Adding your character to your scene

This page explains how to add your character to a scene in the TopDown Engine.

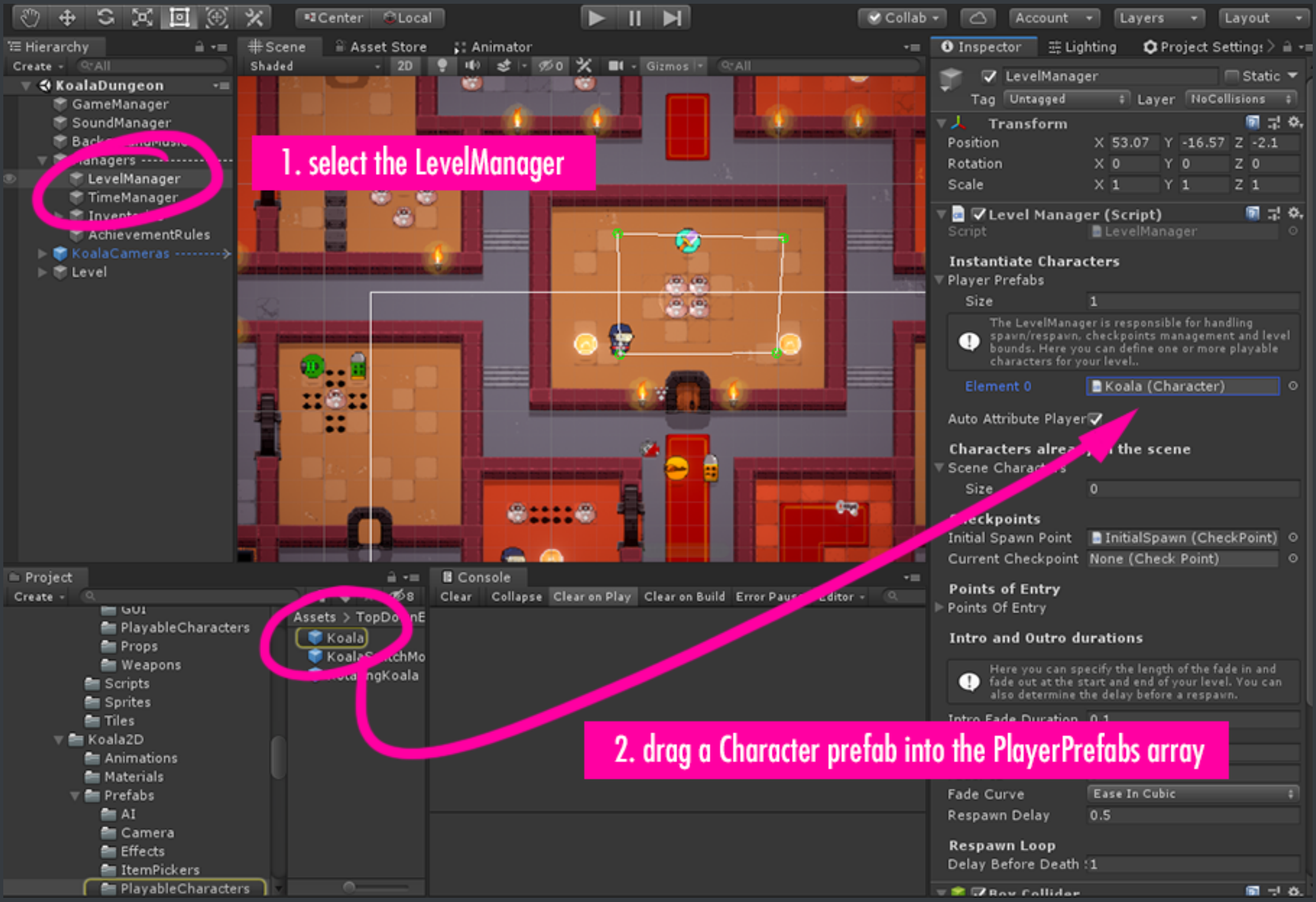
Introduction

Once you've created your character, you'll probably want to play with it. There are multiple ways to do it, and this page explains the different ways you can add characters, whether they're playable or not, to your scene.

