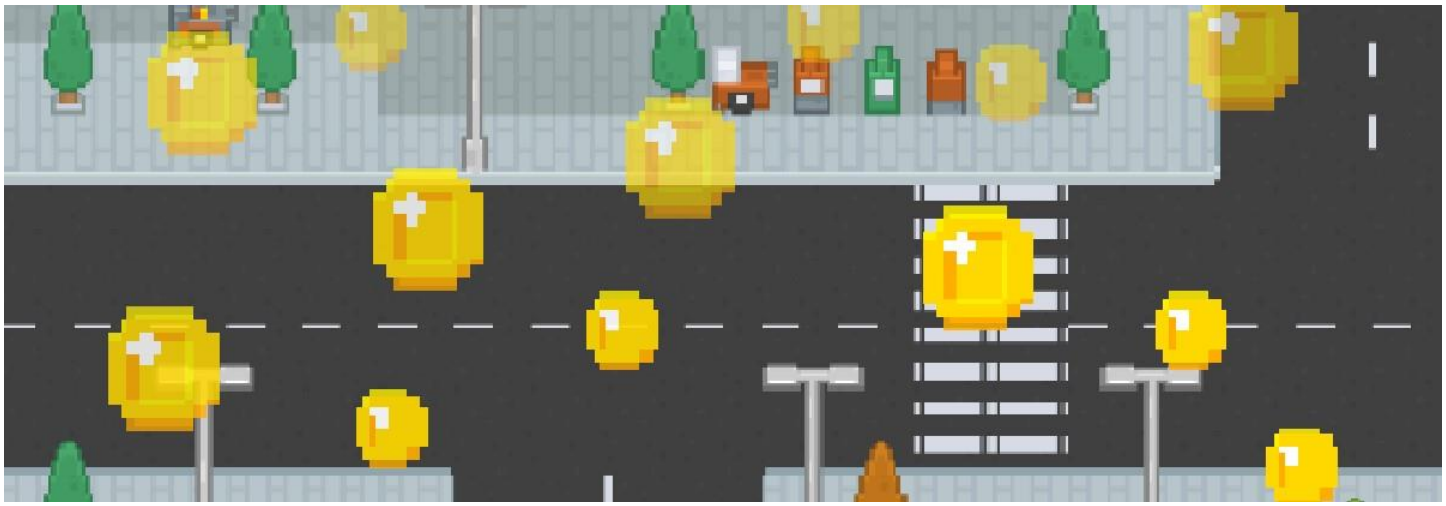
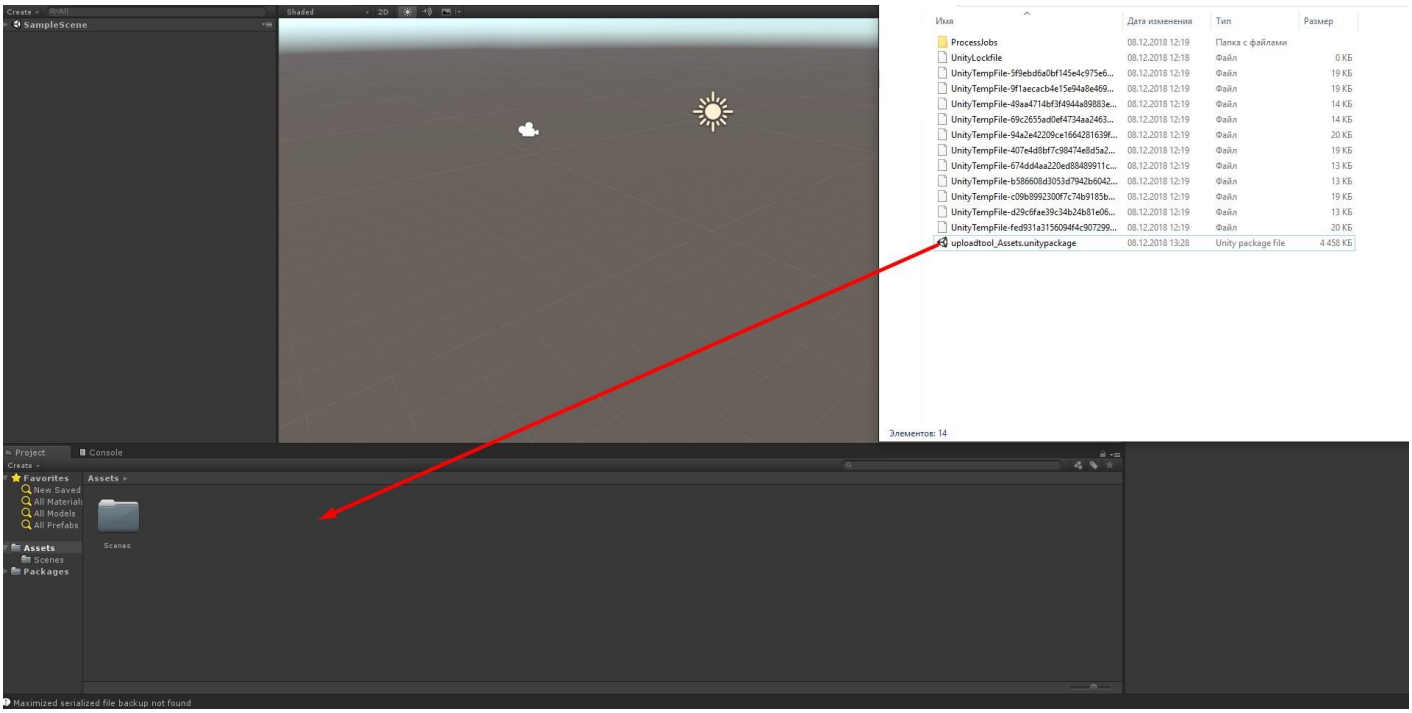


