Getting Started

Thank you again for supporting Tactics Toolkit. This document is a step-by-step guide for setting up a new scene using this asset pack. It's come to my attention that some users have taken the provided example scenes and replaced assets within them. This is completely fine, but I do feel it may not be the most optimal way to learn the intricacies of the systems within.

Folder Structure

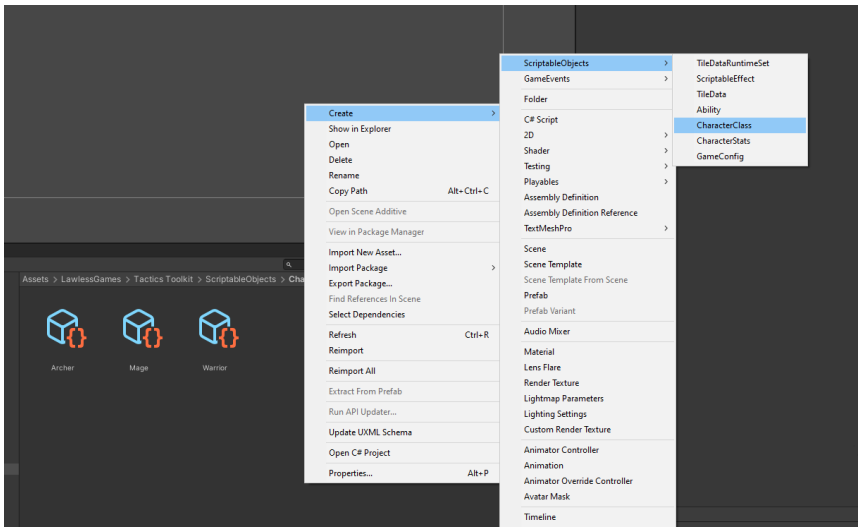
- **Abilities:** The abilities folder contains the ability shapes I use for my example abilities. I explain the Shapeparser in the other documentation, so look if you want to know more.
- **Documents:** The documents folder is where you found this file, so I don't think I need to talk too much. Make sure you follow the Read Me steps before you get started because it can cause some issues.
- **Events:** The events folder contains all the events used in the project. There are a lot. I try to separate them by folder as best I can, but there is still a lot.
- **Images:** The images folder contains all the images I use for my demo scenes. I encourage you to use your own but feel free to look at these if you wish to copy the settings.
- **Prefabs:** One of the most essential folders to familiarise yourself with. You'll be interacting with this folder a lot when creating your own scenes.
- **Scenes:** The scenes folder contains all eight demo scenes. Go through each one if you'd like. Or the final one has everything the others have.
- **Scriptable Objects:** Another important folder to look at. Here, I store all the scriptable objects I use, including Character Classes, Abilities and TileDatas.
- **Scripts:** The scripts folder contains all the scripts I use throughout the project. Of course, you'll need to look at these eventually, but it might be worthwhile setting up a scene first and looking at scripts as needed.

Creating a Character/Enemy

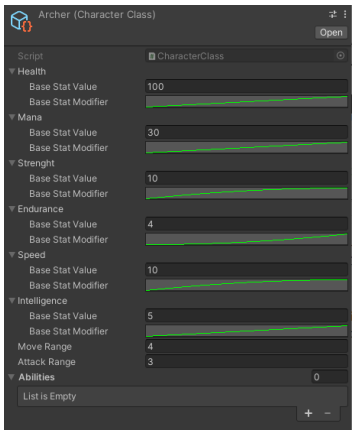
A character or enemy is built up of 3 parts: the character controller, the character class and its abilities. Let's get started by creating a character class and some abilities.

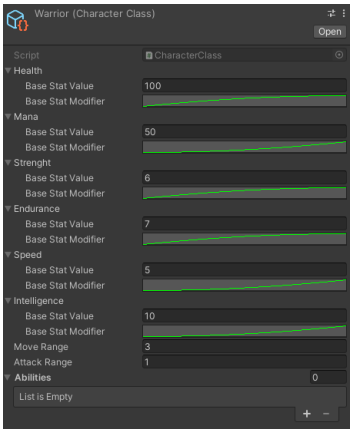
Step 1: Creating a Character Class

Right-click within your project, and go to create, scriptable objects, and character class.



Now set up the stats of the class you created. This impacts how your character's stats will be affected when they level up (See scene 2). Set the starting stats for level 1, and the animation curves affect the weighted random of how much the stat increases. These stats are just examples. You might need to change the script to suit the game you're making.

