The Dungeon

Thank you so much for purchasing asset

Documentation

Asset

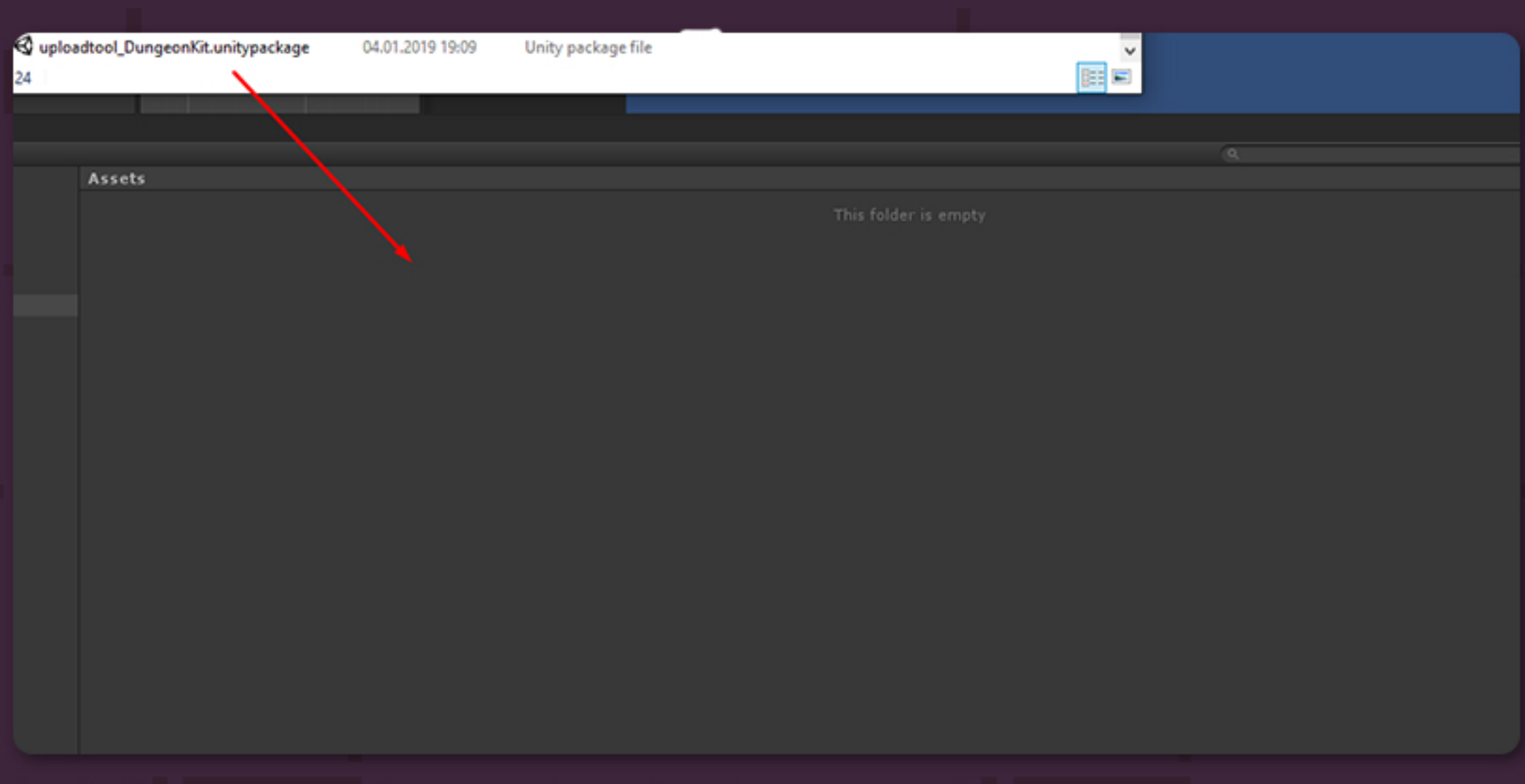
Info

This is an asset of the template dungeon game. In this asset:
1.Top down controller
2.Combat system
3.AI(simple, but easy to edit)
4.Dialog system
5.Shop
6.Level logic and more