Clicker-Idle Game Template

This Clicker-Idle Game Template was made for the competition.
For the reason of hopelessness, it was decided
to add it to the AssetStore

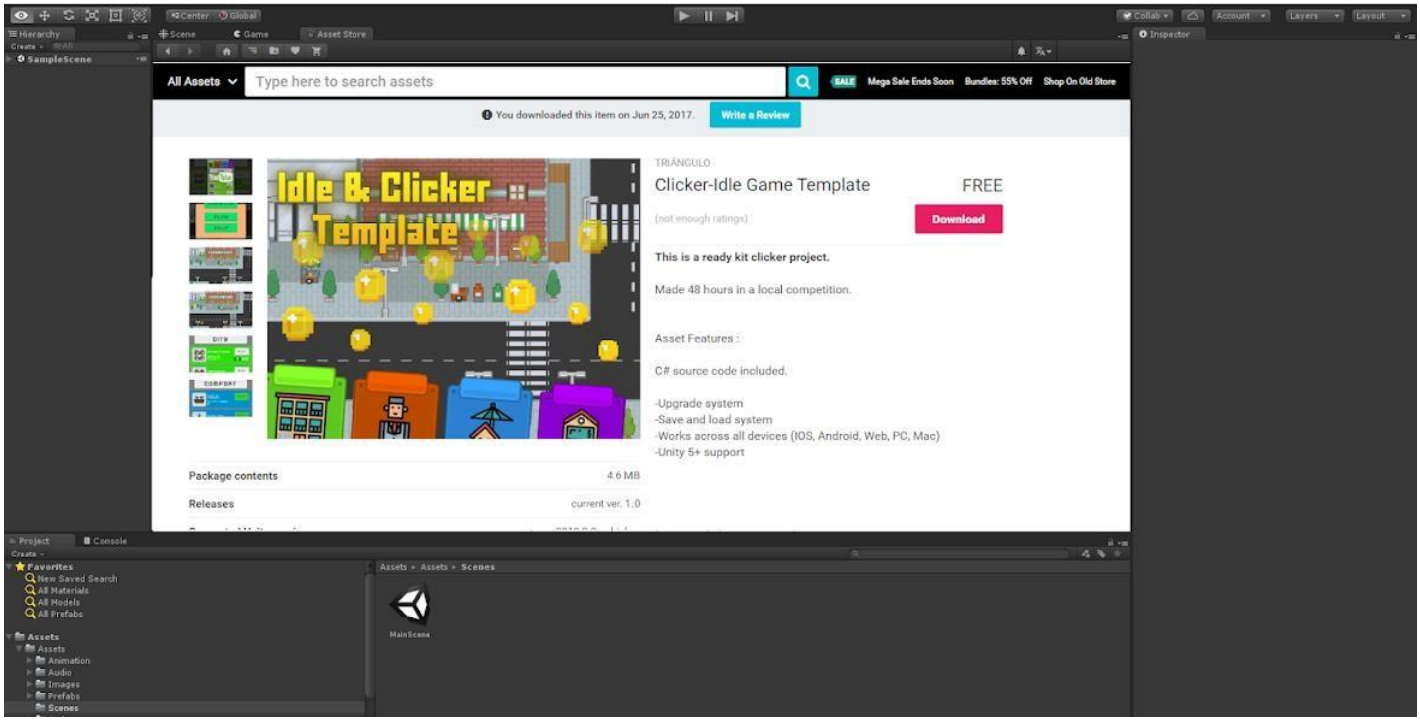
Installation

 Transfer the file to the editor

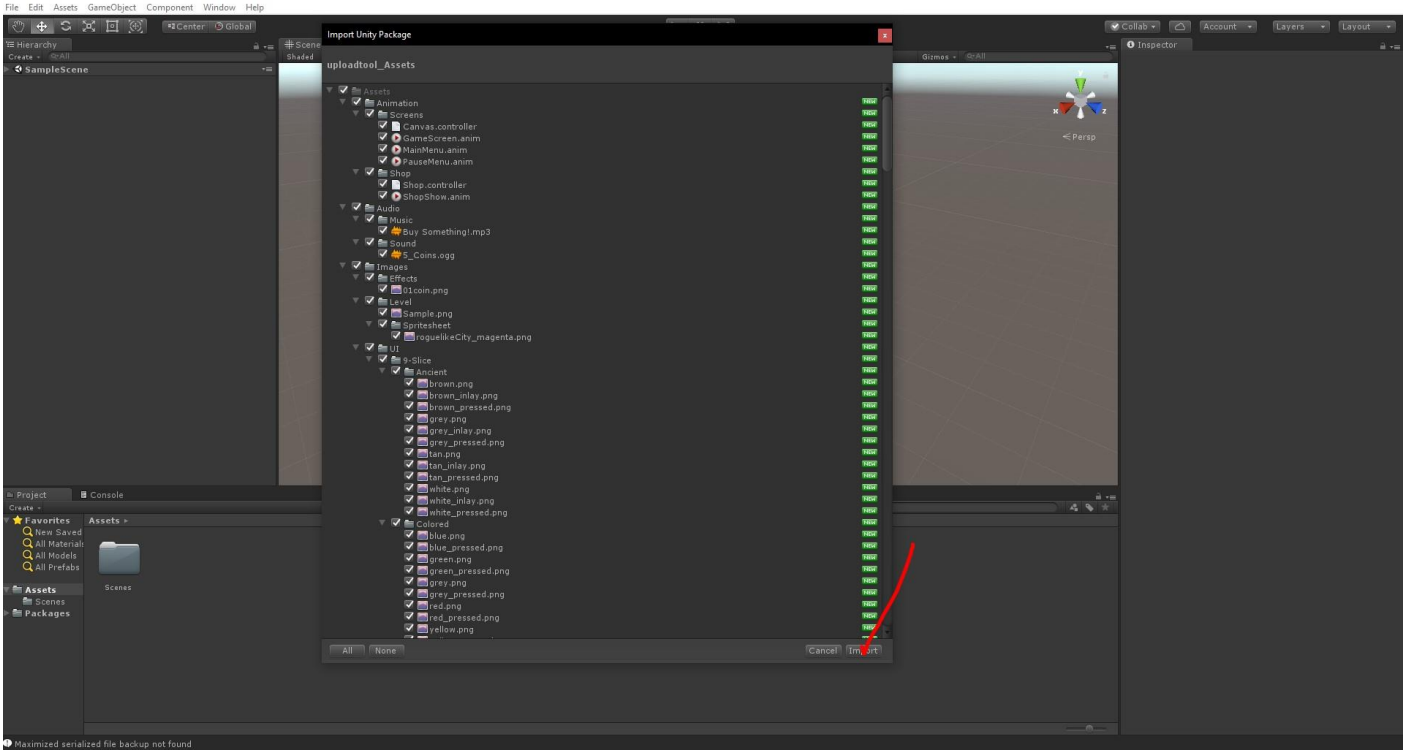


Or

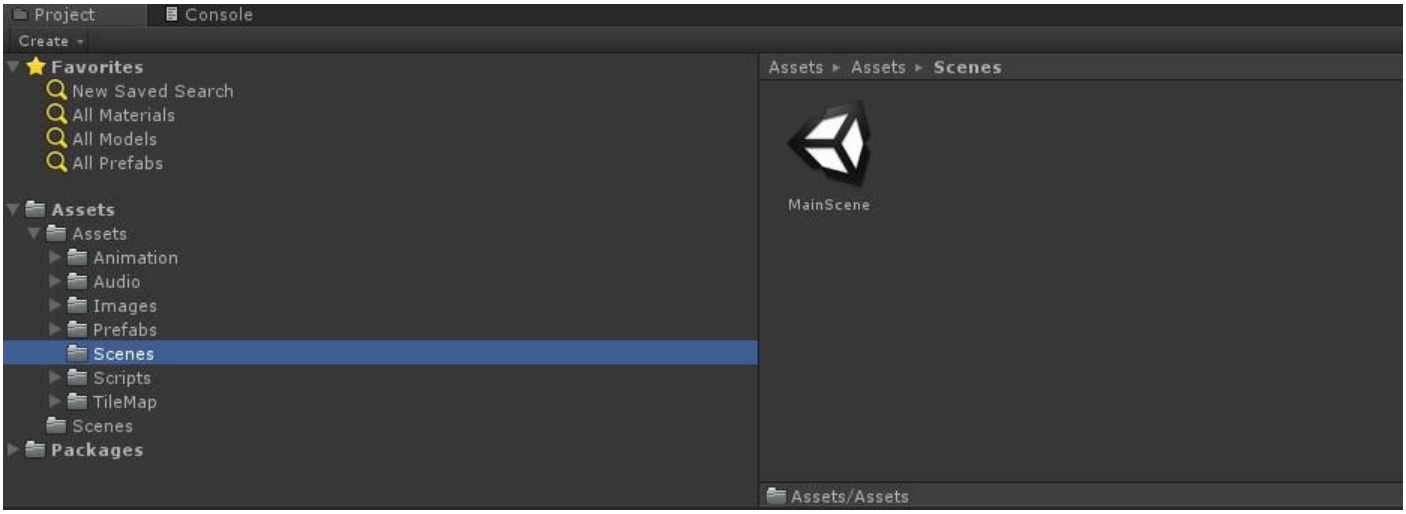
 Go to the product page, download



 Import



 The scene is in Assets / Scenes



Hierarchy

Project:

- Animation - stores the files responsible for the animation .anim, .controller
- Audio - keeps all the music and sounds
- Images - stores all images (UI, Sprites, etc.)
- Prefabs - keeps all copies of game prefabs
- Scenes - keeps all the scenes
- Scripts - stores all scripts responsible for logic
- TileMap - stores all files associated with the Tilemap system

Scene:

- **Main Camera** - main camera on scene
- **[UI]** -here are all the objects responsible for the UI
- **[UI]/Canvas/GameScreen** - the game screen, which contains all the elements responsible for the game logic
- **[UI]/Canvas/PauseMenu**- screen, called when you pause
- **[UI]/Canvas/MainMenu**- screen, called when the game starts
- **[Level]** - here everything is related to the level, TileMap, Background
- **[Managers]** - all managers are stored here.
- **[Managers]/Data** - on the object is the script responsible for storing Data and its loading, saving
- **[Managers]/Singleton**- on the object is the script responsible for storing all managers for further calling from other scripts
- **[Managers]/Manager**- on the object are scripts responsible for all the logic
- **[Managers]/EventSystem**- default object for UI to work correctly (Do not delete)
- **[Audio]** - stores sound objects
-

Scripts

Editing scripts is not difficult if you are familiar with c #

```
namespace Idle
{
```

- Idle namespace is used for all scripts to avoid conflicts with your scripts.