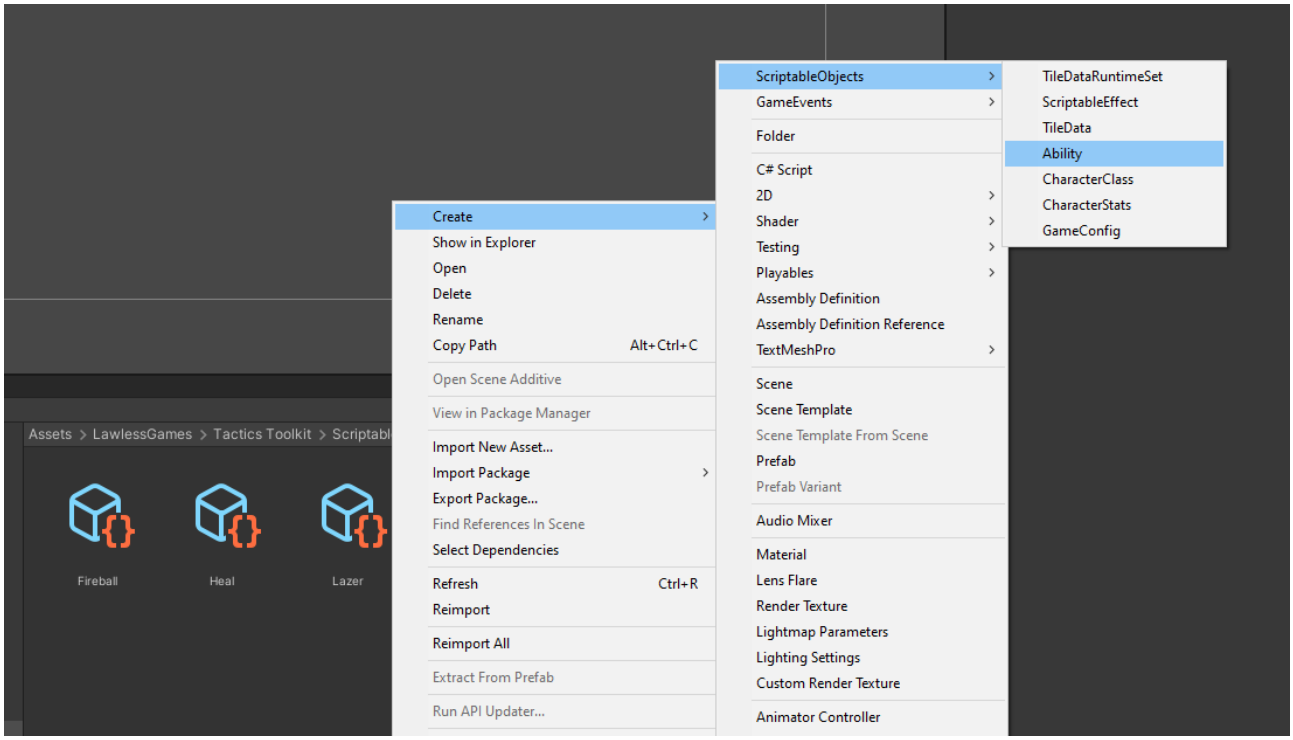




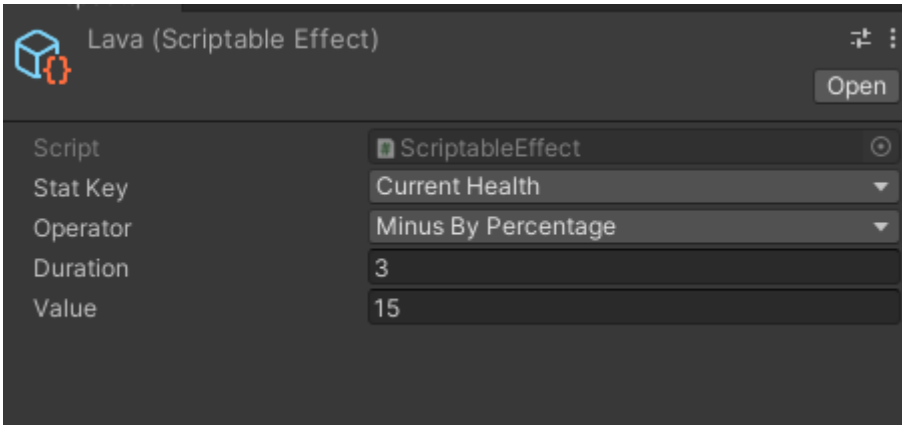
As a starting point, use an concaved curve for strong starts and convex curves for weaker stats.

Step 2: Abilities

Strategy games are always better with a few abilities, so let's get creating. Let's recreate the fireball ability I used in the demo for this example. **Right-click, go to Create, Scriptable Objects, Ability.**



There's a few other objects the ability needs to get the full power. Lets start with the effect. **Right click, go to Create, Scriptable Objects, Effect.** Change the properties to match my lava effect or make up your own.

