**Description**

The game I made has three levels and a title screen. The first level is from the first Mario Bros. game. It is one static screen, two Koopas spawn from the pipes at the top of the level and react when the pow block is hit. The player can jump, move and collide with the tiles and with the pow block. There is music looping throughout the level, and additional sounds when the player jumps or Koopas spawn.

The other two levels in my game are linked, one of them is a level editor, the other is to play the level made in the level editor. The level editor is based off of the first Super Mario Bros. The sprites and enemies are all from the first level of the game. The level editor allows the player to create any levels using the provided tiles, and save the map to a file. The map is saved as a binary file, which contains both the tile data and the entity data. The map file can be sent to other people, and can be played on other machines by replacing their map file. If I were to expand the game, I think a good feature to implement would be some sort of in-game map sharing. This isn’t something I currently would know how to implement, but I’m interested in researching more into networking and multiplayer.

The second level uses a separate class for Entities as the first level. In the tutorial, a Character class is set up for Players and Enemies. This is similar, except it is an Entity class which handles general mechanics such as Movement, Collision and Death. I then use sub-classes for specific Entities which uses the information of the base Entity class. For example, the Goomba sub-class uses the base-class to check if it’s colliding left. If it is, it will start moving right, and vice versa. The sub-class also contains specific data such as Animations.

Another reason for creating a separate Entity class is that level 2 handles collision differently. Instead of having a map of 0s and 1s which correspond to an image for collision, the new collision system uses the tiles data. I cannot use a static image for the level, so the level has to be rendered using the data which is set by the level editor, so I also use this data for checking which tiles are solid.

I tried to recreate the main menu from Super Mario Bros. as accurately as possible. Due to the screen resolution not being the same size, it’s slightly different, but I like the result. The text is rendered by the Text Renderer I made, which renders text using a sprite for the font. It works well, and allows me to have symbols I usually would not be able to have. For example, at the top of the main menu it has a coin count with a coin symbol next to it. I was able to just use the font sprite and assign the coin to a specific character which made it a lot easier. It was the same with the copyright symbol and double slash underneath the logo. The logo was edited using photoshop. I used the Super Plumber Brothers font, which is the same font as the actual logo, then added effects to the text so it was identical to the actual logo text, and changed the word Super to SDL.

**Controls**

Arrow Keys Movement

Escape Pause

Mouse & Left Click Use pause menus

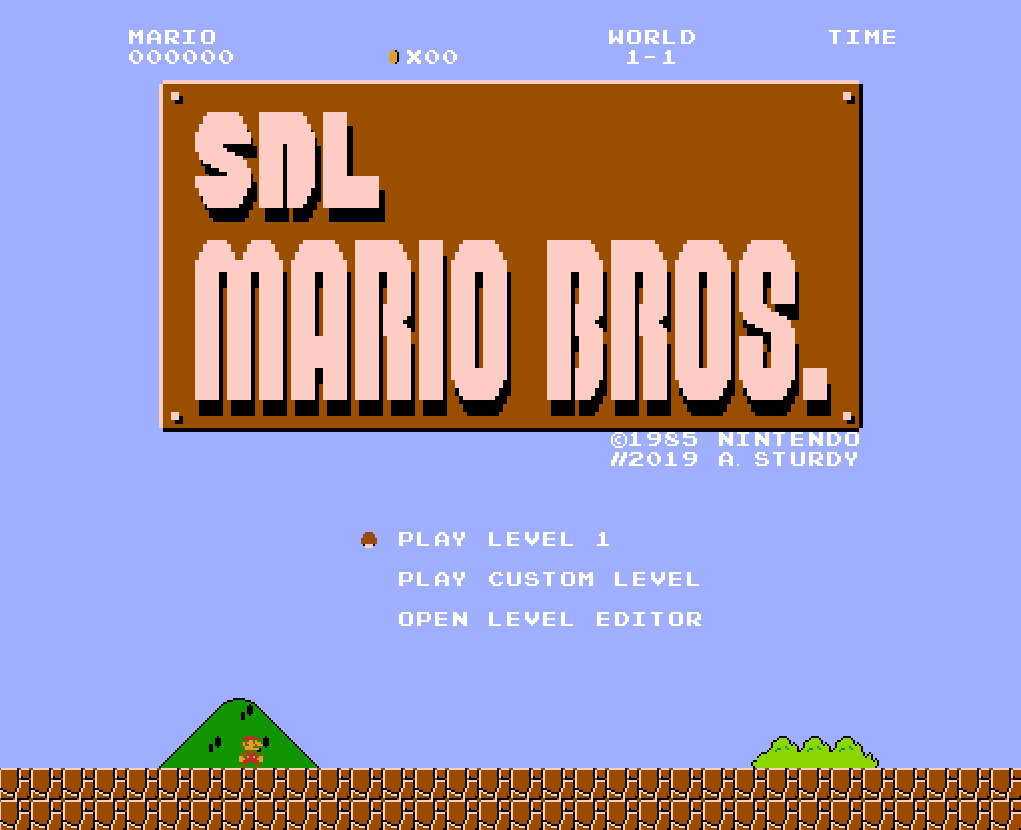
Left Click (On Select Sprites UI) Select Sprite (*Level editor*)

