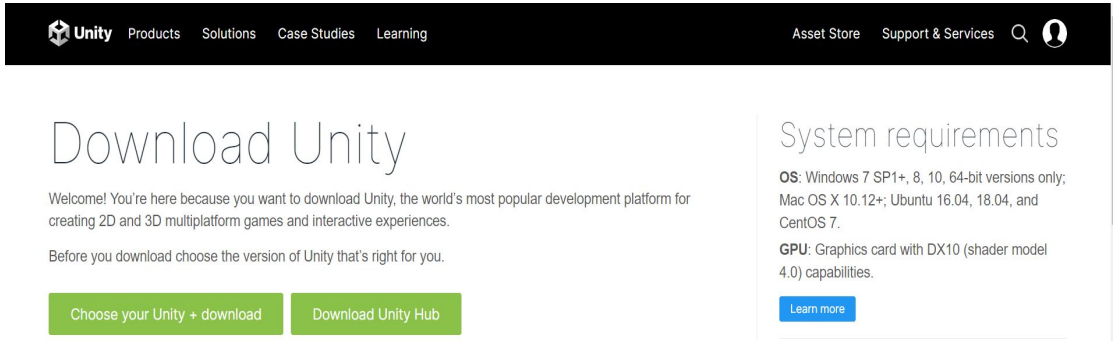
- Joseph Grados
- Ashley Mendez
- Gregory Smith
- Anaya Yorke

What do we do in TAP:

- Expo (to showcase our game)
- CREATE
- Workshops

Unity

- A software development application designed to create other application for a multitude of different hardware like PC, mobile devices, and video game consoles.
  - C#
  - Visual Studio Code
  - Unity Document
  - Works On:
    - Windows (32 or 64 bits)
    - MAC
    - Linux



Now let build our levels!

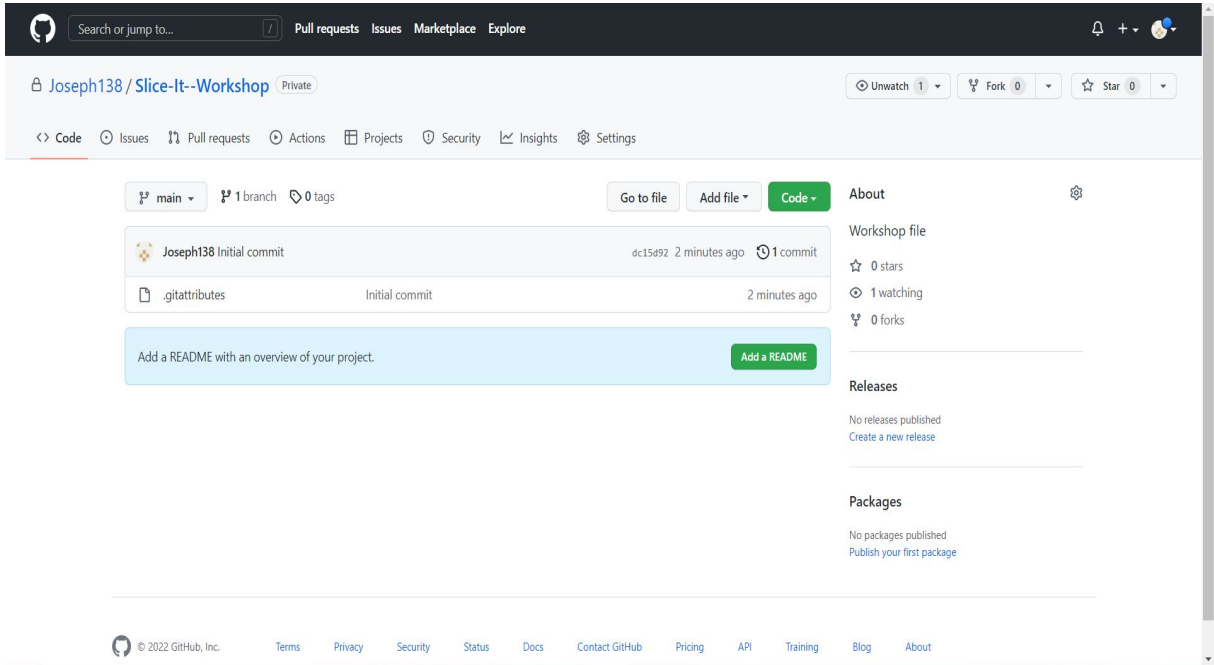
Step 1: Download Unity and Unity Hub

Access from D2L

Let’s make levels

Step 2: Unzip the Slice It game file

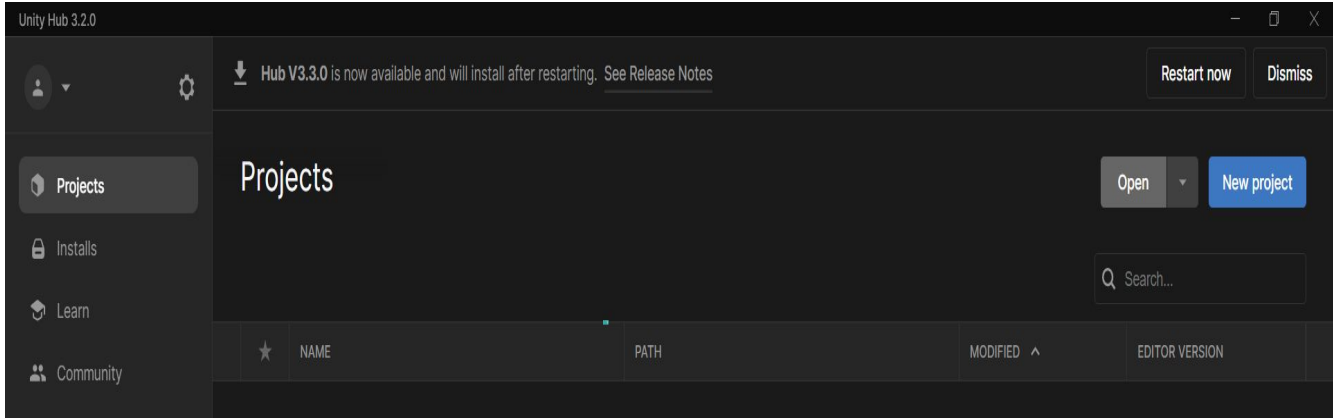
*Access from Github



Let’s make levels

Step 3:

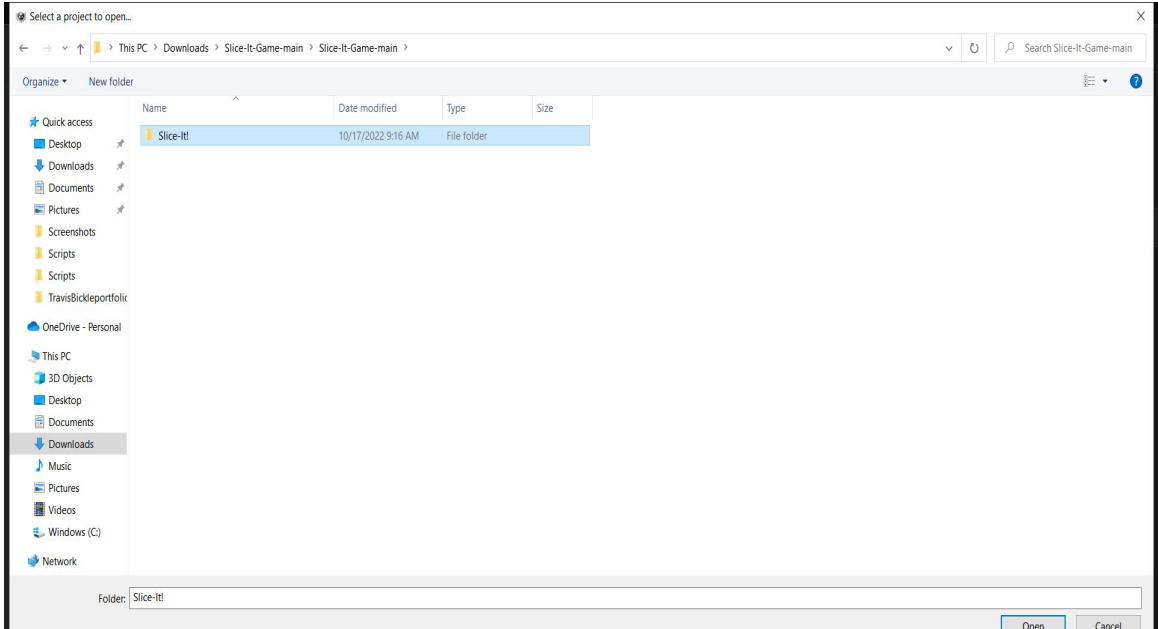
Click Open in Unity Hub



Let’s make levels

Step 4:

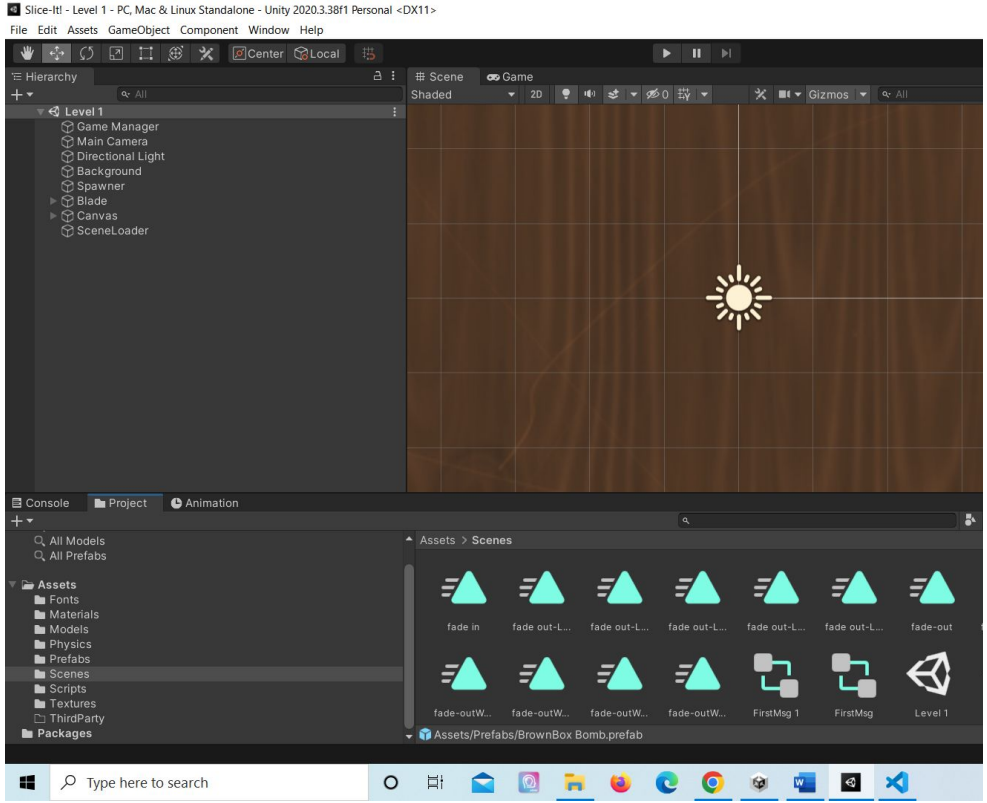
Find the Slice It folder, then click it into you reach Slice-It! and then hit open



Let’s make levels

Step 5:

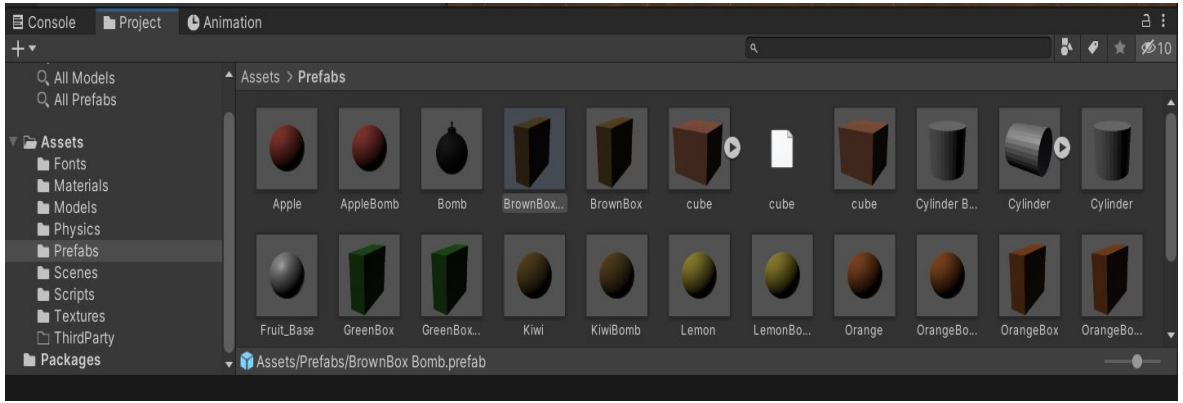
Check all your scripts, prefabs, and there should be one scene named Level 1



Let’s make levels

Step 6:

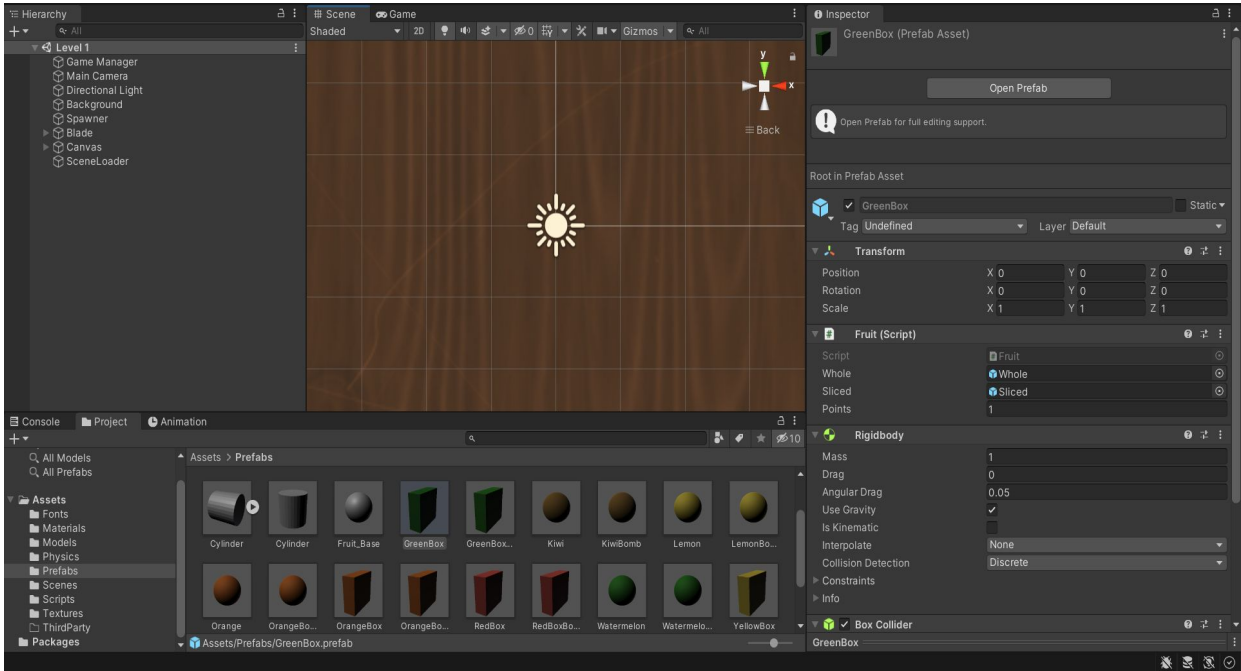
Goes to the prefabs folder



Let’s make level

Step 7:

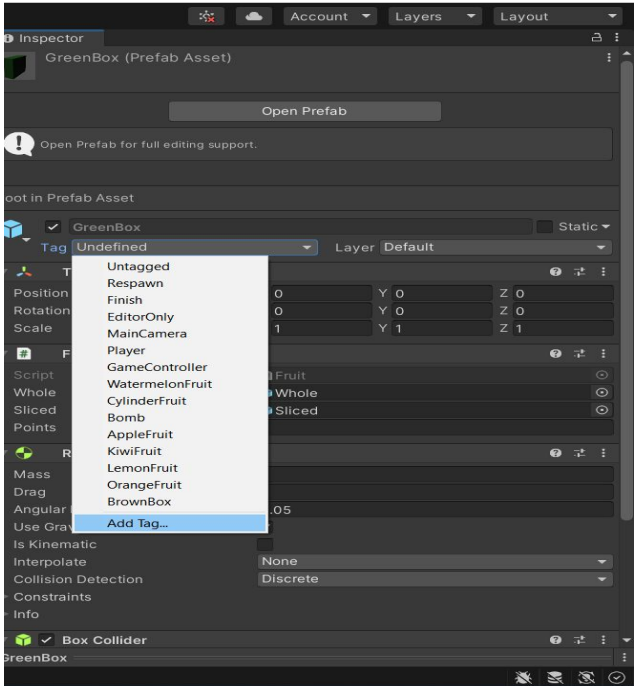
Clicked on the “GreenBox” prefab



Let’s make level

Step 8:

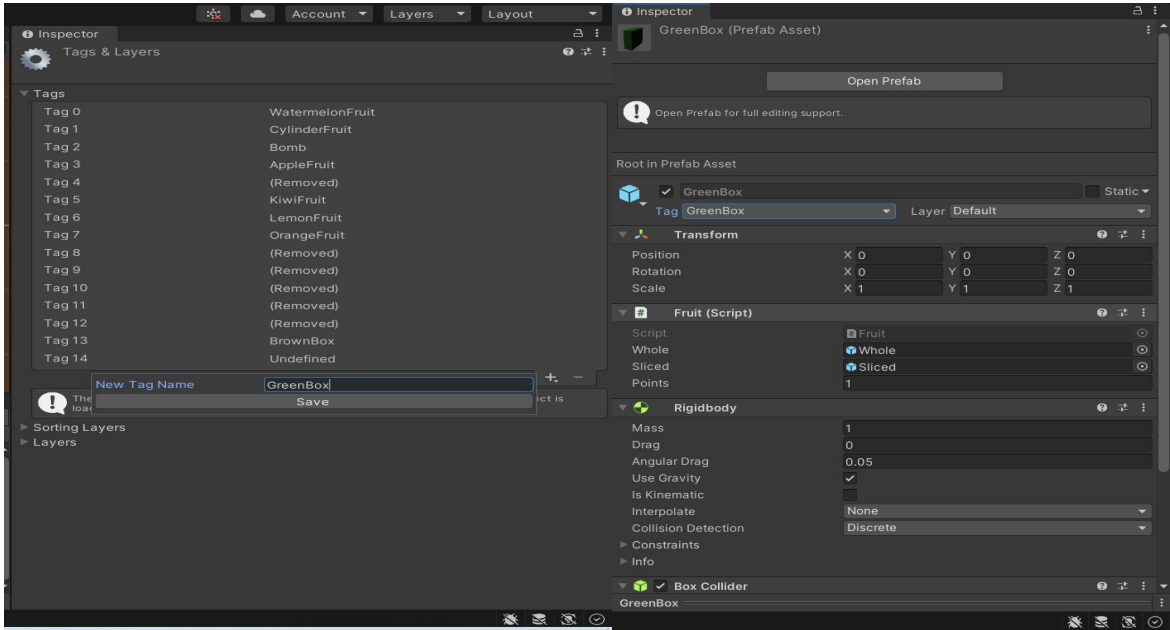
An the tag icon, clicked the add tag sign.



Let’s make levels

Step 9:

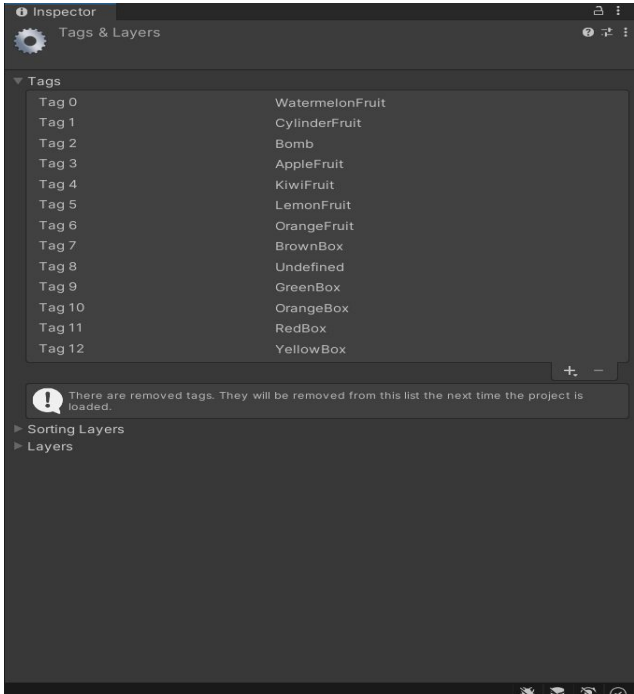
Name this new tag “GreenBox” and go back the it’s prefab and put the tag in the asset.



Let’s make levels

Step 10:

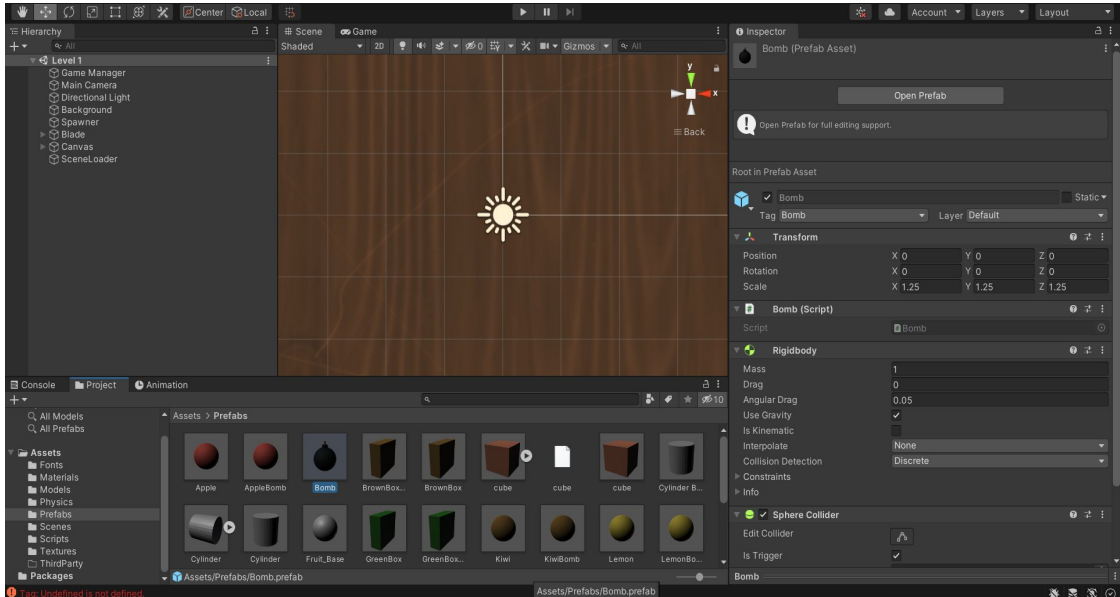
Now repeat this process to all the other prefabs, except the bombs. For reference it’s helpful to name the tag after the color and shape of the prefab