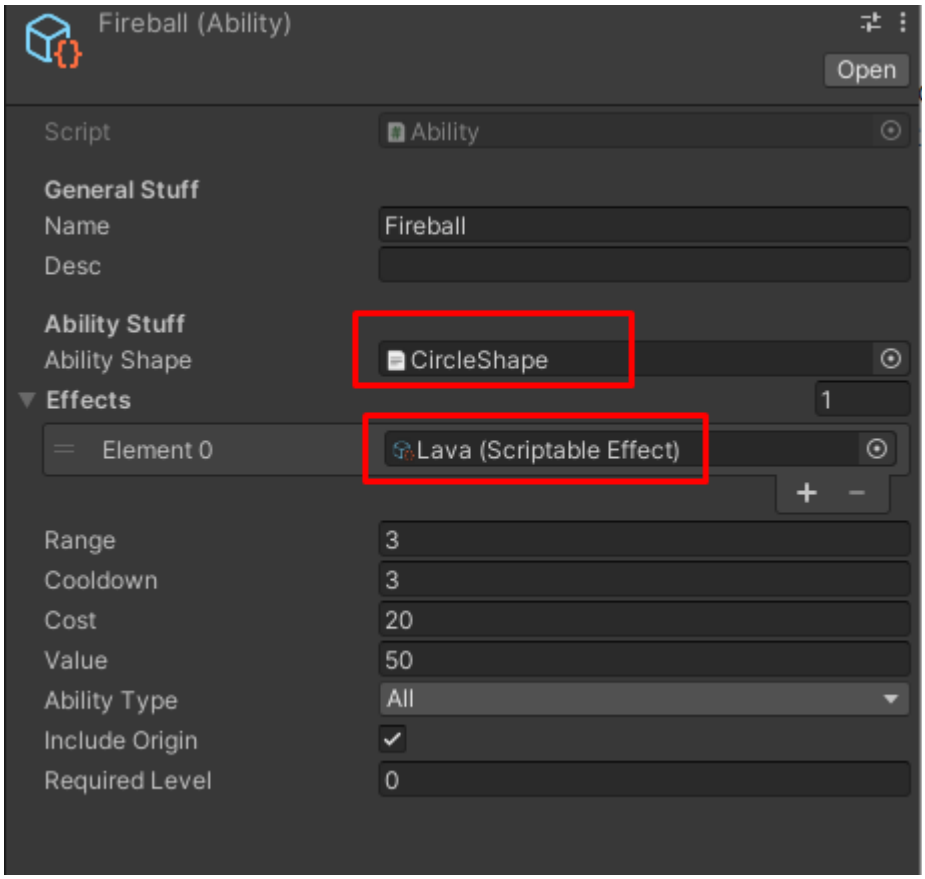


Now, this is good to go to attach to the new ability you created.

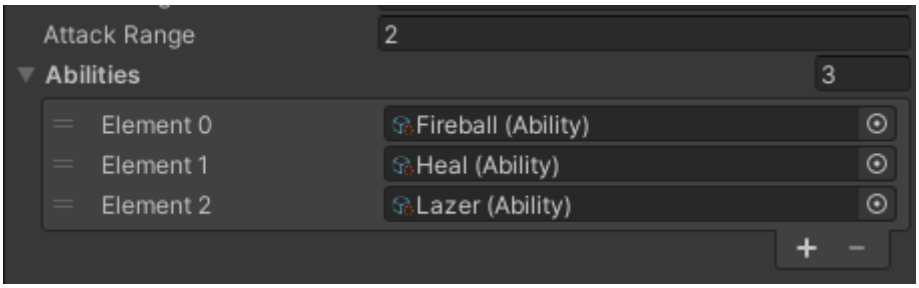
Next, we need the shape of the ability. Again, the rules are specified in the other document, so refresh yourself if you wish. For this, we want a simple circle shape. Or make it bigger or smaller if you like by adding more layers of 2's.

```
0 0 2 0 0
0 2 2 2 0
2 2 1 2 2
0 2 2 2 0
0 0 2 0 0
```

Now lets set up the ability. Attach the text file and effect to the ability and change the properties to match my fireball abiltiy. Or play around with it to see what happens. I explain each property in the other document.



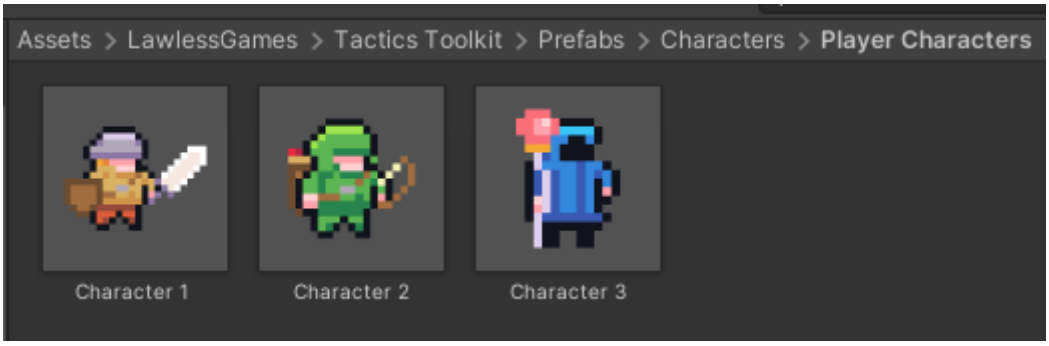
Lastly add the ability to the Abilities List for the character class you created earlier.



