COMP2150 Level Design Assessment Reference Guide

This document is intended as a reference guide for the various elements within the Level Design Assessment project. It is recommended you have a play around with the project while referencing this guide before trying to create your level. This document assumes basic knowledge of Unity from COMP1150.

For additional details on assessment requirements, marking criteria and submission, please refer to iLearn and the discussion forums.

Setup

Please make sure you are using **Unity 2022.3.18** before getting started. You can find the correct version of Unity in the Unity Download Archive. Once that's installed, make sure you have the most up-to-date version of your assignment repo (from Github Classroom) and open it through Unity Hub.

Using the 2D Game Kit

Open the Unity project inside the repository. It contains four folders:

2DGameKit: The default assets for the project, including sprites, scripts, etc. You shouldn't modify anything in this folder.

Prefabs: The **Required** and **Optional** features for your game. You shouldn't modify any of these prefabs, but should be using them when building your level.

Examples: This folder contains two scenes: A Sample Scene and a Prefab Demonstration scene. Both are worth investigating and playing with to learn how the various elements of the project work.

Scenes: This folder contains two scenes: an End of Game scene (that you don't need to modify) and a Level Design scene, which is the scene you will be building your level in. The scene contains the three keys and door that you will need to include in your final level, so you can relocate them to fit your design.

If you accidentally delete your scene, you can create a new one by pressing **Kit Tools > Create Template Scene… > Create Template Scene in Assets.** Don't forget to rename this scene to "Level Design".

Tilemap Tutorial

Before you can get started on the assignment, you will need to learn a new tool which will help you develop your levels: Tilemaps.

Tilemaps are a feature in Unity that makes building 2D tile-based games easier. Many games represent the world as a 2D grid of tiles. Laying out these tiles manually can be very tedious. The Tilemap editor streamlines this process.

You will use Tilemaps to create your level's basic geometry, so it is important you get familiar with the tool. Unity Learn also hosts a great tutorial series covering this feature, which you can find here.

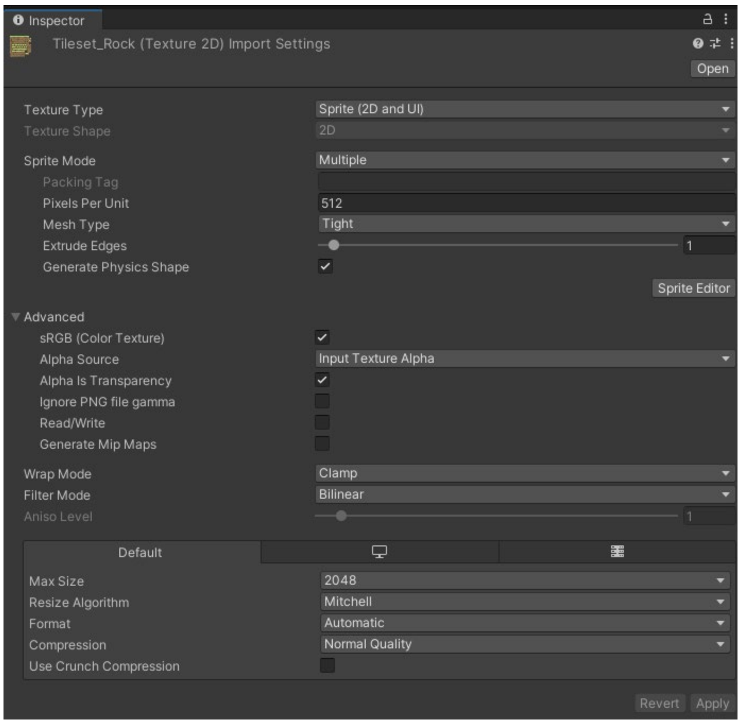
The project provided to you already has the tilemap set-up, so you only need to worry about placing tiles. However, if you want more in-depth knowledge of how to create a tilemap from scratch, please see the document on iLearn entitled "Bonus: Tilemaps From Scratch".

Tilemap Components

Creating a tilemap requires a few different components. These have all been set-up for you, but it's a good idea to get familiar with what they do.

Tilemap Sprites

A tilemap is made from a collection of sprites. These sprites are generally stored in a sprite sheet, which is a single image split into many different sprites. You can see the sprite sheet used in this project by navigating to **Assets > 2DGameKit > Art > Sprites > Environment > Tilesets > Tileset_Rock.png.** You don't need to alter this file at all, but you can have a look to understand how the tilemap is constructed. It should look like this in the Inspector:

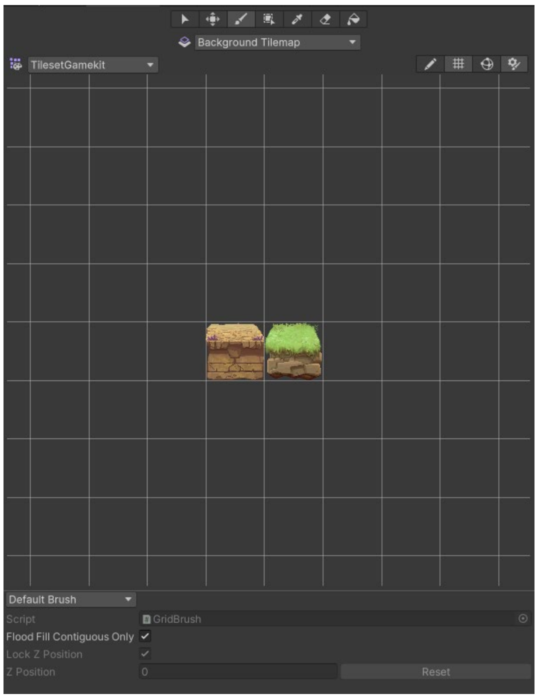


Tileset_Rock (Texture 2D) Import Settings

Texture Type	Sprite (2D and UI)
Texture Shape	2D
Sprite Mode	Multiple
Packing Tag	
Pixels Per Unit	512
Mesh Type	Tight
Extrude Edges	1
Generate Physics Shape	✓
Advanced	
sRGB (Color Texture)	✓
Alpha Source	Input Texture Alpha
Alpha Is Transparency	✓
Ignore PNG file gamma	
Read/Write	
Generate Mip Maps	
Wrap Mode	Clamp
Filter Mode	Bilinear
Aniso Level	1
Max Size	2048
Resize Algorithm	Mitchell
Format	Automatic
Compression	Normal Quality
Use Crunch Compression	

Tile Palette

A “tile palette” can be thought of like a painter’s colour palette, where they select different colours to paint onto the canvas. The tile palette allows you to select the tiles you want to use and then place them on the screen. We’re going to come back to how to do this, but for now you can navigate to or open the Tile Palette panel **(Window > 2D > Tile Palette).** You should see something like this, showing the tiles available to us:

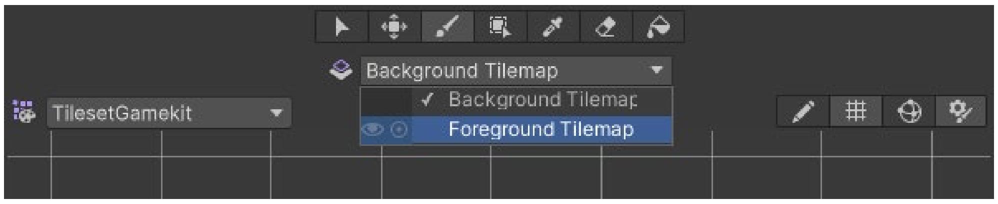


Script: GridBrush
Flood Fill Contiguous Only: ✓
Lock Z Position: ✓
Z Position: 0

This is one of the two default tile palettes the project comes with, **TilesGameKit.** The other palette, **TilesetRockWatersBlockers** can be selected from the drop down in the top left of the Tile Palette panel, but we don’t recommend using this one.

Tilemaps

To add tiles to the scene, we need a tilemap. Think of a tilemap as another layer of your scene that you can place tiles on. Your scene already contains two tilemaps: a **Foreground Tilemap** and a **Background Tilemap**. Any edits made using your Tile Palette will only effect the selected Tilemap. You can change between Tilemaps by selecting them at the top of the Tile Palette panel from the **Active Target** drop-down.

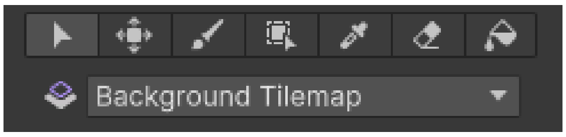


These two tilemaps may appear the same, but have distinct properties. The **Background Tilemap** contains no colliders, and will draw tiles a darker colour at the back of your scene, so you can use this to add more depth to your level. The **Foreground Tilemap** will be drawn at the front of your scene, and these tiles have colliders. This allows you to create the geometry of your level.

Note that you can also select your tilemaps in the hierarchy. Be careful when working on your project that you don't accidentally select the wrong tilemap, always double-check your Active Tilemap is set to the one you want before making changes.

Using the Tilemap Tool

Now that you've been introduced to the various elements of the Tilemap Tool, it's time to explore how to use it. With your Tilemap Palette Panel open, take a look at the tool bar along the top. You should see seven icons:



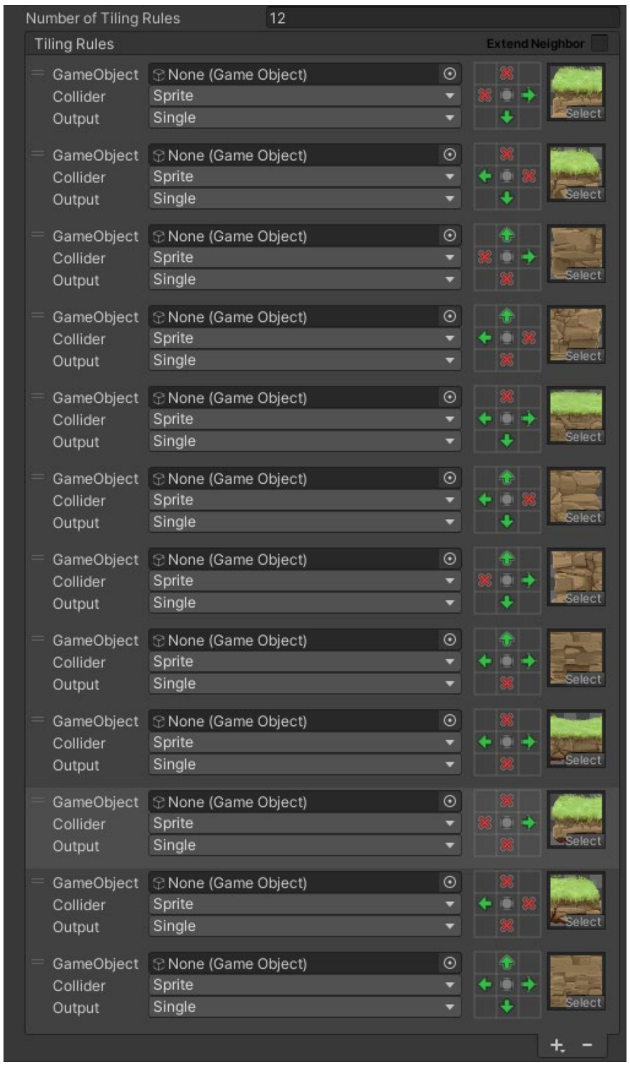
These tools are as follows:

- **The Selection tool (S):** Selects an area of tiles in the active tilemap in your scene.
- **The Move tool (M):** Moves selected tiles (note, you need to select the tiles with the selection tool first) on the active tilemap in your scene.
- **The Brush tool (B):** Paint the tile selected in the Tile Palette panel onto the active tilemap in your scene.
- **The Box tool (U):** Paint a rectangle of tiles onto the active tilemap in your scene.
- **The Pick/Marquee tool (I):** Set the brush to a tile from the active tilemap in your scene (instead of selecting it in the Tile Palette panel).
- **The Erase tool (D):** Erase tiles.
- **The Fill tool (G):** Fills an area of the active tilemap in your scene with the selected tile, using the mouse position and current tiles as reference points.

Try creating a few different configurations of tiles using these tools. First, select one of the two tiles in the TilePalette window and try drawing them in your Scene view with the brush tool (note: you cannot add tiles by clicking on the Game view).

The first thing you'll notice is that as you draw, the tiles will change to correspond to those around them. That is because these are **Rule Tiles,** which means they are configured to change based on tiles adjacent to them on the same tilemap (a rule tile on the Foreground Tilemap will not respond to a tile on the Background Tilemap, and vice versa).

You can take a look at the rules for these tiles by navigating to **Assets > 2DGamekit > Art > TilemapPalettes > Tileset** and selecting either **Tileset Alien Rules or Tileset Rock Rules** and viewing them in the inspector. You should see something like this:


Number of Tiling Rules 12
Tiling Rules
Extend Neighbor

GameObject None (Game Object)
Collider Sprite
Output Single

GameObject None (Game Object)
Collider Sprite
Output Single

GameObject None (Game Object)
Collider Sprite
Output Single

GameObject None (Game Object)
Collider Sprite
Output Single

GameObject None (Game Object)
Collider Sprite
Output Single

GameObject None (Game Object)
Collider Sprite
Output Single

GameObject None (Game Object)
Collider Sprite
Output Single

GameObject None (Game Object)
Collider Sprite
Output Single

GameObject None (Game Object)
Collider Sprite
Output Single

GameObject None (Game Object)
Collider Sprite
Output Single

GameObject None (Game Object)
Collider Sprite
Output Single

GameObject None (Game Object)
Collider Sprite
Output Single

It's a good idea to get a handle on how this works, to the extent of understanding why a certain tile might look the way it does. However, if you want to learn how to create your own Rule Tiles, see the "Bonus: Tilemaps From Scratch" document on iLearn.

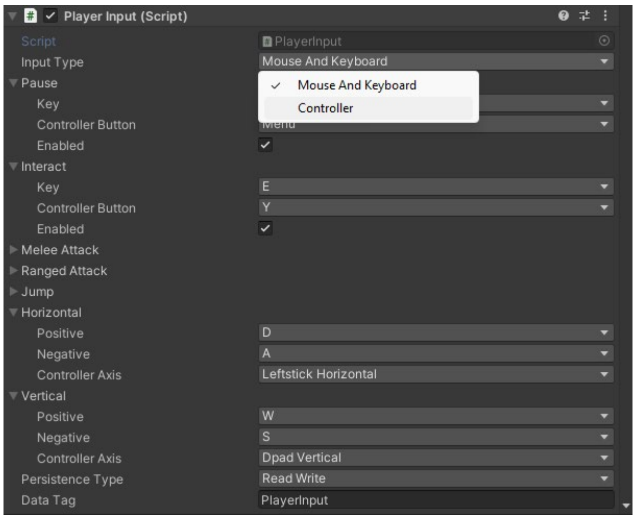
These tools can take a while to master. Experiment by drawing a scene using the tiles. Get familiar with all these tools, including switching between Tilemaps.