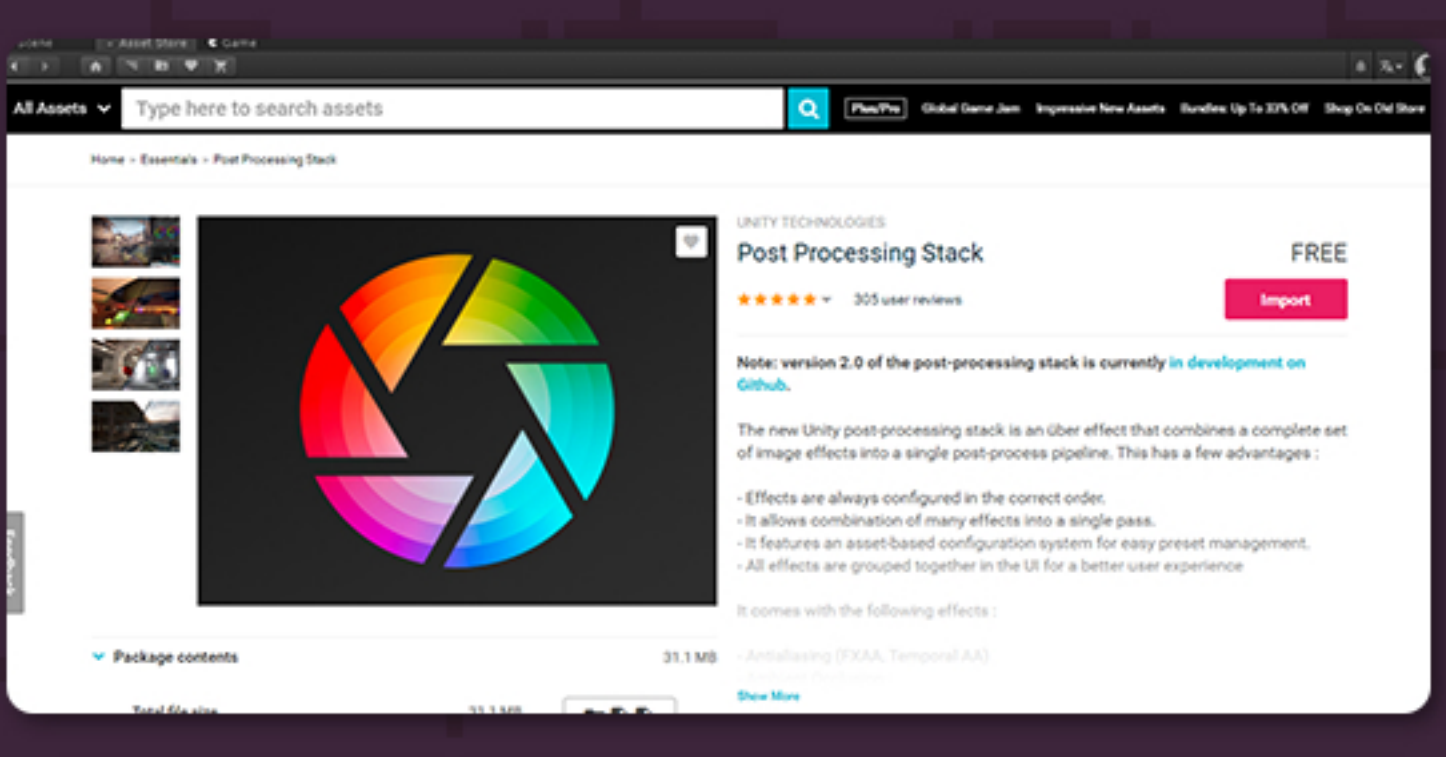
Asset

Installation

Transfer the file to the editor



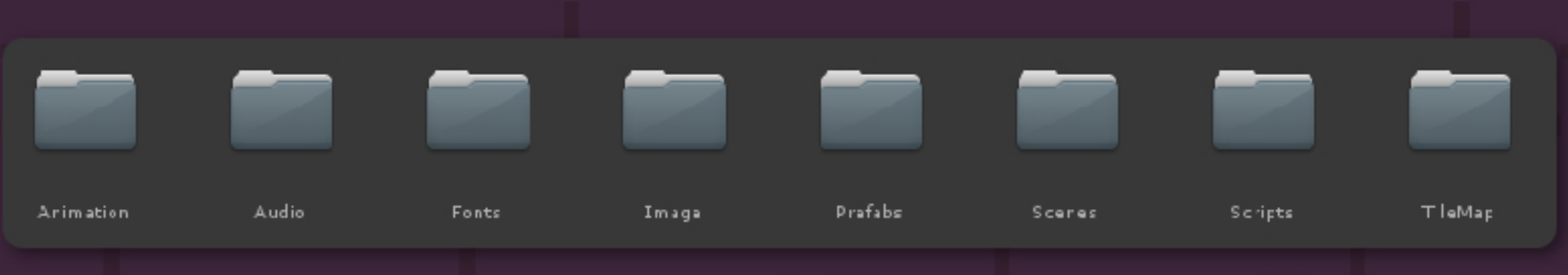
Go to the product page, download



Asset

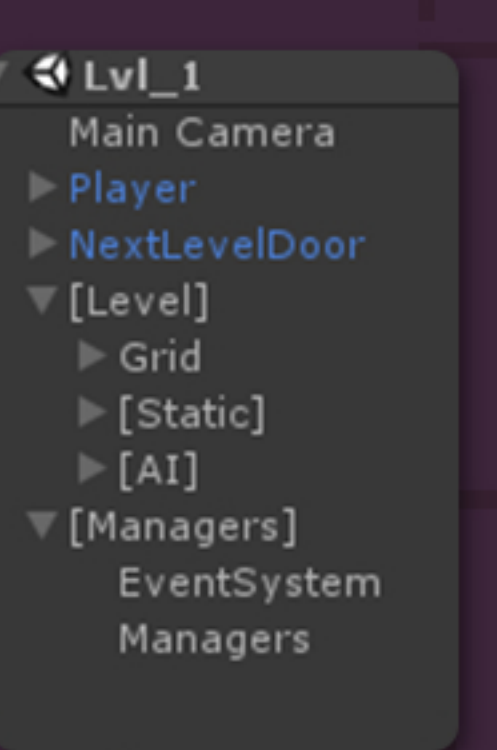
Hierarchy

Project:



- Animation – stores the files responsible for the animation .anim, .controller
- Audio – keeps all the music and sounds
- Fonts – keeps all the fonts
- 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
- Player – Player GameObject
- Next level door – Next level door GameObject
- [Level] – Keeps most of the level
- [Level]/Grid – Level Tilemap
- [Level]/Static – Static objects such as boxes, etc
- [Level]/AI – Keeps AI GameObjects
- [Managers] – Keeps all managers
- [Managers]/EventSystem- default object for UI to work correctly (Do not delete)
- [Managers]/Managers – on the object are scripts responsible for all the logic