Step 2: Create a character prefab

In the prefabs, characters folder, there are two prefabs. Base and Enemy Base. These are to be treated as a parent to your character variants. This helps because if you ever need to change a character prefab, like adding a new component, for example. If you add it to the Base, then it will be added to every variant of that Base.

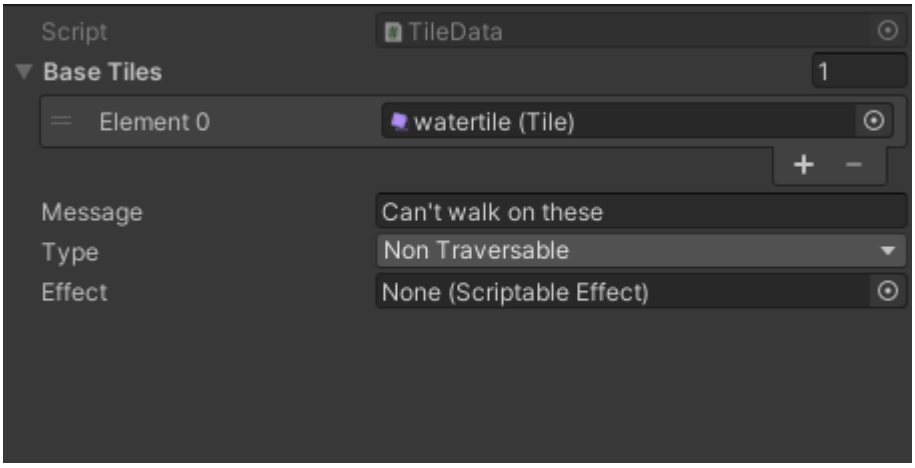
To create a variant, drag a base into the hierarchy. Now all you need to do is change the graphic to suit your character and attach the class you created earlier. Update the level if you want, but everything else should be fine and rename the game object. Then, drag it back into the project and create the prefab variant. For example, all my character prefabs are in the Player Characters folder.

