Game Engine Basic Design

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generatedThe World properties Editor:

This will be a small window with variables to edit which will affect the game being created or the look of the editor. Some Values include the game window aspect ratio, the current layer in use, the opacity of all objects not on this layer etc. you can also edit the cameras position and scale to move around the game.

Graphical user interface, application

Description automatically generated

The Object Inspector Window:

This window will be used to see the Attributes and properties of the current Active Object. You will be able to add Component Classes to the Object, to customise the object to the desired use. This window will also have a tab that shows all the Objects in the game. You will be able to group objects together and parent objects to each other. You will also be shown each objects name. When created the name of the objects will always be the same but can be changed in the Object inspector.

Graphical user interface, application

Description automatically generatedObject import window:

Graphical user interface, application

Description automatically generatedFor this window you will be able to import textures. You will select a image file and then you will be asked how many sprites there are in the image and the width and height of a sprite. The engine will then display them in this window. You can then click on a sprite and place it in the game viewport. This will bring up the inspector for that object where you can change its basic attributes and add components to the object.

Game Viewport:

The game viewport window will display the game you are creating. It will have a play and stop and pause button. Stop will reset the game back to the state before the play button was pressed. The stop and pause buttons will be inactive unless the play button has been pressed, at this point the play button becomes inactive until the pause/stop button is pressed. The pause button will stop the game in its current state and save to copy with the new positions of everything that moved (mainly for simulations).

Window Popups:

Inspector Window:

In the Inspector Window you will be able to add component to the selected Object in the game. You will be able to add component like a player Controller, Physics Collider etc.

Import Window:

In the Import Window you will be shown a popup where you can select to import an image file, a sprite file, or a sound file.

A screenshot of a computer

Description automatically generated with medium confidence

If import is selected, a file explorer will be opened where you can select the image file to import. When a file has been selected, it is copied to the assets directory in the game save file and loaded from there. This is to make it easy to save a game file on one computer and transfer it easily without the program erroring because of a missing file. Once selected a window will open showing the Image and 3 sets of changeable variables. Theses can be changed to create textures for use in the game.

A screenshot of a computer

Description automatically generated with medium confidence

When the Create button is pressed, a “Spritesheet” is created taking the three attributes used in the texture creation window is created Once created, there will be a new “collapsing Header” in the Textures Tab of the Import window. This will contain all the textures for the imported image. Each imported image will get its own collapsing header to separate different textures and make textures easier to find.

A screenshot of a computer

Description automatically generated with medium confidence

For import Sprite, the same process happens as when creating a texture accept after creating the create button, a new window is created. This window contains more fields to create animated sprites. Each sprite animation is given a trigger that will play the animation when called. This trigger may be activated by a player’s movement, button pressed or lack of movement. When creating a sprite loop, you give it the Frames (sprites + time length for it to display) and a trigger. You can choose the trigger form the Enums Triggers. I have limited each Enum to only be used once per sprite created as bugs could arise if 2 sprite animations are trying to play at the same time. I have also error trapped it so if a sprite animation doesn’t exist for a trigger, the sprite animation will stay on the current one selected. You can see the Sprite Animation loop before you add it to the final object so you can see what you have currently selected.

A screenshot of a computer

Description automatically generated with medium confidence

The next screen is where you select the component you want to add to the sprite. You can add a PlayerController or an EnemyController or no component to the sprite. You can also select the Image to use to identify the sprite in the import window. A screenshot of a computer

Description automatically generated with medium confidence

How classes will work:

Game Object class:

Main class made up of a few variables. This class will have an array of component classes. These will be used to store data like the objects position, size, colour etc. all the components will be subclasses of the main component class. I will do this so that when saving the objects and serializing to a Json format I can use 1 converter class for all of the components. The converter class will dictate how the object is serialized and deserialized.