Asset

Scripts

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

```
namespace DungeonKIT
{
```

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

```
//Play animation in animator PlayAnimation("Animation name")
public void PlayAnimation(string animationName)
{
    animator.Play(animationName);
}

//Play clip in playable director PlayPlayableDirector(timelineAssets[id], DirectorWrapMode. )
public void PlayPlayableDirector(TimelineAsset timelineAsset, DirectorWrapMode directorWrap)
{
    playableDirector.extrapolationMode = directorWrap; //Set director wrap mode in playableDirector
    playableDirector.playableAsset = timelineAsset; // Set clip
    playableDirector.Play(); //Play
}
```

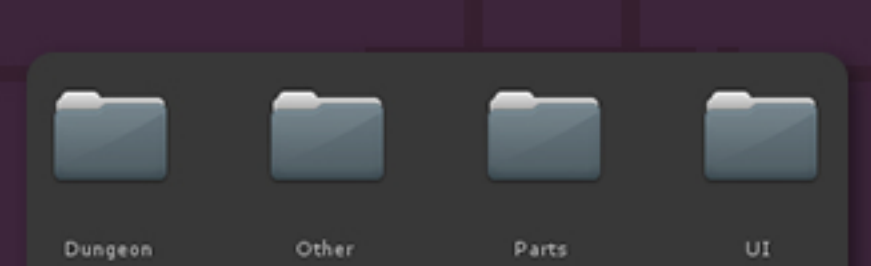
- All code has comments describing each action, so editing is easy