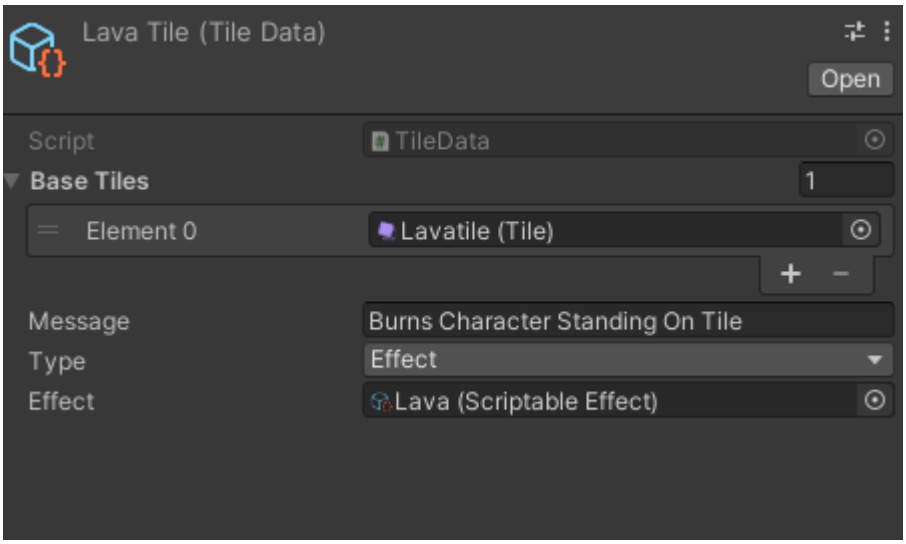


Creating The Tile Data List

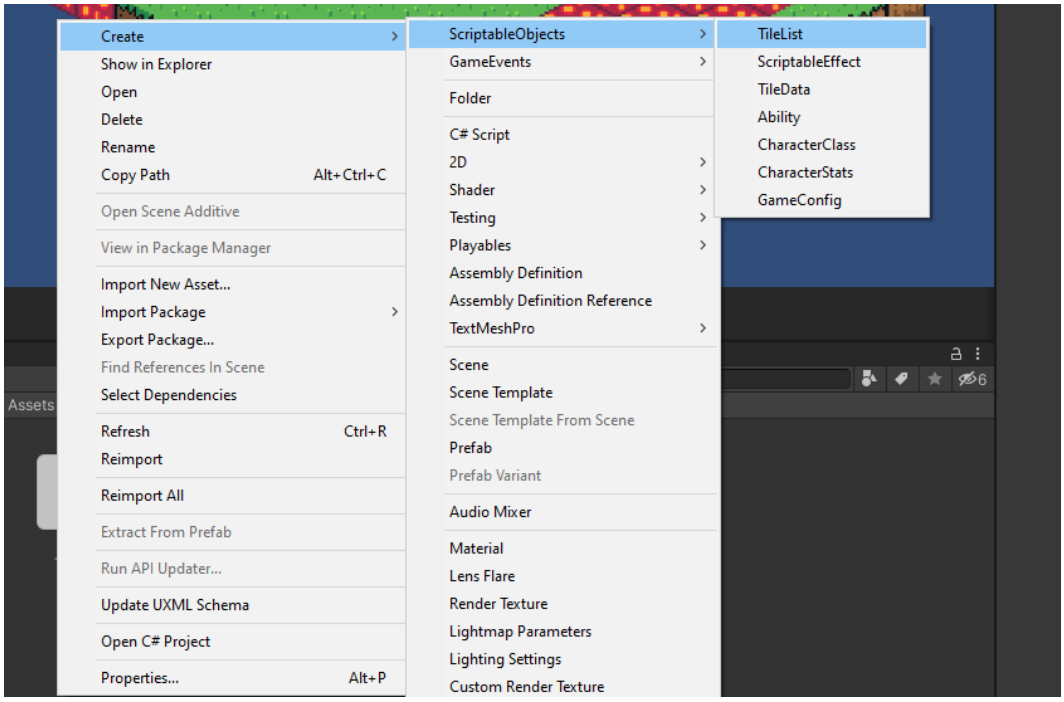
Before starting with TileDatas, you'll want to create the tile palette you use for your game. Tile Data uses the Tile objects that are created when you add images to the palette.

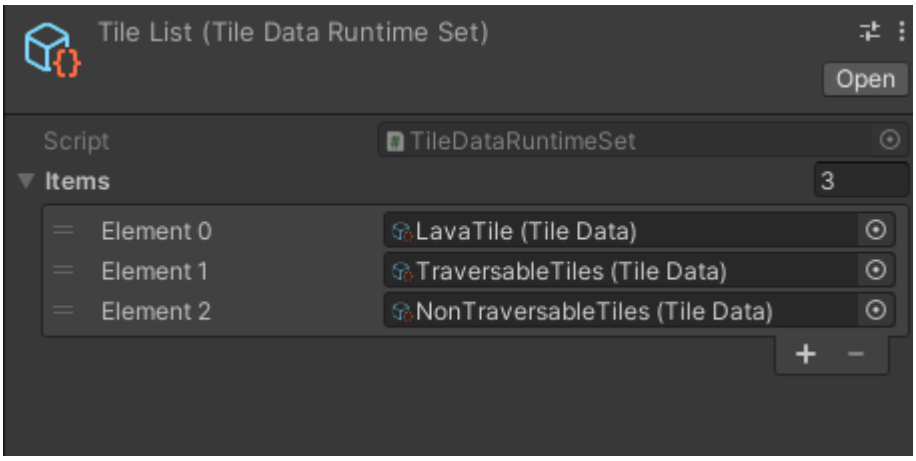
All the TileDatas I've created are in the Scriptable objects folder and are great examples of this. Even though I created a TileData for the basic grass tiles, that isn't necessary. There are two types of tile data you need. A non-traversable list of tiles you cannot walk on. And some effect tile data. You can create an effect like the ability and add it to a tile data. Then every time a character walks on the tile, the effect will be attached to them.





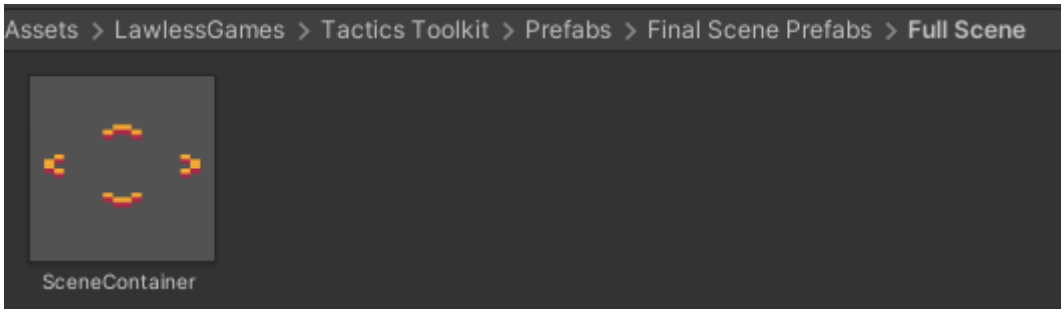
Last create a TileList and add all the TileDatas to the list.