Let’s make levels

Step 11:

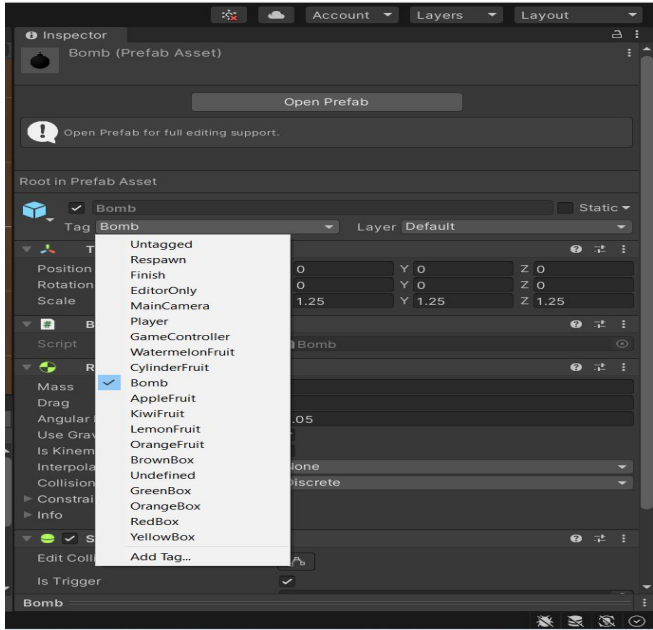
Now click on the bomb prefab



Let’s make levels

Step 12:

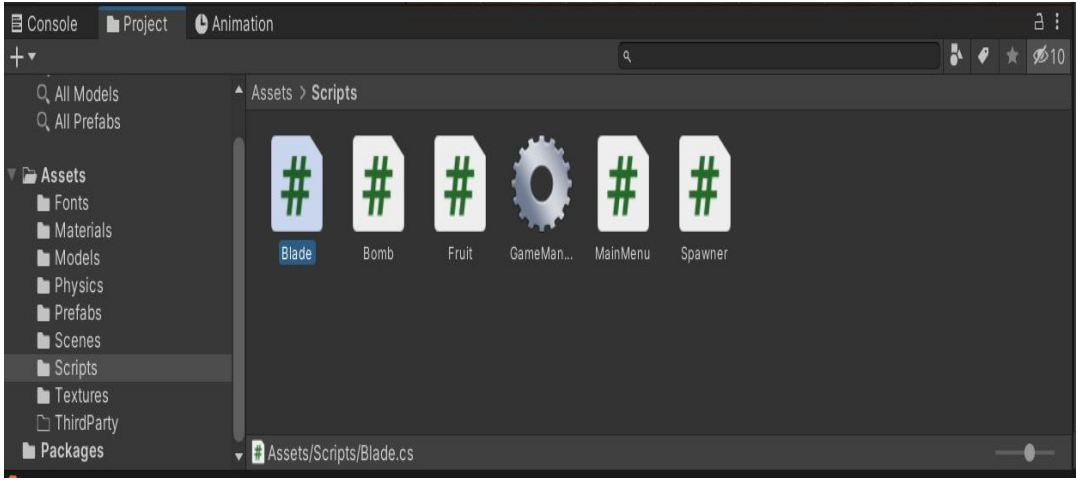
As same as before clicked the tag icon and the plus sign and type “Bomb” as the new tag.



Let’s make levels

Step 13:

Go to the scripts folder, and clicked the blade script.



Let’s make levels

Step 14:

Go to line 19 it’s where we first start to create some code.

Blade.cs

C: > Users > Joseph > Downloads > Slice-It-Game-main (Workshop copy) > Slice-It-Game-main > Slice-It! > Assets > Scripts > Blade.cs

```
1  using UnityEngine;
2  using UnityEngine.UI;
3  using UnityEngine.SceneManagement;
4
5  public class Blade : MonoBehaviour
6  {
7      public Vector3 direction { get; private set; }
8
9      private Camera mainCamera;
10
11     private Collider sliceCollider;
12     private TrailRenderer sliceTrail;
13
14     public float sliceForce = 5f;
15     public float minSliceVelocity = 0.01f;
16
17     private bool slicing;
18     //Can edit
19
20     //All shapes (fruits)
21
22     private Text WinMsg;
23     private Text LoseMsg;
24
```

Let’s make levels

Step 15:

```
//Can edit
public int GreenBoxCount = 0;
```

Create a “public int” object within the scripts and put a prefabs name with count added. Then put an equal sign with a 0 next to it, finally put a semicolon to make it statement. The photo is an example for reference.