Asset

Images

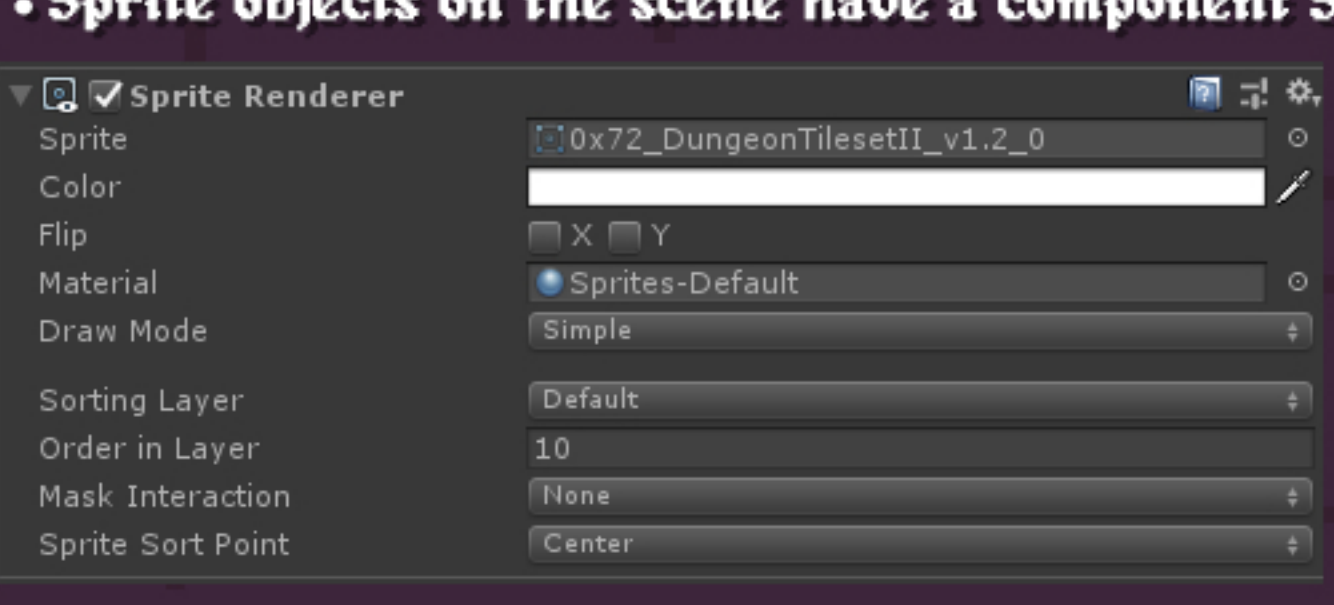
Images are pretty easy to edit.

- If you want to redraw the level, you can simply edit the file Dungeon_Tileset, save with the replacement and the tiles will automatically change