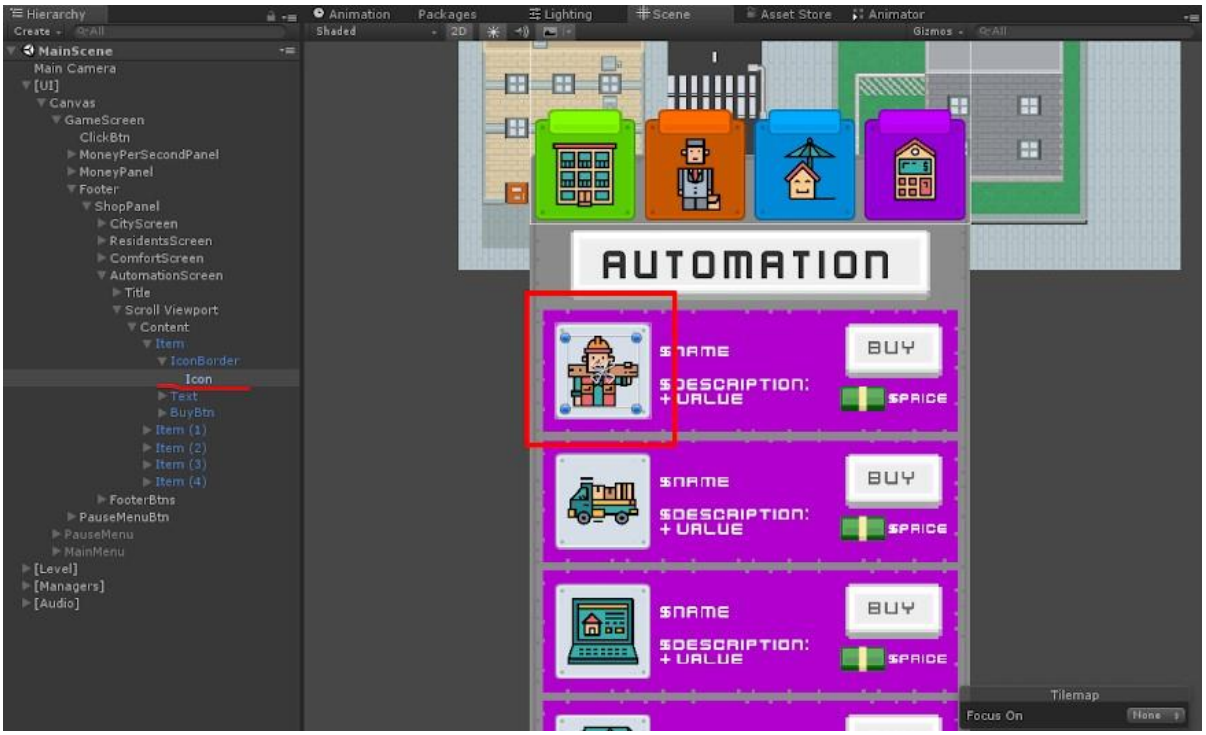
```
//Output File Name
public string fileName;

//Method to save, accepts any types Save(int[]), Save(string[]), Save(SampleClass[])
public void Save(object[] objects)
{
    //Create a local instance of a binary formatter
    BinaryFormatter binaryFormatter = new BinaryFormatter();

    //Open or create file
    using (FileStream fileStream = File.Open(Application.persistentDataPath + "/" + fileName + ".bin", FileMode.OpenOrCreate))
    {
        binaryFormatter.Serialize(fileStream, objects); //We write our Objects to a file.
        fileStream.Close();  //Close the file
    }
}
```

- All code has comments describing each action, so editing is easy.

Images

Images are pretty easy to edit.