Let’s make levels

Step 16:

Do the same of all other prefabs, and remember to use the same objects and statements to avoid confusion.

```
//Can edit
public int GreenBoxCount = 0;
public int WatermelonCount = 0;
public int BrownBoxCount = 0;
public int KiwiCount = 0;
public int LemonCount = 0;
public int AppleCount = 0;
public int OrangeCount = 0;
public int OrangeBoxCount = 0;
public int RedBoxCount = 0;
public int YellowBoxCount = 0;
public int CylinderCount = 0;
```

Let’s make levels

Step 17:

In the blade script go to line 98. Where we are going to type “private void OnTriggerEnter(Collider other){}”.

```
97   // Can edit
98       private void OnTriggerEnter(Collider other)
99       {
100
101
102      }
```

Let’s make levels

Step 18:

Make space between the method brackets. Then take one of your tags inside this statement. Which is “if (other.gameObject.tag == “GreenBox”){}.

```
if (other.gameObject.tag == "GreenBox"){}
```

Let’s make levels

Step 19:

Inside of the if statement bracket put in the count object in the beginning of code by an add increment of one. Example: shown in the photo.

```
if (other.gameObject.tag == "GreenBox"){
    GreenBoxCount++;
}
```

Let’s make level

Step 20:

Do the same to all other prefab counts, but remember your tags and special prefabs to would fit in. Save all the progress you’ve made so far.

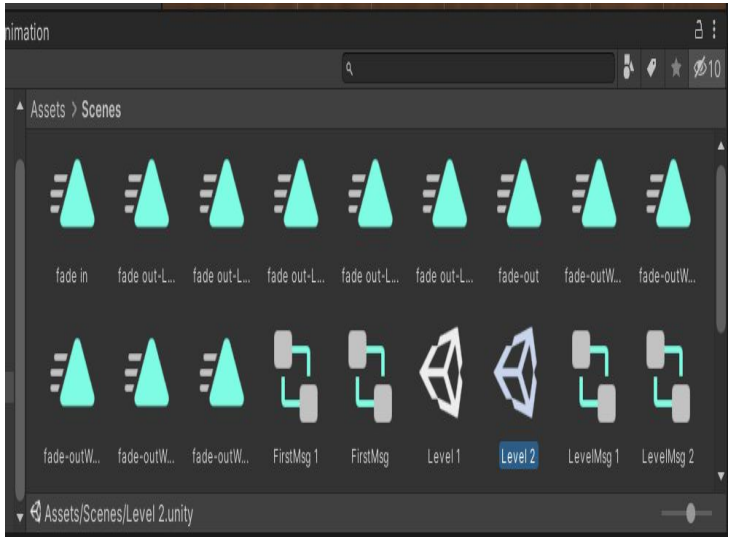
```
if (other.gameObject.tag == "GreenBox"){
    GreenBoxCount++;
}
if(other.gameObject.tag == "WatermelonFruit"){
    WatermelonCount++;
}
if(other.gameObject.tag == "AppleFruit"){
    AppleCount++;
}
if(other.gameObject.tag == "CylinderFruit"){
    CylinderCount++;
}
if(other.gameObject.tag == "BrownBoxFruit"){
    BrownBoxCount++;
}

if(other.gameObject.tag == "KiwiFruit"){
    KiwiCount++;
}
if(other.gameObject.tag == "LemonFruit"){
    LemonCount++;
}
if(other.gameObject.tag == "OrangeFruit"){
    OrangeCount++;
}
if(other.gameObject.tag == "OrangeBoxFruit"){
    OrangeBoxCount++;
}
if(other.gameObject.tag == "RedBoxFruit"){
    RedBoxCount++;
}
if(other.gameObject.tag == "YellowBoxFruit"){
    YellowBoxCount++;
}
```

Let’s make levels

Step 21:

Go back to the game main and go to scenes. Then click on level 1 and press CTRL + D to duplicate the scene. That’s the easy way to make multiple scene with the same format in the Level 1 scene.



Let’s make levels

Step 22:

Go back to the blade script and where going to make the condition to transfer different level if the player met the required task.

```
//Level One
if(BoxCount + AppleCount + WatermelonCount>24){
    WinMsg.text = "You Win!";

}
```

Let’s make level

Step 23:

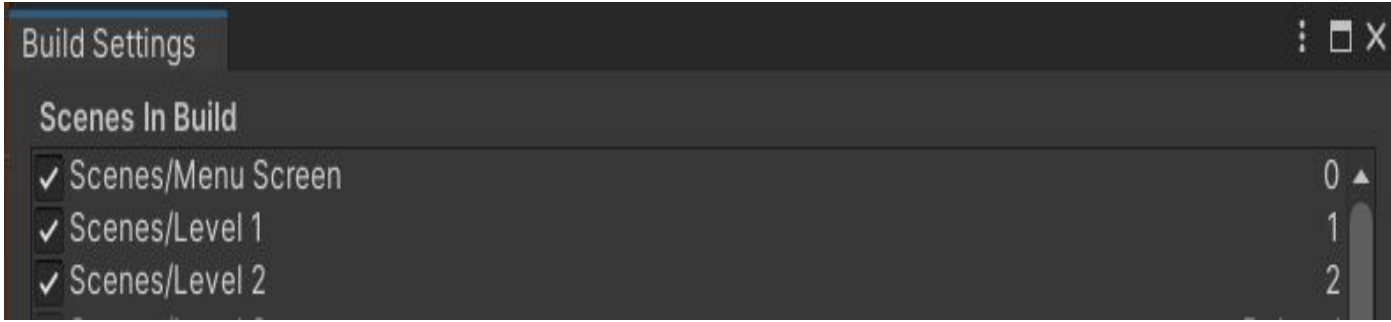
Add UnityEngine.SceneManagement just above the public class method.

```
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;
```

Let's make level

Step 24:

Go back to the unity game menu to click on file, then click on build setting. There you will see the index or order of the scene the game is processing.



Let’s make Levels

Step 25:

Under the box count increment codes we will type in a condition or rather a task to make to advance to the next level. We’re use an example from the photo. “SceneManager.LoadScene(SceneManager.GetActiveScene().build index + 1;” will transfer the games into the next scene which should be level 2.

```
//Level One
if(BoxCount + AppleCount + WatermelonCount>24){
    WinMsg.text = "You Win!";
    SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);
}
```

Let’s make levels

Step 26:

Below the if statement for level one make an else if statement where a player hits a bomb. Which should revert all prefab counts to zero.