To stop editing the tilemap, you need to close the Tilemap Palette panel (it's annoying, we know). Make sure you do this before you start placing your prefabs, or you might edit your tilemap unintentionally. You can open the panel again when you want to edit your tilemap.

Note: You don't have to ever select the Erase tool, as you can simply **hold Shift while using the Brush or Box tool**. As a result, you can erase large areas quickly by selecting the Box tool and holding shift while drawing a box. Give it a try, but be careful using it to delete large areas, as it is quite resource intensive and can (rarely) cause Unity to lag.

Controls

The player can be controlled using a keyboard or gamepad. You can switch between these two input methods by selecting the Ellen prefab (**Assets > 2DGamekit > Prefabs > Ellen**) and setting the Input Type in the **Player Input** component in the Inspector. Do not modify these controls, as your marker will be working with multiple projects and not have time to learn new schemes each assignment.



Player Input (Script)

Script	PlayerInput
Input Type	Mouse And Keyboard
Pause	
Key	
Controller Button	Menu
Enabled	✓
Interact	
Key	E
Controller Button	Y
Enabled	✓
Melee Attack	
Ranged Attack	
Jump	
Horizontal	
Positive	D
Negative	A
Controller Axis	Leftstick Horizontal
Vertical	
Positive	W
Negative	S
Controller Axis	Dpad Vertical
Persistence Type	Read Write
Data Tag	PlayerInput

Keyboard control	Gamepad control	Action performed
A/D	Left analog stick	Move left/right
Space	B	Jump
S	Down on the D-pad	Crouch (combine with jump to move through a Passthrough Platform)
K	A	Swing staff (after pickup)
O	Right bumper	Shoot gun (after pickup)

The Hierarchy

The hierarchy is set up to contain all the important components to make the game work. You shouldn't need to worry about most of these. Objects immediately underneath **---System---**, cannot be children of a parent, as they contain helper code components that utilize something called DontDestroyOnLoad, such as the background music or the inventory.

The main elements you should familiarise yourself with are:

1. **Ellen**: The player character, found under **---PlayerAssets---.** You shouldn't need to change any of their settings, but you can place them at whatever starting point you want.
2. **Tilemap**: The tilemaps for the scene. These can be found under **---LevelAssets--- > Environment > TileMapGrid.** We have included two tilemaps for you: the Foreground Tilemap for creating the main geometry of your level, and a Background Tilemap for adding background flourishes to give your level more depth and a sense of place.

Note: The Tilemap has a Composite Collider 2D. This can slow down editing when the map gets large. It may be a good idea to simply deactivate the collider while editing the tiles if your computer has issues, just don't forget to reactivate it when you're done.

3. **---LevelAssets---**: This is where you should put the elements that make up your scene. You can add them as children, or add them immediately underneath this empty GameObject.

Prefabs

The following is a list of prefabs and where to find them within the Assets folder.

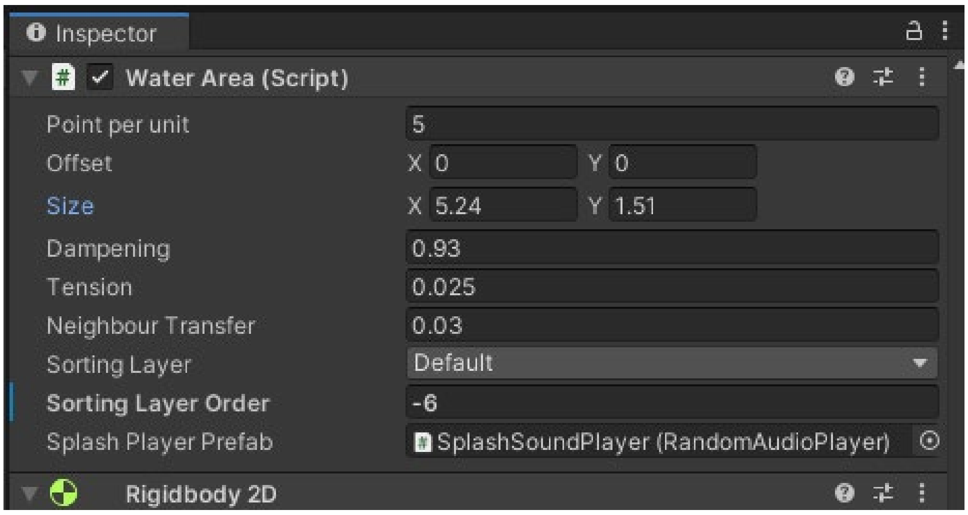
All of these features have an implementation in the "Prefab Demonstration" scene (found in **Assets > Examples**). Keep in mind that the level includes **example implementations**. There are definitely more possible implementations than shown in that level!