 Select any image on scene



1. In the inspector we see the component Image
2. In the Source Image field we transfer any sprite from project files.

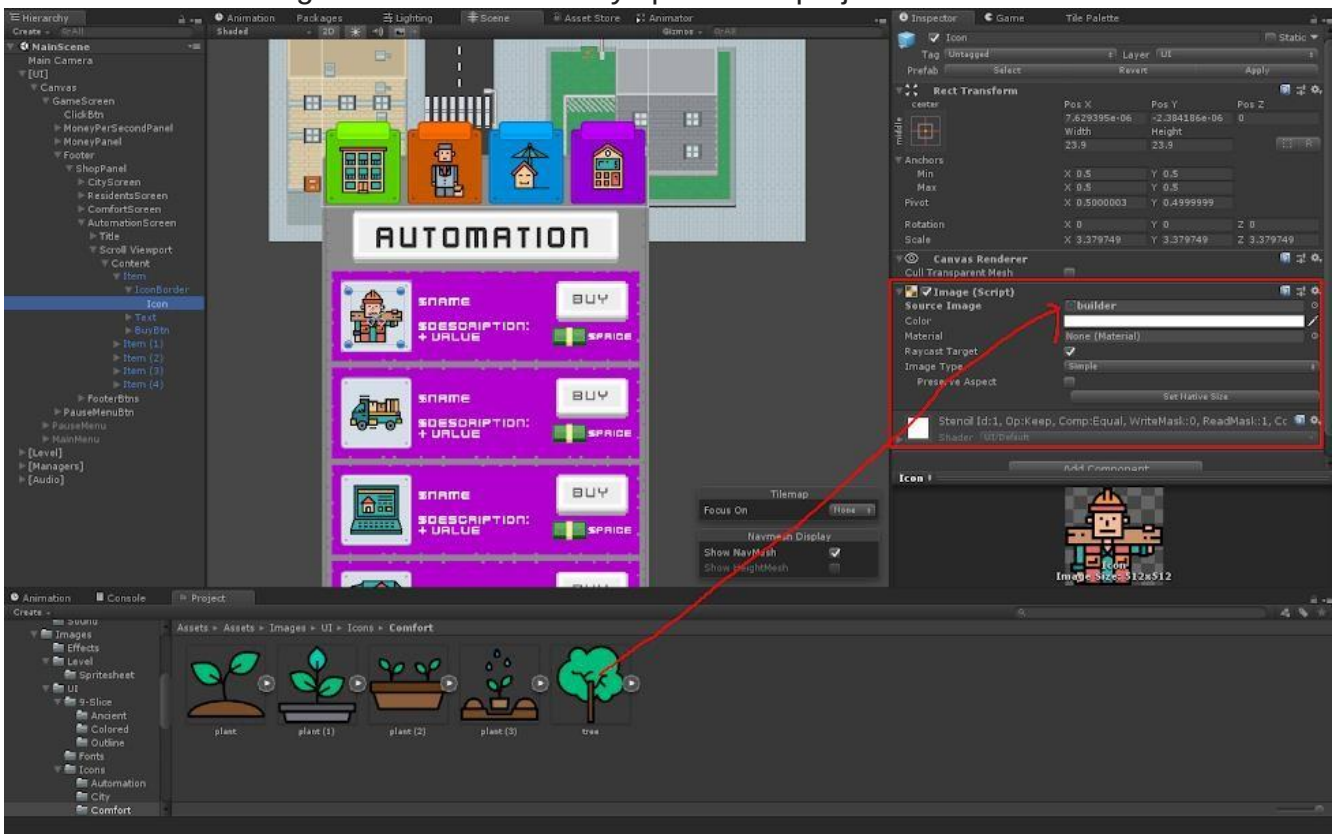
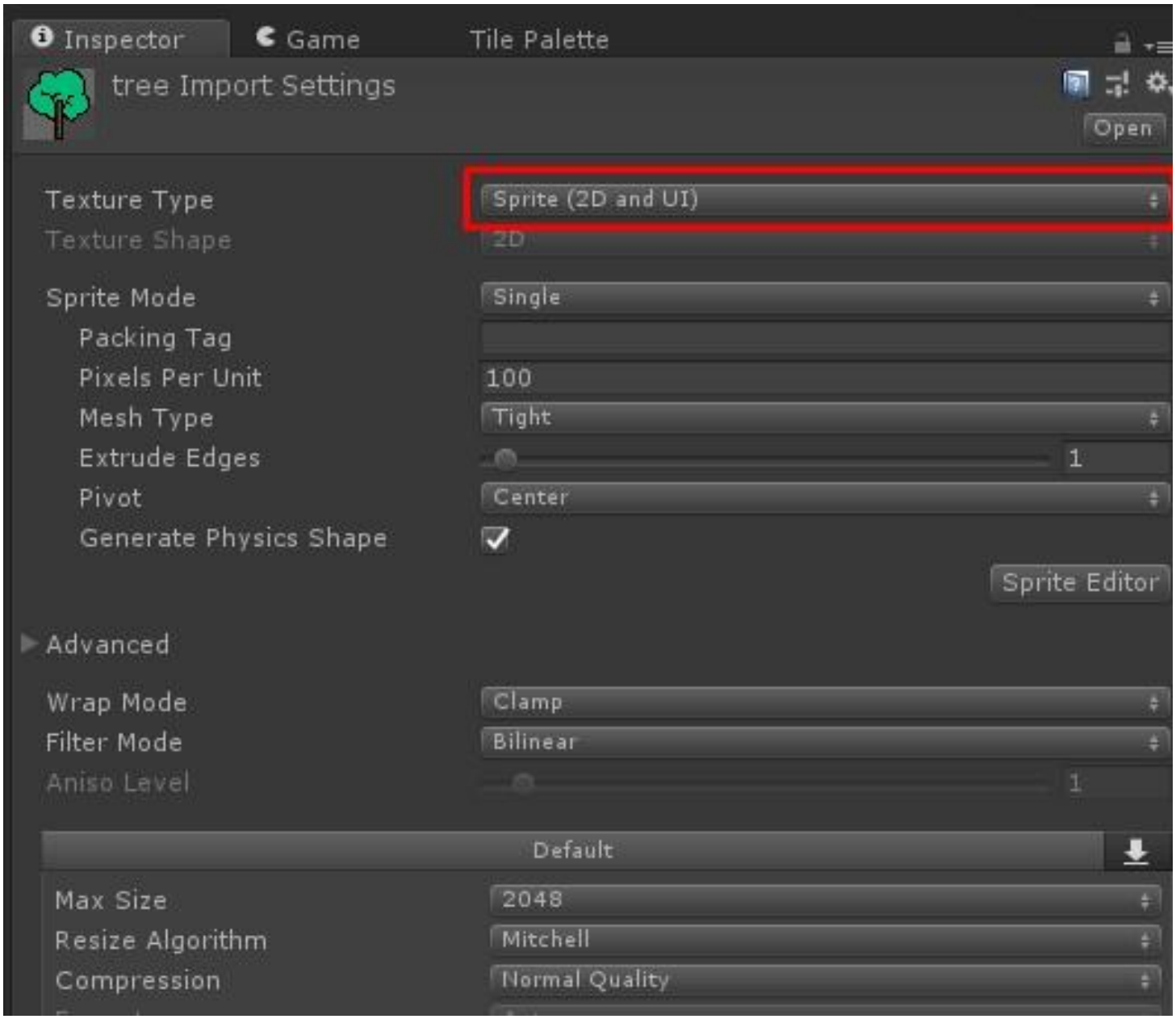
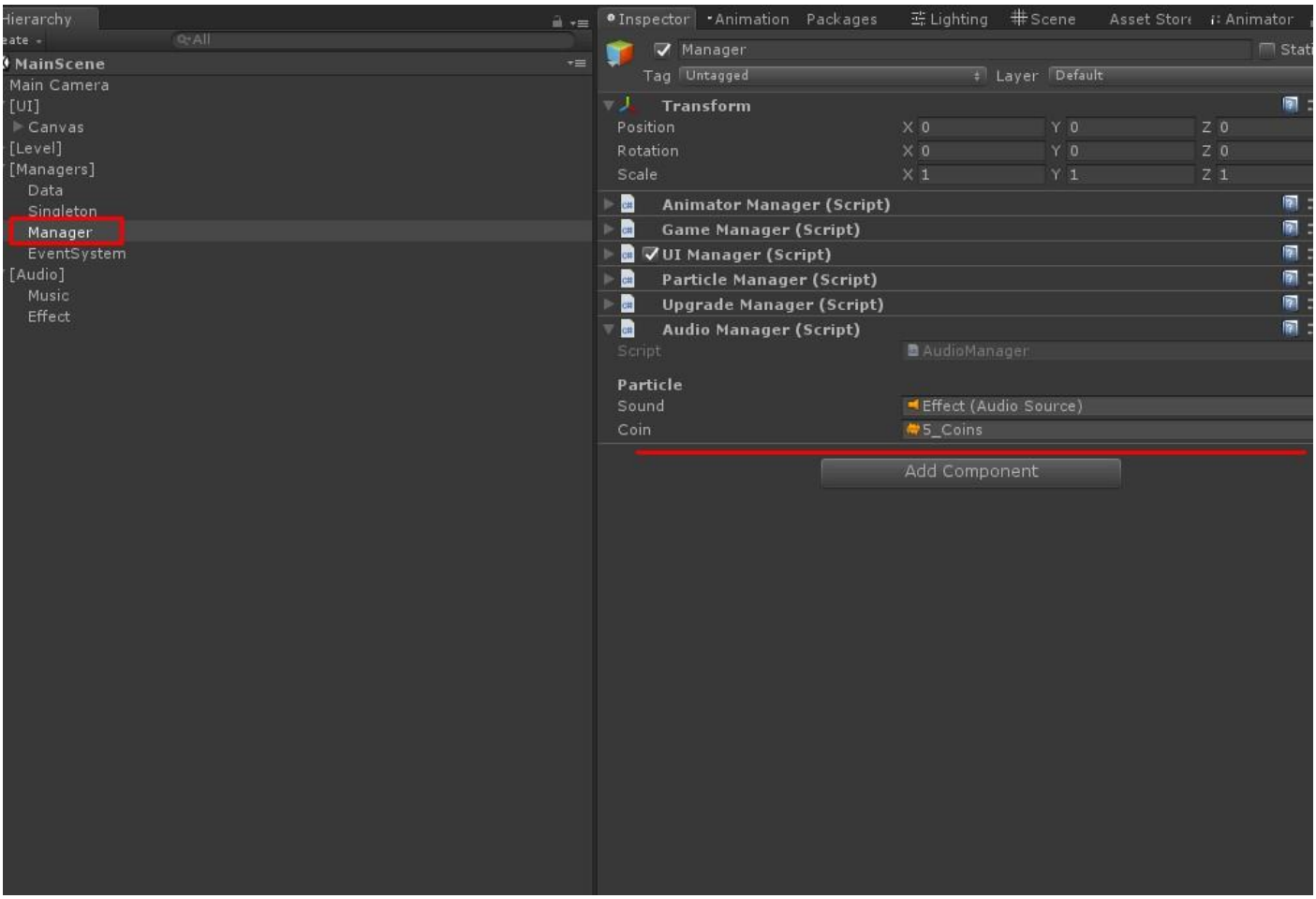