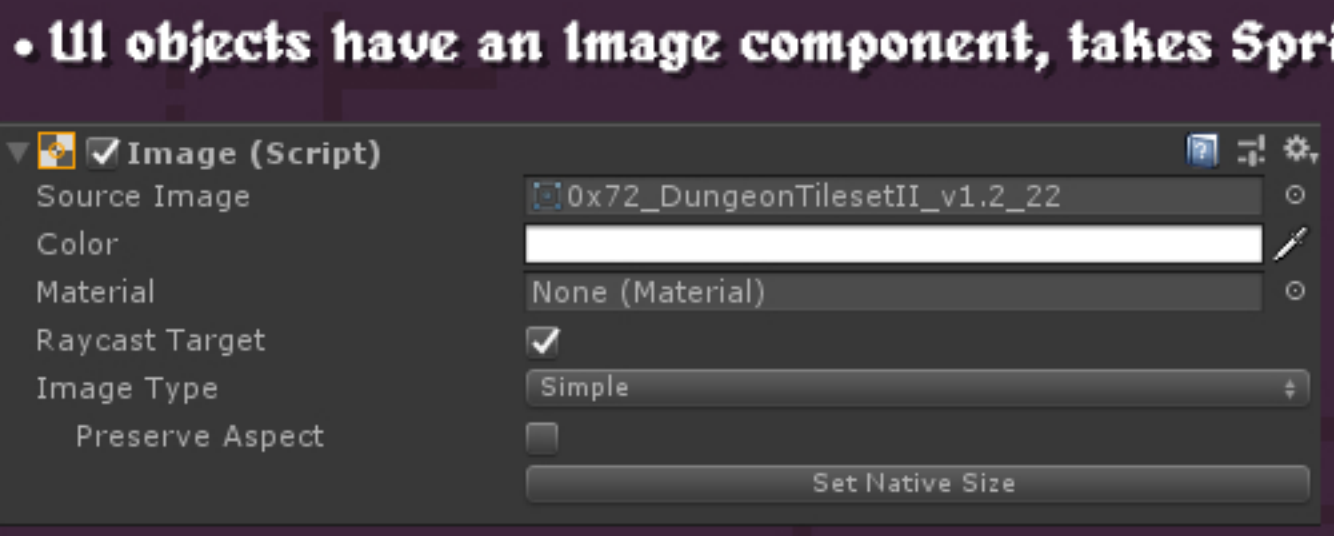


- Everything else you will find in the folder Image

- Sprite objects on the scene have a component Sprite Renderer, takes Sprite



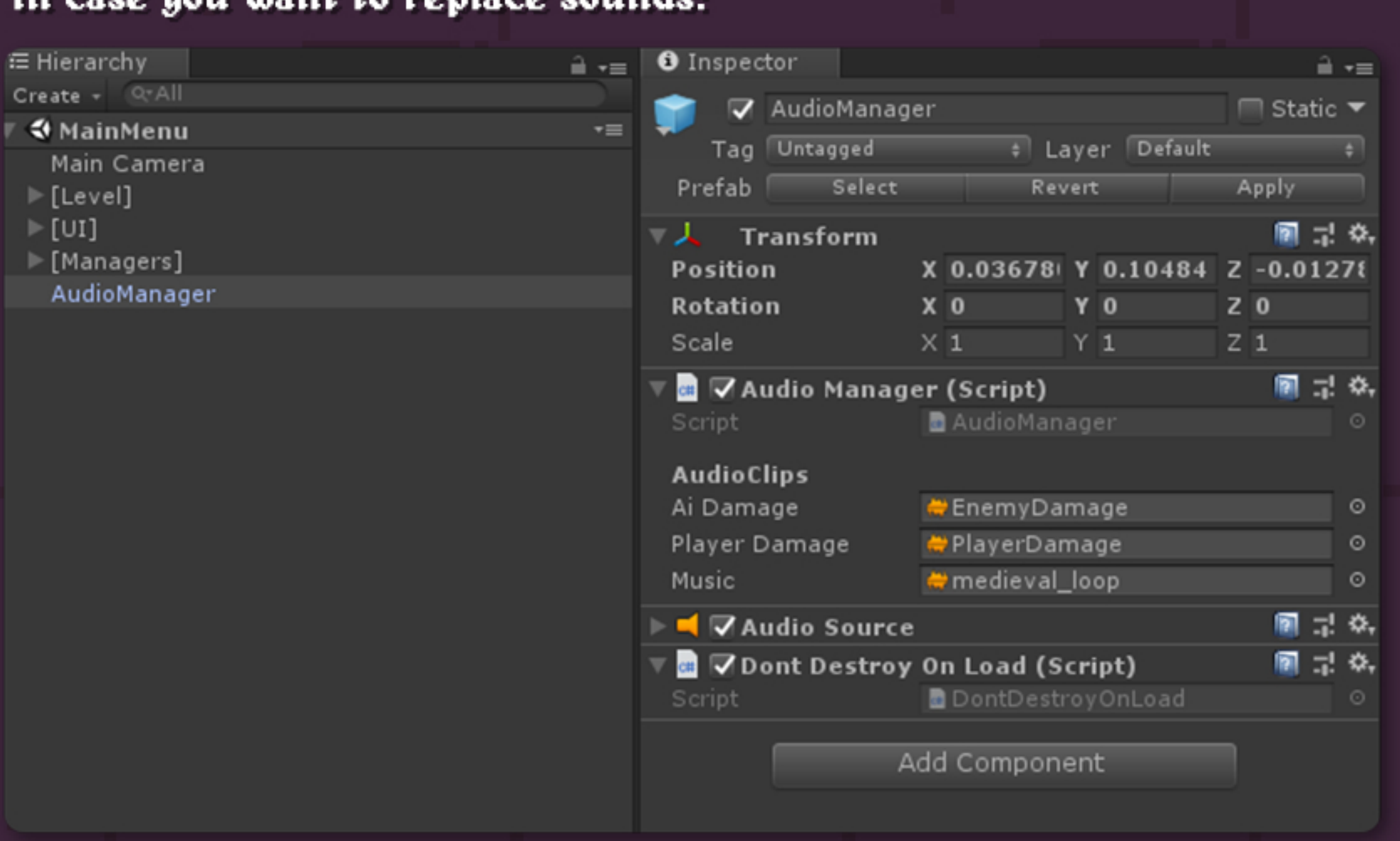
- UI objects have an Image component, takes Sprite