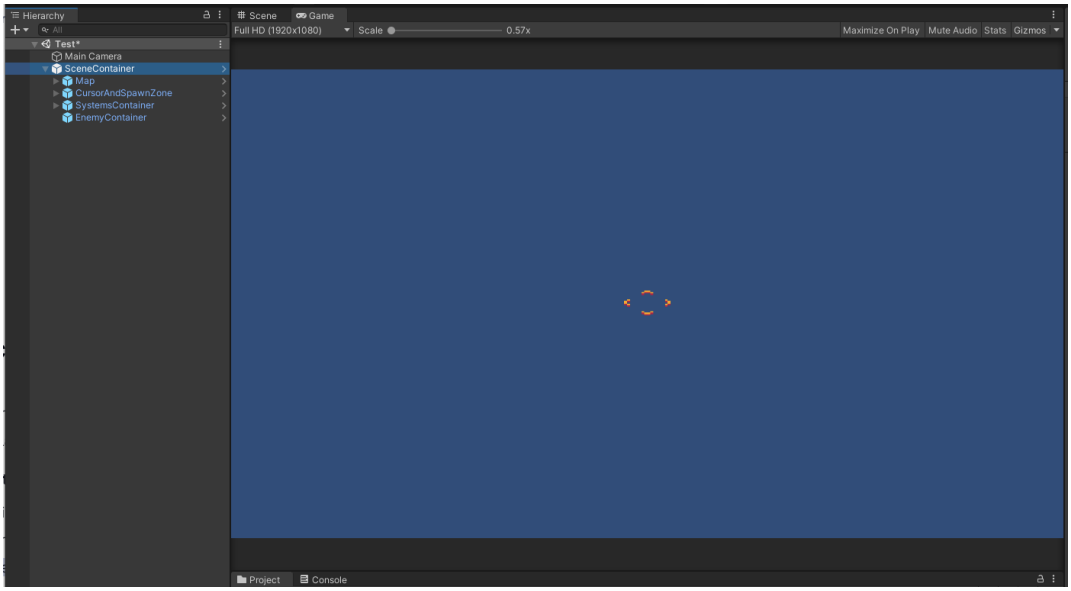


Now Lets Create A Scene

Thankfully creating a new scene is very simple. Within the **Prefabs** folder there is a **Final Scene Prefabs** folder that contains all the systems of the final scene in the toolkit.

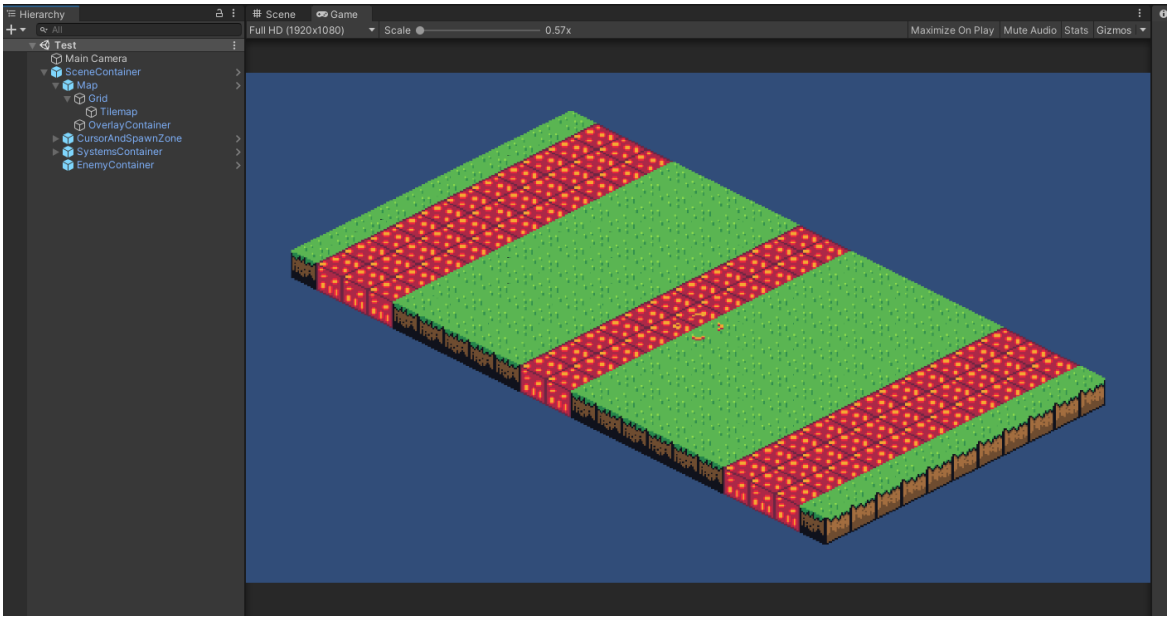


There is a folder with all the components seperated or in one big prefab. For simplicity, lets drag the SceneContainer into an empty scene.