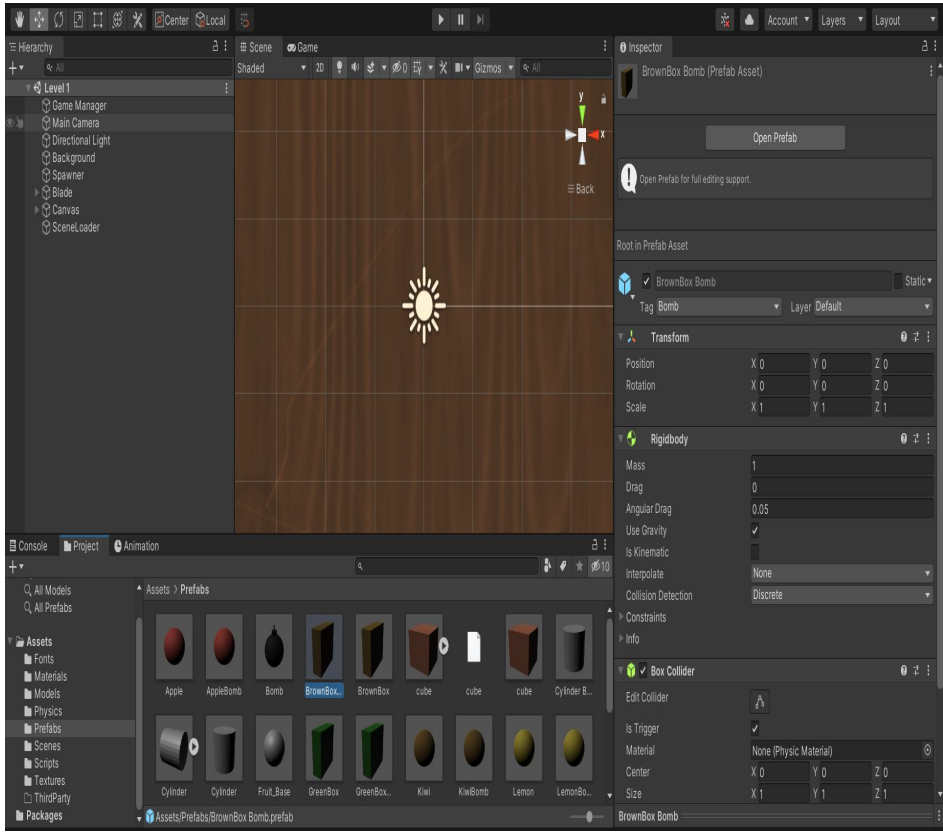
```
else if(other.gameObject.tag == "Bomb"){
    BoxCount = 0;
    WatermelonCount = 0;
    OrangeBoxCount = 0;
    OrangeCount= 0 ;
    BrownBoxCount =0;
    KiwiCount = 0;
    RedBoxCount = 0;
    YellowBoxCount = 0;
    CylinderCount = 0;
    AppleCount = 0;
    LemonCount = 0;

}
```

Let’s make levels

Step 27:

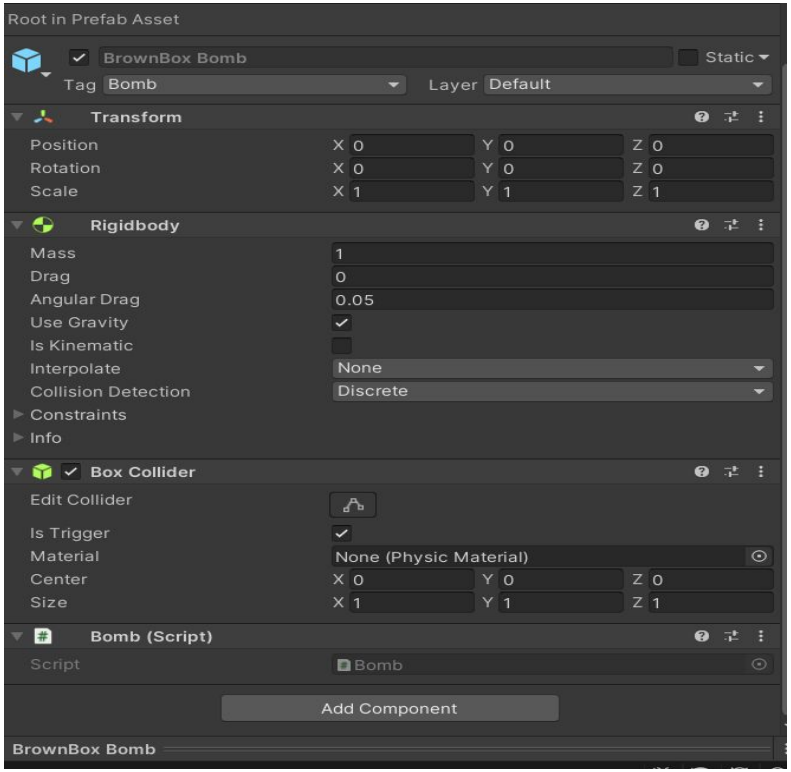
To make fruits as bomb first hit CTRL + D to duplicate them. Then click on the tag icon and put a bomb tag to them.