Image Settings:



Audio

In case you want to replace sounds:



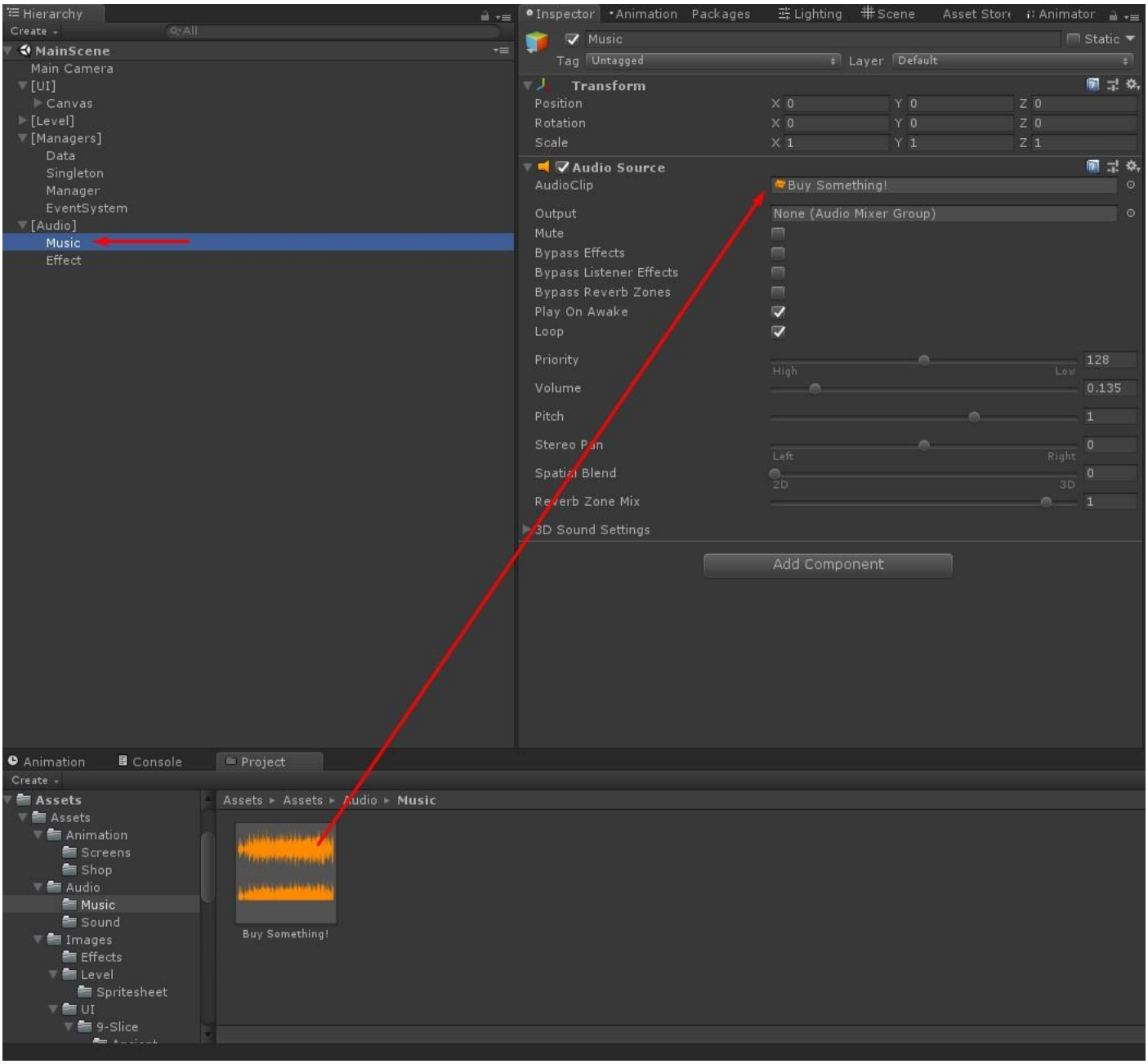
If you want to add a new sound effect:

1. Open the AudioManager script
2. add a new field - public AudioClip NAME
3. and call from any script -

   Managers.Instance.audioManager.PlaySound(Managers.Instance.audioManager.NAME);

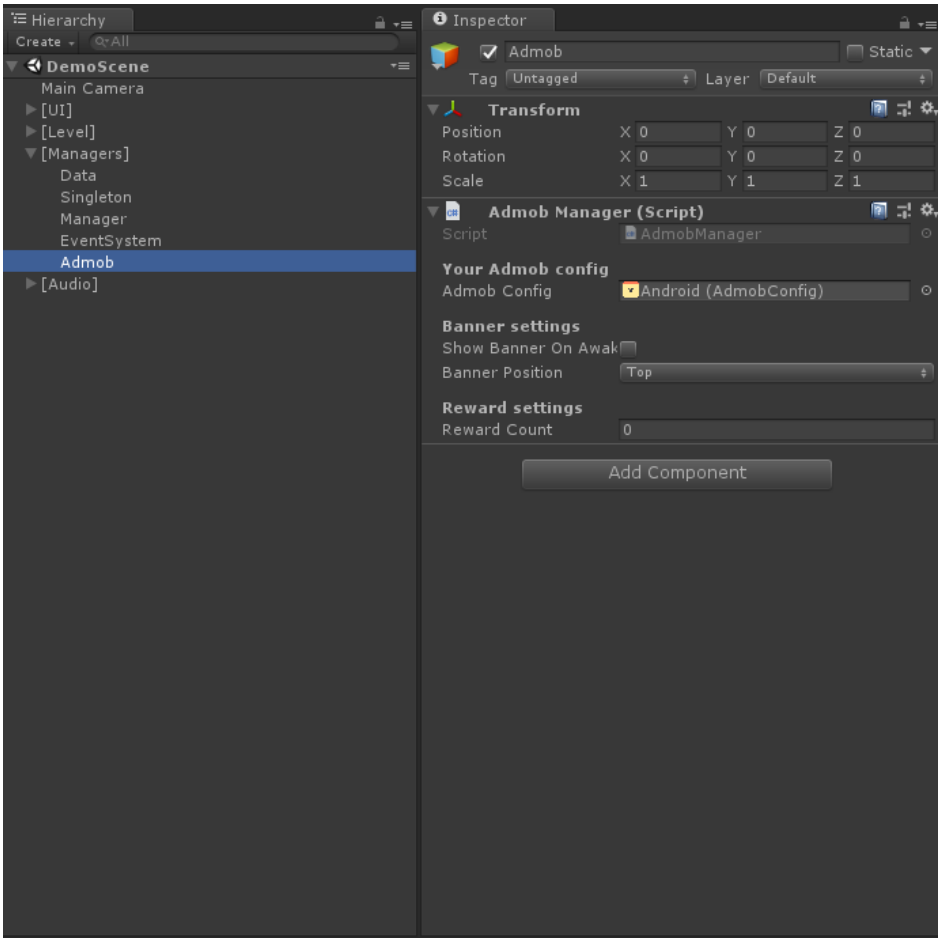
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class AudioManager : MonoBehaviour
{

    [Header("Audio")]
    public AudioSource Sound;

    [Header("Audio Effects")]
    public AudioClip Coin;