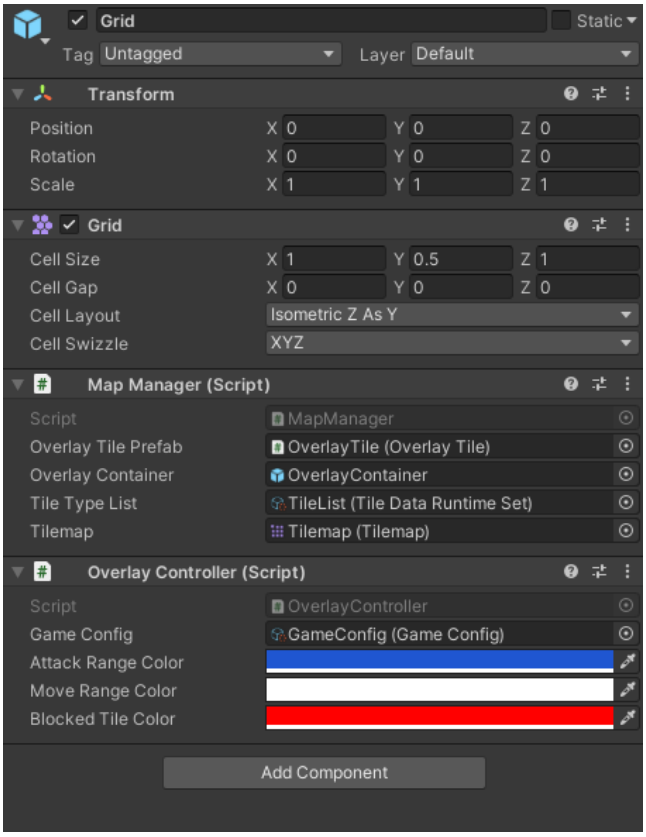


There are 3 things we need to do to make this scene a level.

First, let's create a map. To do that, open up Map and click on Grid. Then, use the Grid as a normal Tilemap Grid and draw a map you want to use.

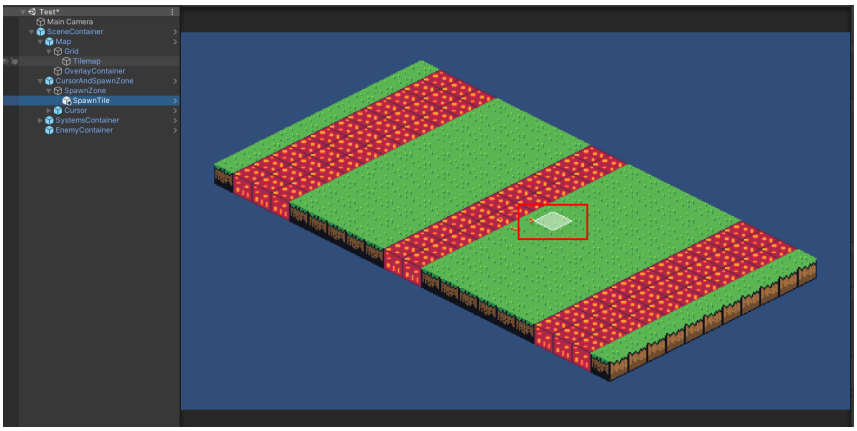


And attach the TileList you created earlier to the MapManager component attached to the Grid.

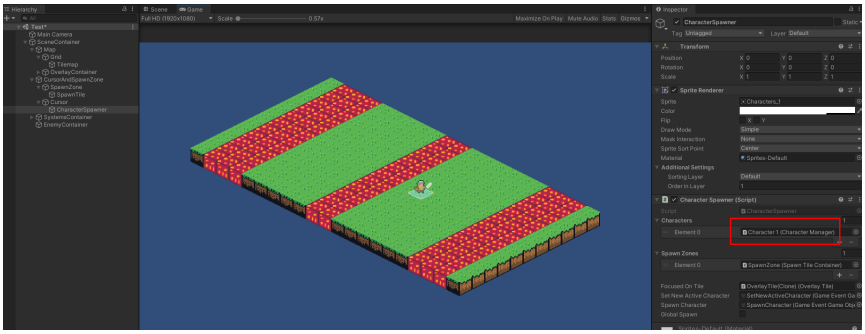


Now if you press play the overlay container will be filled with Overlay tiles that we can interact with.