Required Prefabs

Acid

Prefab:
Prefabs/Required/Acid
The Acid hurts the player when they fall into it. The player will lose one (1) point of health and will be sent back to the last checkpoint (or the beginning of the level).

If you want to increase the size of the Acid, please **do <u>not</u> use the scale tool**, but instead edit the following script variables in the object, as seen below:



Point per unit	5
Offset	X 0 Y 0
Size	X 5.24 Y 1.51
Dampening	0.93
Tension	0.025
Neighbour Transfer	0.03
Sorting Layer	Default
Sorting Layer Order	-6
Splash Player Prefab	SplashSoundPlayer (RandomAudioPlayer)

Checkpoints

Prefab:
Prefabs/Required/Checkpoint

If the player falls in acid they will restart at the most recent checkpoint they have touched, as long as they still have some health (if they have 0 health, they will restart at the beginning of the level). Note that checkpoints should be placed on the ground. Checkpoints light up upon activation.

Chomper

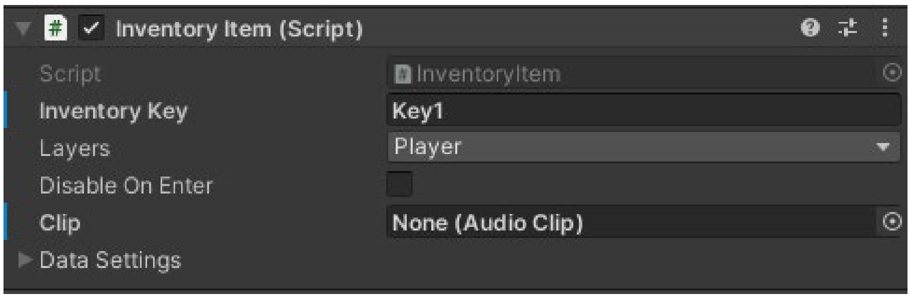
Prefab:
Prefabs/Required/Chomper
An enemy that bites/melee's the player, dealing one (1) damage per hit.

Health Pickup

Prefab:
Prefabs/Required/HealthPickup
The Health Pickup adds one (1) point of health to the player, by default.

Key

Prefab:
Prefabs/Required/Key
A collectible key. You need to pick up three to open the Key Door (Note: **Clip** should be left blank).



Inventory Item (Script)

Script	InventoryItem
Inventory Key	Key1
Layers	Player
Disable On Enter	☐
Clip	None (Audio Clip)
▸ Data Settings	

Key Door

Prefab:
Prefabs/Required/KeyDoor
A door that opens when all three keys are collectable, transitioning to a different scene.

Note: The Key Door has already been configured to work in the assignment template to transition between your Main level & the Game Over scene. Please check the Level Design Discussion Forum on iLearn if you are encountering problems.

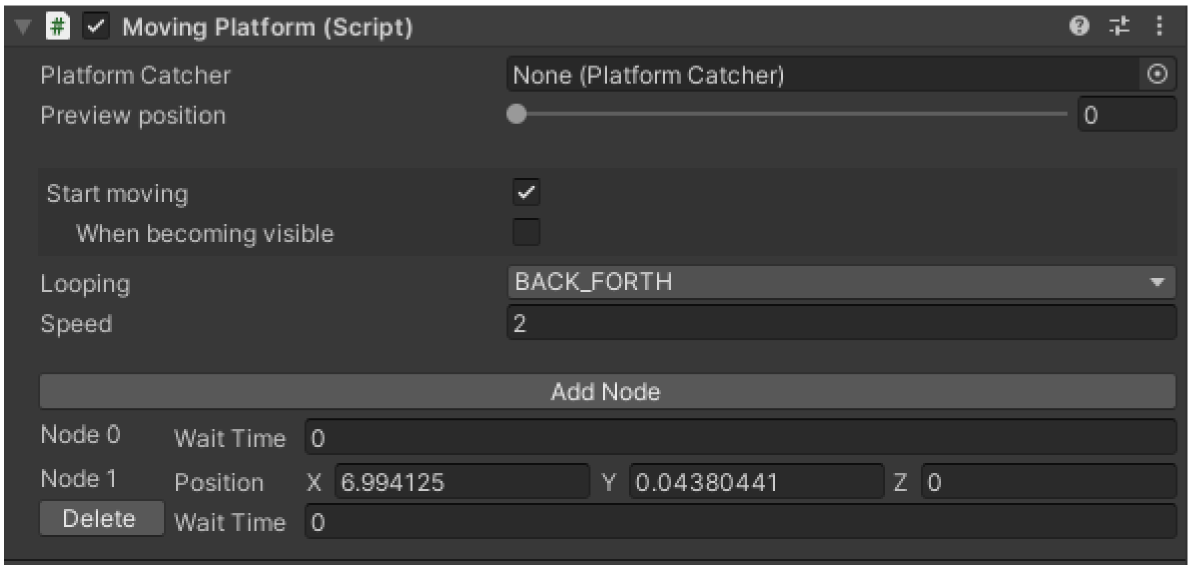
Passthrough Platform

Prefabs:
Prefabs/Required/PassThroughPlatform | Prefabs/Required/PassThroughPlatformLong
This is a platform that the player can move upwards through. If the player is crouching on top of a Passthrough Platform and presses the jump button, they will instead fall through it.