    // Method of playing effect, accepts any effect from cached
    public void PlaySound(AudioClip sound)
    {
        Sound.clip = sound;
        Sound.Play();
    }
}

```

If you want to replace music:



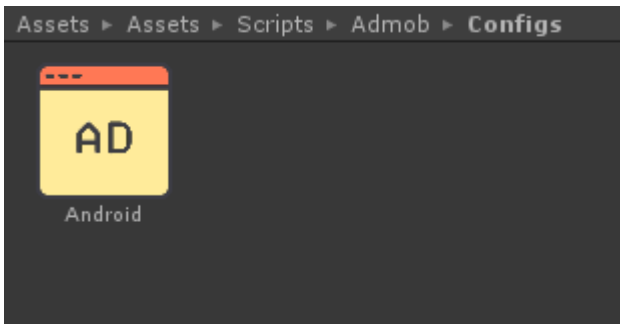
Admob

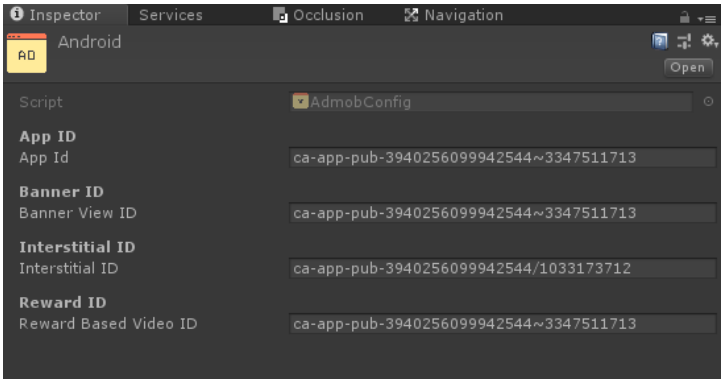
Customization



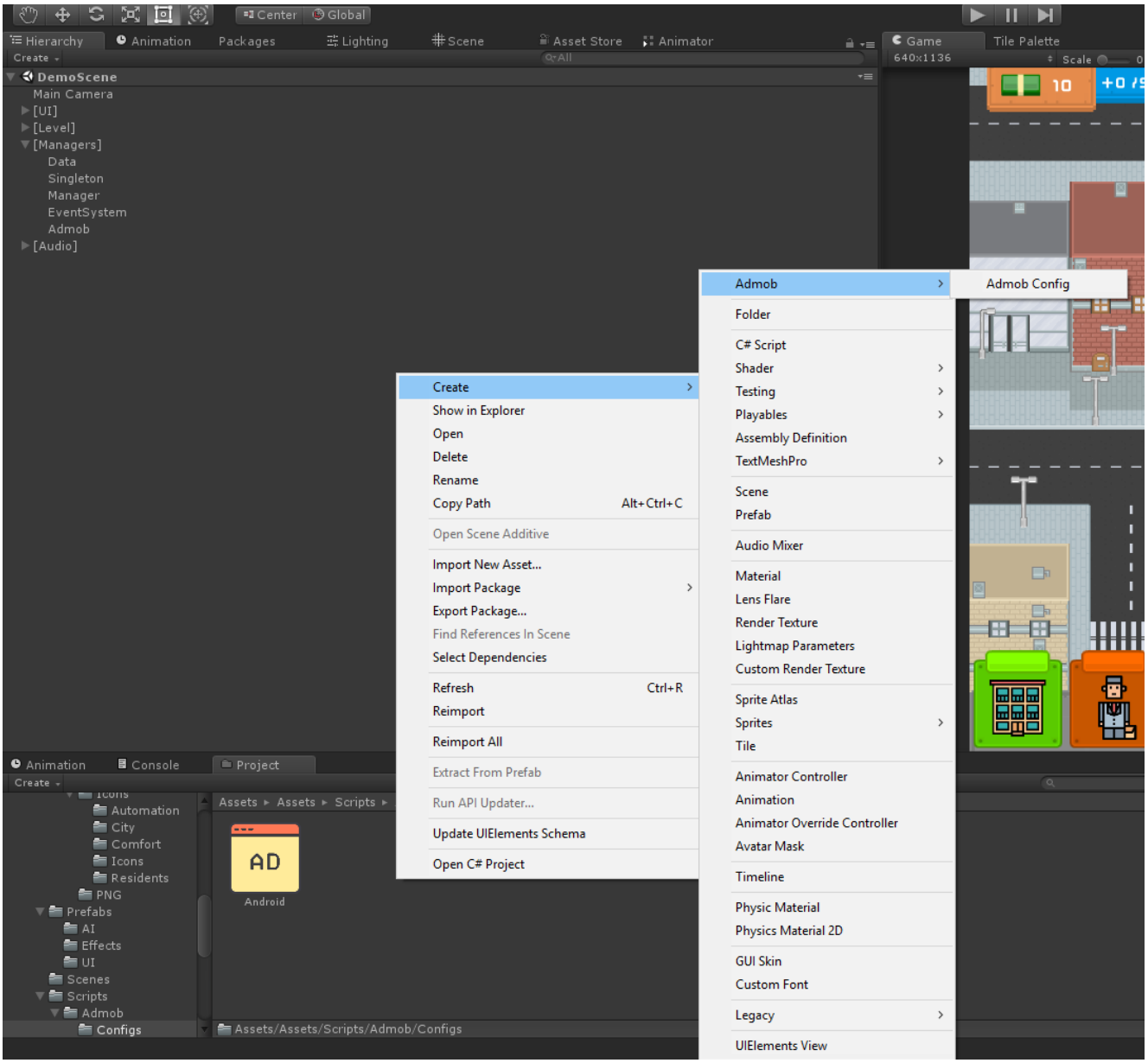
Elements:

1. Admob config - your config with ID settings





- Creating your config:



Admob code:

- **All code associated with Admob is stored in AdmobManager.cs**

An example of using code in your scripts is described in GameManager.cs:

```
//Mthhod call ads from AdmobManager
public void ShowRewardAd()
{
    //AdmobManager.instance.ShowBanner(); //An example of calling a banner from a script
    //AdmobManager.instance.HideBanner(); //Hide banner


    //AdmobManager.instance.ShowInterstitial(); An example of calling a Interstitial from a script

    AdmobManager.instance.rewardCount = 10; //Set reward value
    AdmobManager.instance.ShowReward(); //Call the video
}
```