Asset

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

```
[Header("AudioClips")] //List of sounds
public AudioClip aiDamage;
public AudioClip playerDamage;

public AudioClip music;
```

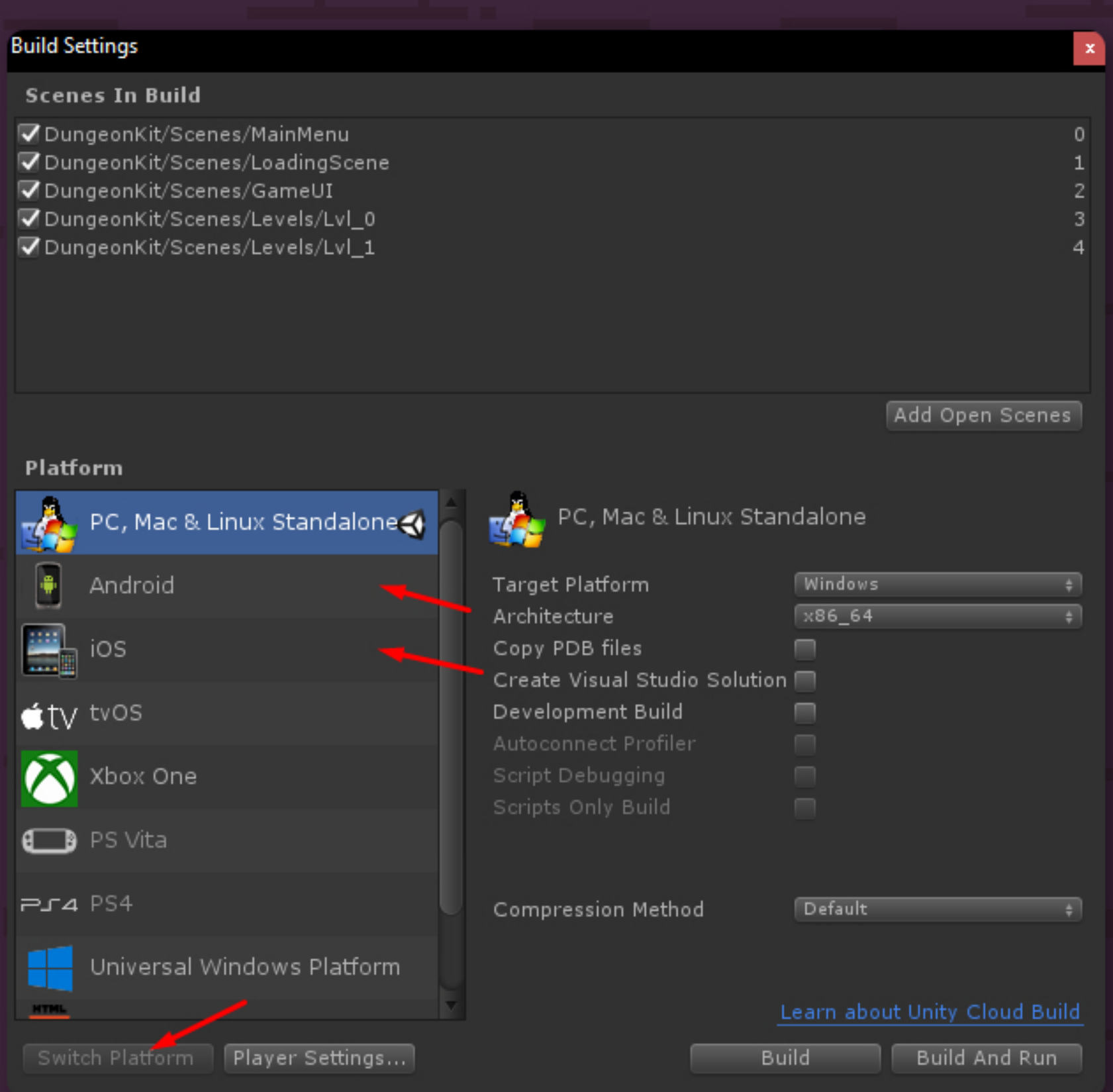
3. and call from any script –

```
AudioManager.Instance.Play(audioSource, AudioManager.Instance.playerDamage, false);
```

Asset

Platform

To activate mobile control, switch platform



When you turn on the mobile platform, the joysticks will run when you go to levels