The Level Manager

The **LevelManager** class is a very central class in the TopDown Engine. It's responsible for spawning the player(s), handling checkpoints as well as points of entry in a scene. It basically defines the rules of your level as a whole. It will also be used as the main way for classes throughout the engine to communicate with the player.

It's recommended to have the LevelManager in charge of **spawning** your character. This means it won't be present in the scene until your press Play, and will be **instantiated** at runtime. To let your LevelManager know what character it should instantiate, select the LevelManager object in your scene's hierarchy, unfold the PlayerPrefabs array, and just drag and drop a Character prefab from your project view into the Element0 field.



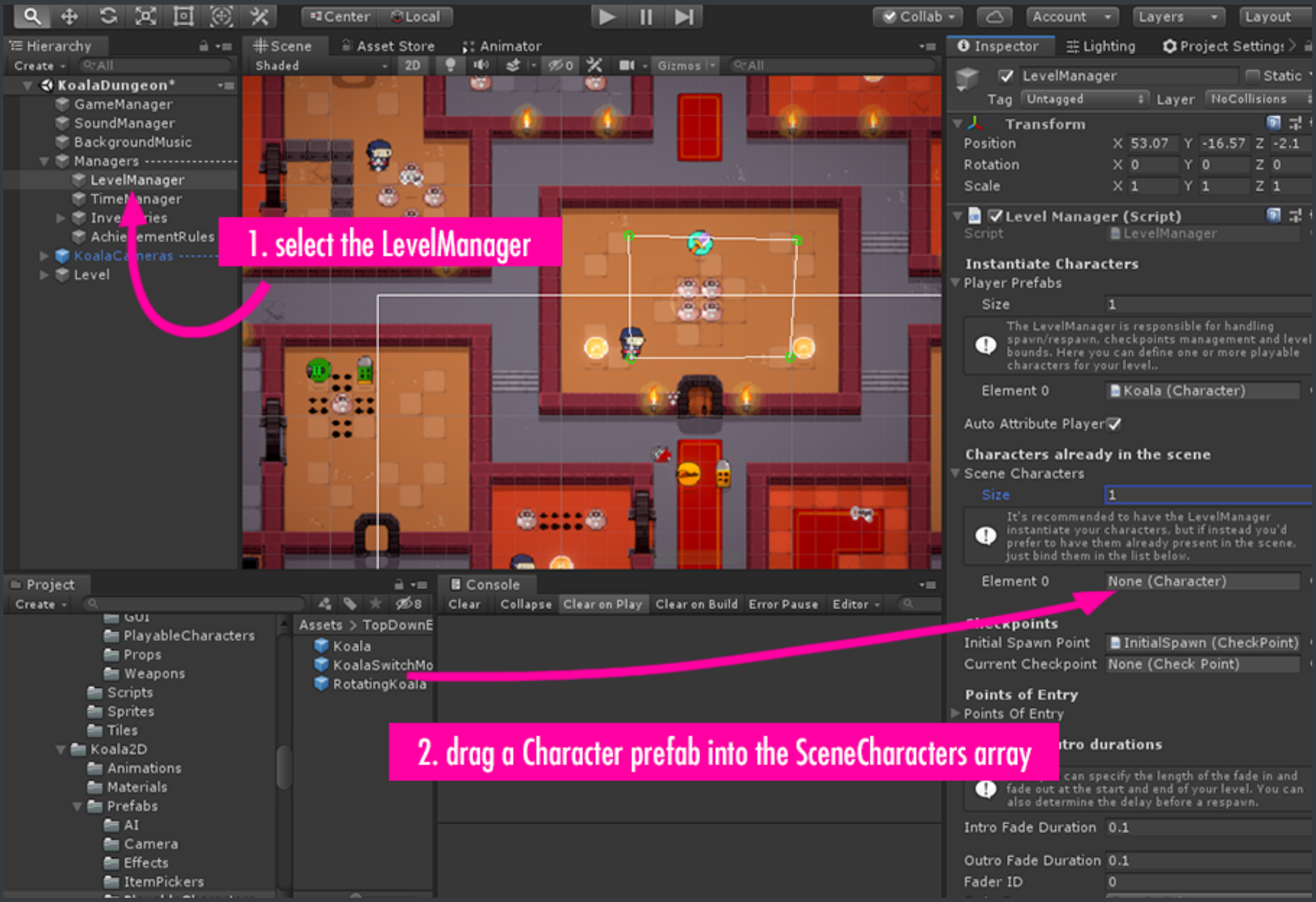
LevelManager
Tag Untagged Layer NoCollisions
Transform
Position X 53.07 Y -16.57 Z -2.1
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Level Manager (Script)
Script LevelManager
Instantiate Characters
Player Prefabs
Size 1
The LevelManager is responsible for handling spawn/respawn, checkpoints management and level bounds. Here you can define one or more playable characters for your level..
Element 0 Koala (Character)
Auto Attribute Player ✓
Scene Characters
Size 0
Initial Spawn Point InitialSpawn (CheckPoint)
Current Checkpoint None (Check Point)
Points of Entry
Points Of Entry
Intro and Outro durations
Here you can specify the length of the fade in and fade out at the start and end of your level. You can also determine the delay before a respawn.
Fade Curve Ease In Cubic
Respawn Delay 0.5
Respawn Loop
Delay Before Death 1