Moving Platforms

Prefabs:
Prefabs/Required/MovingPlatform | Prefabs/Required/MovingPlatformLong
A variant of the passthrough platform that follows a defined path, using nodes. Multiple nodes can be used, as well as multiple loop types (Once, back and forth). You can modify these settings in the Moving Platform Script Component.

Note: We have found setting the Speed variable to anything above 2 results in the player not moving with the platform properly when moving down. These speeds should be set to 2 by default, but it is worth checking if you have unexpected results. This becomes especially important if you have a platform activated by a pressure pad like in the “Prefab Demonstration” scene.


Moving Platform (Script)

Platform Catcher	None (Platform Catcher)
Preview position	0
Start moving	✓
When becoming visible	
Looping	BACK_FORTH
Speed	2

Node 0	Wait Time	0
Node 1	Position	X 6.994125 Y 0.04380441 Z 0
Delete	Wait Time	0

Spikes

Prefab: Prefabs/Required/Spikes
When the player collides with the Spikes, the player loses one (1) point of health.

Spitter

Prefab: Prefabs/Required/Spitter
An enemy that spits acid at the player.

Weapon Pickups (Staff and Gun)

Prefab: Prefabs/Required/Staff_Pickup | Prefabs/Required/Gun_Pickup
The player does not initially have a weapon. There are two weapon pickups which vary on which weapon they give the player.

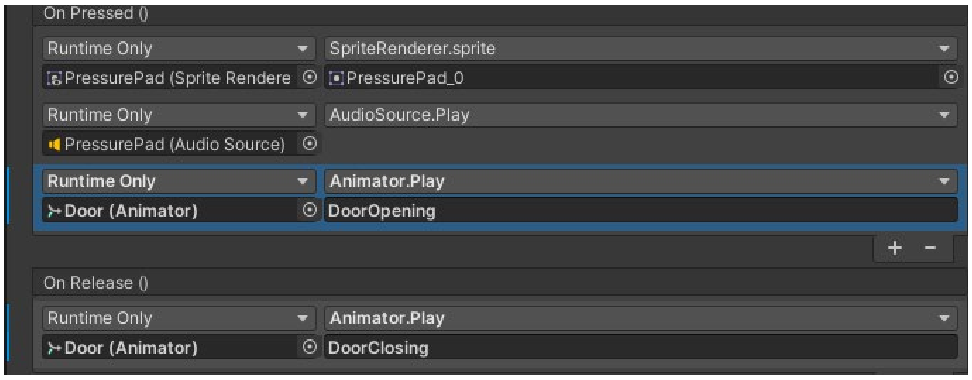
- *Staff_Pickup* gives the player the staff.
- *Gun_Pickup* gives the player the gun.

Optional Prefabs

Trigger Door

Prefab: Prefabs/Optional/TriggerDoor
A barrier that cannot be bypassed without triggering a thing (For example, a pressure pad, a switch, or a trigger collider, although you could probably find other ways to trigger it open).

Can be open or closed using the Animator (**Animator.Play**(***DoorOpening*** *or* ***DoorClosing***). By default, this transition is triggered by the Trigger Condition SwitchFlipped.



On Pressed ()

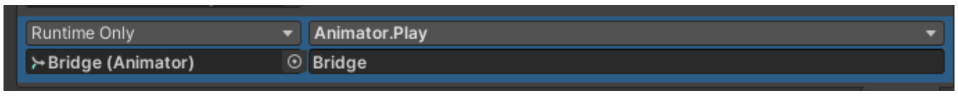
Runtime Only | SpriteRenderer.sprite
PressurePad (Sprite Rendere | PressurePad_0
Runtime Only | AudioSource.Play
PressurePad (Audio Source)
Runtime Only | Animator.Play
Door (Animator) | DoorOpening

On Release ()