Let’s make levels

Step 28:

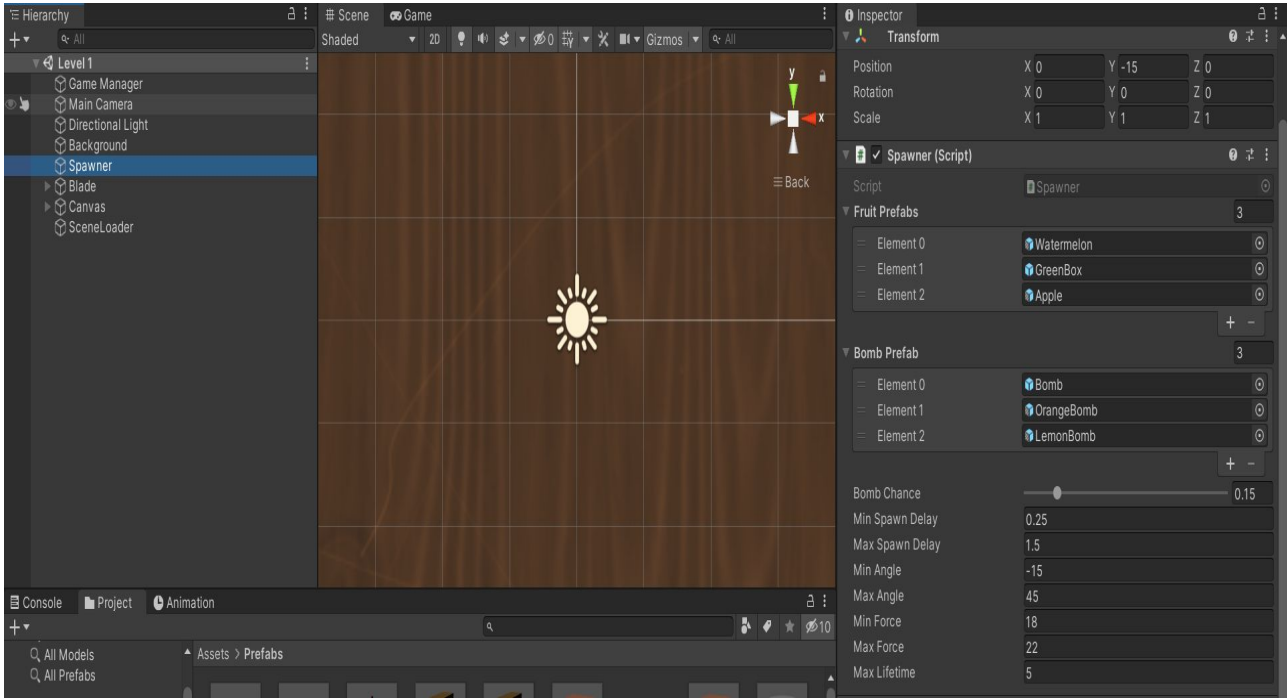
Remove the fruit scripts from them and instead replace it with a bomb scripts which will make it act like a bomb.



Let’s make levels

Step 29:

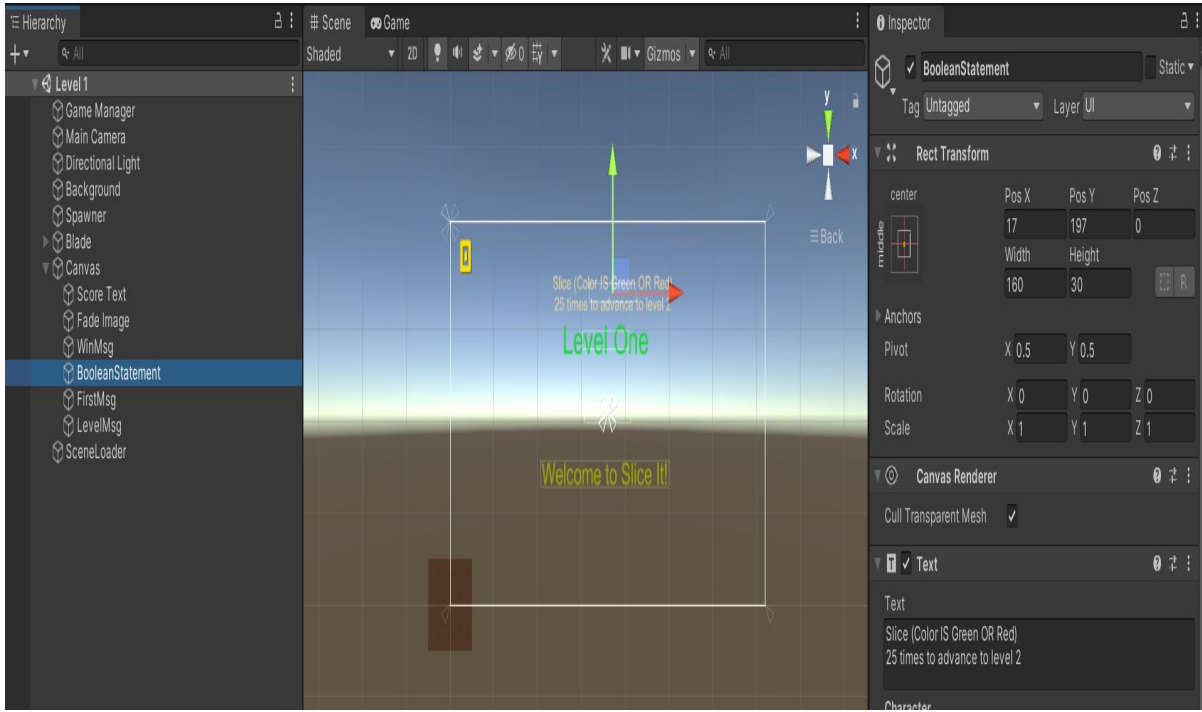
Click on the spawner object and clicked on the prefabs folder. Then drag and dropped the appropriate prefabs object in the fruit array and the bomb array. There you can manuiplate the chances of bombs for difficulty.



Let’s make levels

Step 30:

Click the canvas object and duplicate either the “Firstmsg” or “Levelmsg” to made a boolean statement message. This will help not only you but other player on what to do in the game.



Hurray you’ve done it!

You’ve create a level in Unity.

Since Unity is a complex application there will be times where it can get frustrating, but remember with resilience and dedication anyone can make a good game.