Binding your character to the LevelManager.

You can learn more about the LevelManager in the dedicated part of the documentation.

Having your character already present in your scene

While the method above is the recommended one, you can also have your character already present in your scene and still have the LevelManager reference it. For this to work, you'll need to have a Character object in your scene, and drag it into the LevelManager's SceneCharacters array.



LevelManager — Tag: Untagged — Layer: NoCollisions

Transform
Position X 53.07 Y -16.57 Z -2.1
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Level Manager (Script)
Script: LevelManager

Instantiate Characters
Player Prefabs
Size: 1

The LevelManager is responsible for handling spawn/respawn, checkpoints management and level bounds. Here you can define one or more playable characters for your level..

Element 0: Koala (Character)
Auto Attribute Player ✓

Characters already in the scene
Scene Characters
Size: 1

It's recommended to have the LevelManager instantiate your characters, but if instead you'd prefer to have them already present in the scene, just bind them in the list below.

Element 0: None (Character)

Checkpoints
Initial Spawn Point: InitialSpawn (CheckPoint)
Current Checkpoint: None (Check Point)

Points of Entry
Points Of Entry

Intro and outro durations

can specify the length of the fade in and fade out at the start and end of your level. You can also determine the delay before a respawn.

Intro Fade Duration: 0.1
Outro Fade Duration: 0.1
Fader ID: 0

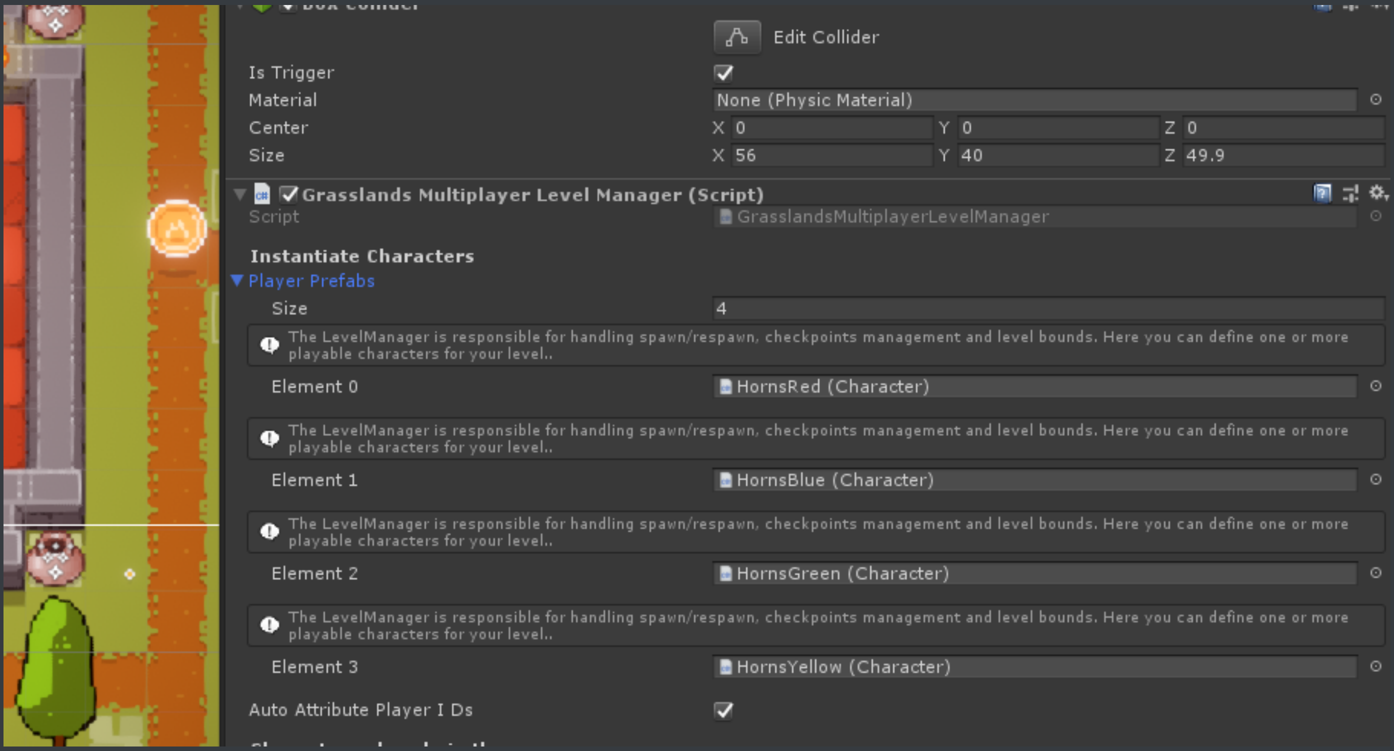
Binding a character from your scene to the LevelManager.

Other characters (AIs, NPCs)

For other characters, that are not meant to be controlled by the player, it's much easier, you just have to drag their prefab somewhere in the scene.

Multiplayer

In multiplayer, you'll have to let your MultiplayerLevelManager know what characters to instantiate. This is done just like for a single character :



Edit Collider

Is Trigger ✓