Runtime Only | Animator.Play
Door (Animator) | DoorClosing

Bridge

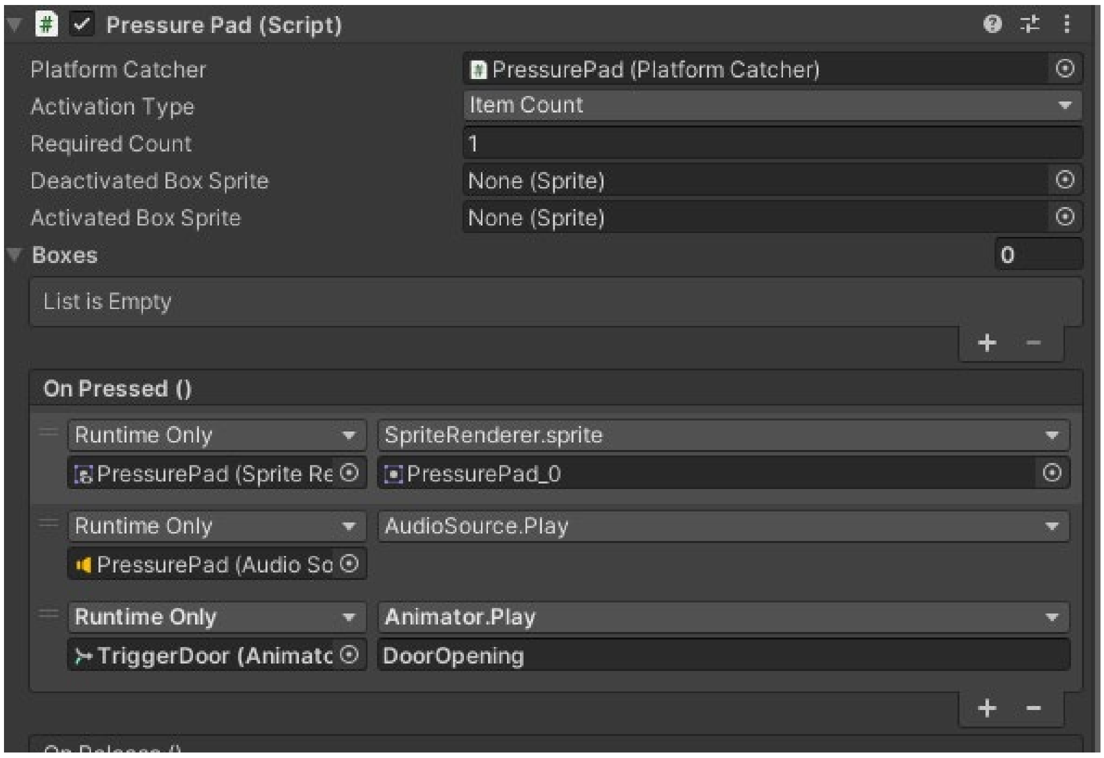
Prefab: Prefabs/Optional/Bridge
A barrier that cannot be bypassed without triggering a thing, similar to the TriggerDoor. Can be moved once using **Animator.Play**(***Bridge***).



Runtime Only | Animator.Play
Bridge (Animator) | Bridge

Pressure Pad

Prefab: Prefabs/Optional/PressurePad
An object that when the player stands on, can trigger some event (such as a bridge activating, or a door opening). This can be set by adding the object's animator to the OnPressed() condition in the Inspector, as shown below.



Pressure Pad (Script)

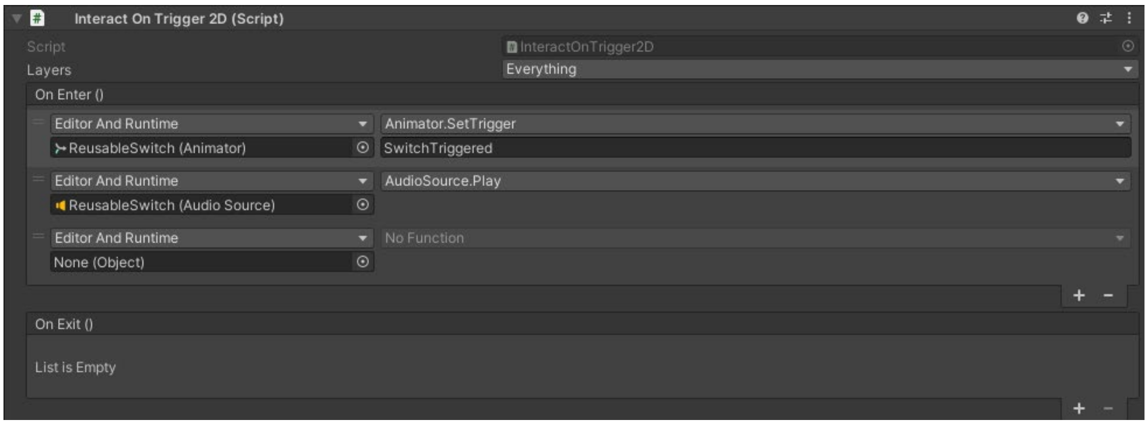
Platform Catcher: PressurePad (Platform Catcher)
Activation Type: Item Count
Required Count: 1
Deactivated Box Sprite: None (Sprite)
Activated Box Sprite: None (Sprite)
Boxes: 0
List is Empty

On Pressed ()
Runtime Only | SpriteRenderer.sprite
PressurePad (Sprite Re | PressurePad_0
Runtime Only | AudioSource.Play
PressurePad (Audio So
Runtime Only | Animator.Play
TriggerDoor (Animatc | DoorOpening

On Released ()

Reusable Switch

Prefab: Prefabs/Optional/ReusableSwitch
A switch which, like the pressure pad, activates an object when it collides with another object. You can specify what object activates it (such as the player) by setting the “Layers” drop-down in the inspector. By default, it reacts to Everything.