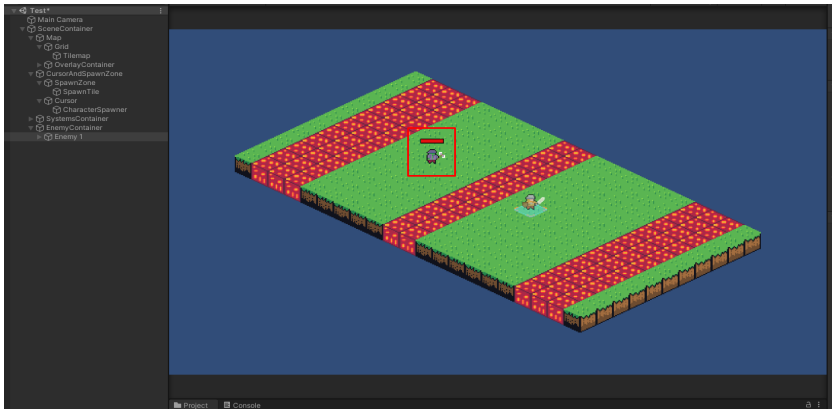
Next, let's create a spawn zone to spawn in characters. In the **Prefabs** folder, find the **SpawnTile**. Next, open up **CursorAndSpawnZone** and drag a **SpawnTile** into the **SpawnZone** game object. Position it wherever you want. Also, if you want to skip this step, you can open up **Cursor -> CharacterSpawner** and set **GlobalSpawn** to true.



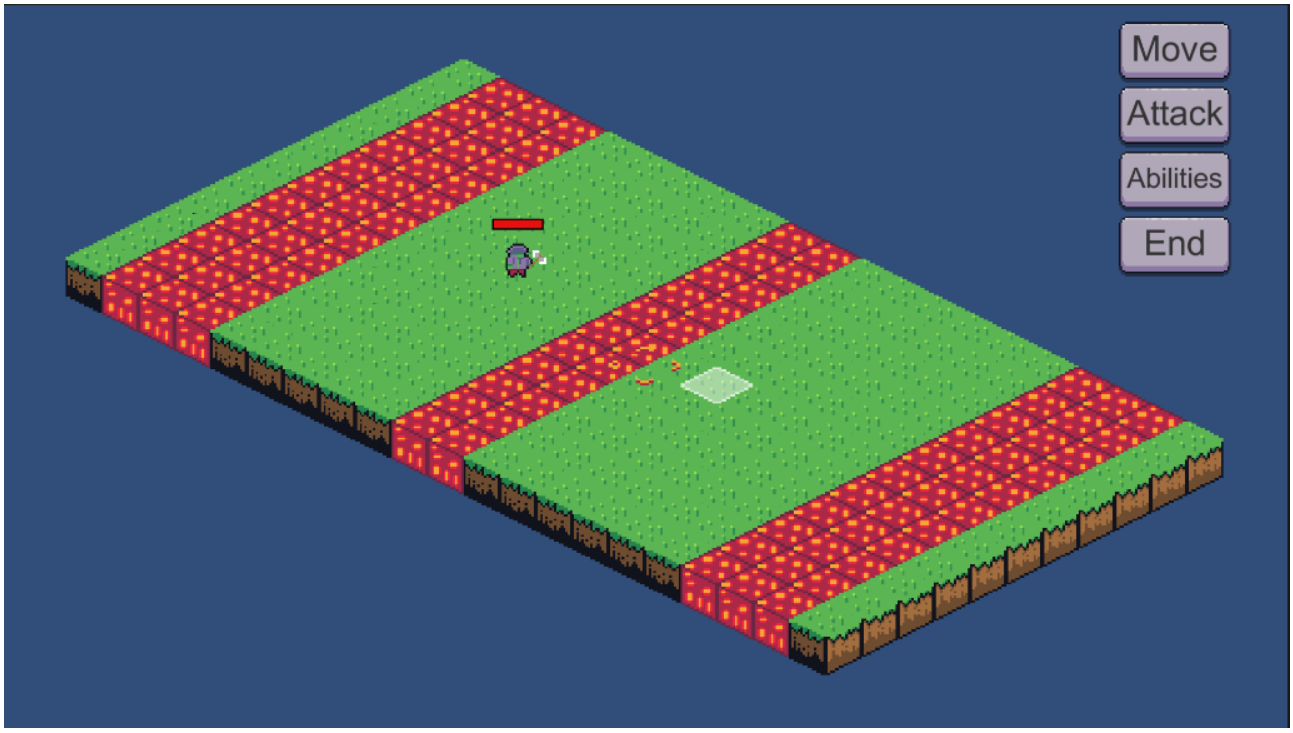
Next, on the CharacterSpawner, we just need to add the characters we want to be able to create in these levels. You could skip this step and drag your characters onto the scene, but this is more realistic for these types of games.



Now let's drag in some Enemies. Create an enemy the same way you create a character, except with the **Enemy Base**. Then, drag the enemies into the **EnemyContainer**.



Finally, you can drag in the button container to start playing. Feel free to replace the graphics on the buttons with whatever you want, but they are all already hooked up with the events they need.



And that's it. Your level is ready to go. Press play and enjoy your custom build TRPG level.