Material None (Physic Material)
Center X 0 Y 0 Z 0
Size X 56 Y 40 Z 49.9

✓ Grasslands Multiplayer Level Manager (Script)
Script GrasslandsMultiplayerLevelManager

Instantiate Characters
Player Prefabs
Size 4
The LevelManager is responsible for handling spawn/respawn, checkpoints management and level bounds. Here you can define one or more playable characters for your level..
Element 0 HornsRed (Character)
The LevelManager is responsible for handling spawn/respawn, checkpoints management and level bounds. Here you can define one or more playable characters for your level..
Element 1 HornsBlue (Character)
The LevelManager is responsible for handling spawn/respawn, checkpoints management and level bounds. Here you can define one or more playable characters for your level..
Element 2 HornsGreen (Character)
The LevelManager is responsible for handling spawn/respawn, checkpoints management and level bounds. Here you can define one or more playable characters for your level..
Element 3 HornsYellow (Character)
Auto Attribute Player I Ds ✓

Binding character to the LevelManager in a multiplayer context.

Accessing the player character via script

The best way to access the player(s) via script is via the **LevelManager** class, as it keeps a reference to it at all times (or to them if you have more than one playable character in your level). From any class, you can access the player array via :

```
LevelManager.Instance.Players
```

Players is an array, so if you only have one playable character, you can access it via :

```
LevelManager.Instance.Players[0]
```

And it's an array of Characters, so from there it's very easy to interact with your player :

```
// this will freeze your main character
LevelManager.Instance.Players[0].Freeze();

// sets the main character's max Health to 50
LevelManager.Instance.Players[0].gameObject.MMGetComponentNoAlloc<Health
>().MaximumHealth = 50;

// forces the character to dash
LevelManager.Instance.Players[0].gameObject.MMGetComponentNoAlloc<Charac
terDash>().StartDash();
```