Interact On Trigger 2D (Script)

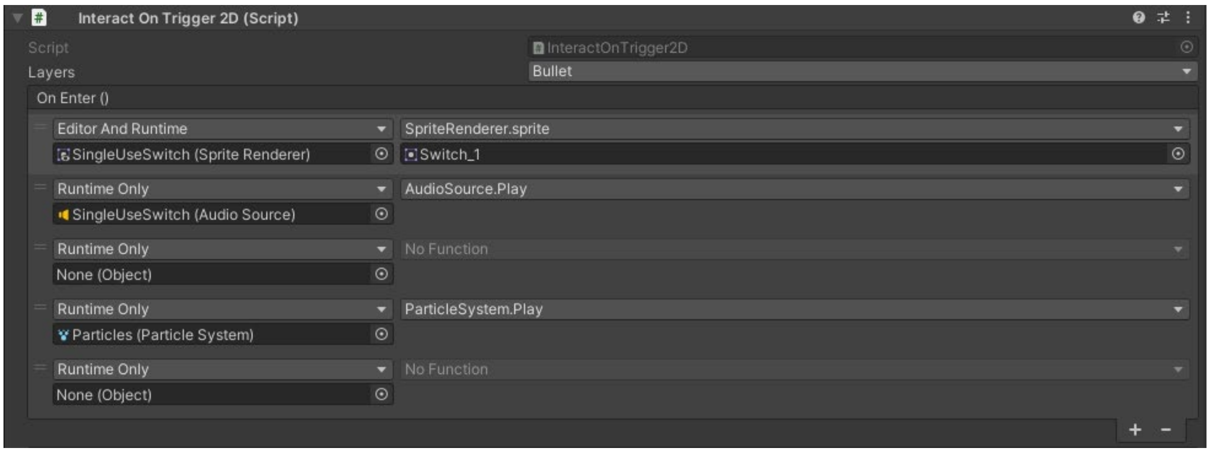
Script: InteractOnTrigger2D
Layers: Everything

On Enter ()
Editor And Runtime | Animator.SetTrigger
ReusableSwitch (Animator) | SwitchTriggered
Editor And Runtime | AudioSource.Play
ReusableSwitch (Audio Source)
Editor And Runtime | No Function
None (Object)

On Exit ()
List is Empty

Single Use Switch

Prefab: Prefabs/Optional/SingleUseSwitch
A switch which operates like the Resuable Switch, but only switches “On”. You can specify what object activates it by setting the “Layers” drop-down in the inspector. By default, it reacts to bullets.



Interact On Trigger 2D (Script)

```
Script        InteractOnTrigger2D
Layers        Bullet
On Enter ()
Editor And Runtime                      SpriteRenderer.sprite
SingleUseSwitch (Sprite Renderer)       Switch_1
Runtime Only                            AudioSource.Play
SingleUseSwitch (Audio Source)
Runtime Only                            No Function
None (Object)
Runtime Only                            ParticleSystem.Play
Particles (Particle System)
Runtime Only                            No Function
None (Object)
```

Pushable Block

Prefab: Prefabs/Optional/PushableBlock
A large block that the player can push back and forth by running into it. Can be stood on!

Destructible Column

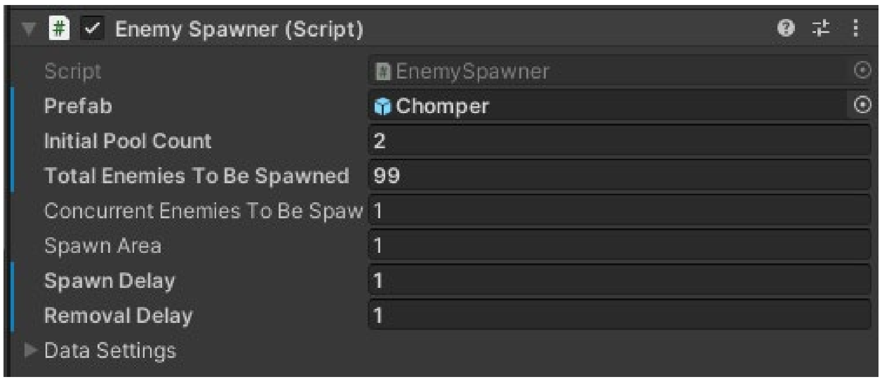
Prefab: Prefabs/Optional/DestructableColumn
A barrier that can be opened by hitting it with the Staff. Has one (1) health by default.

Destructible Wall

Prefab: Prefabs/Optional/DestructableWall
Another barrier that can be opened by hitting it with the Staff, but thicker than the column. Has five (5) health by default.

Enemy Spawner

Prefab: Prefabs/Optional/EnemySpawner
Can be used to spawn groups of enemies. This GameObject has several properties, such as making sure only x amount of enemies can be spawned at a time. Below is an example of an implementation set to ensure that only one enemy is spawned at a time and that enemy will always be there.



Enemy Spawner (Script)

```
Script                          EnemySpawner
Prefab                          Chomper
Initial Pool Count              2
Total Enemies To Be Spawned     99
Concurrent Enemies To Be Spaw   1
Spawn Area                      1
Spawn Delay                     1
Removal Delay                   1
Data Settings
```