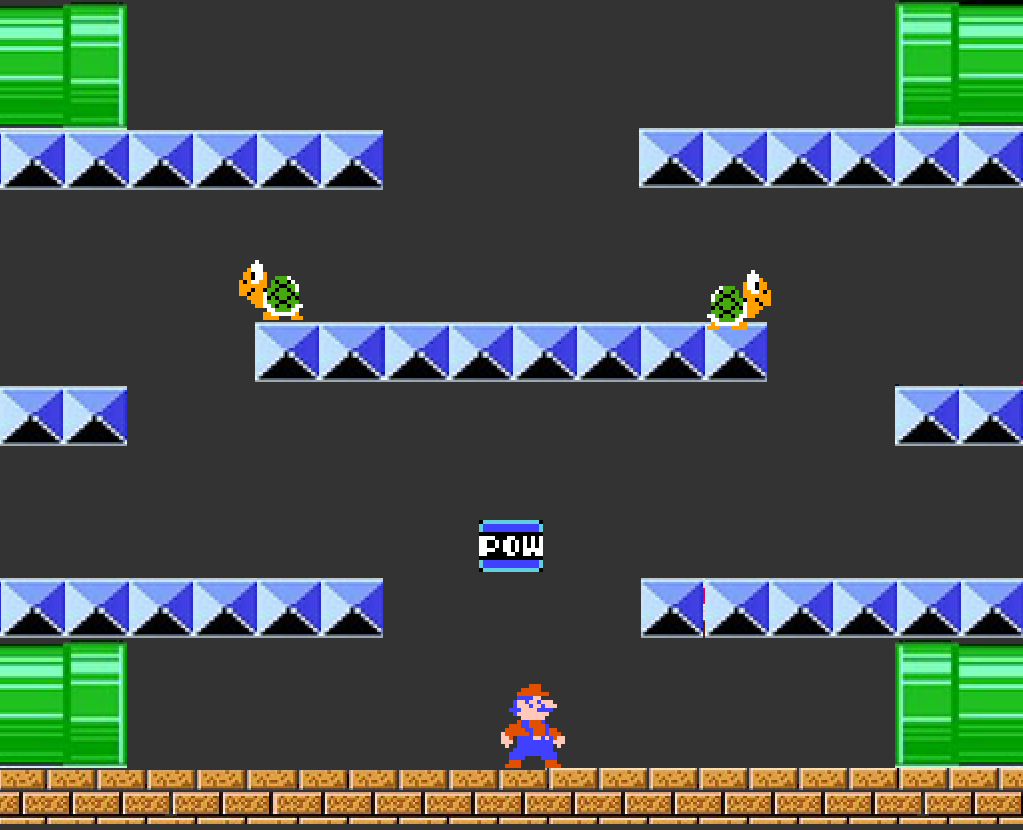
Left Click (On game level canvas) Draw Sprite (*Level editor*)

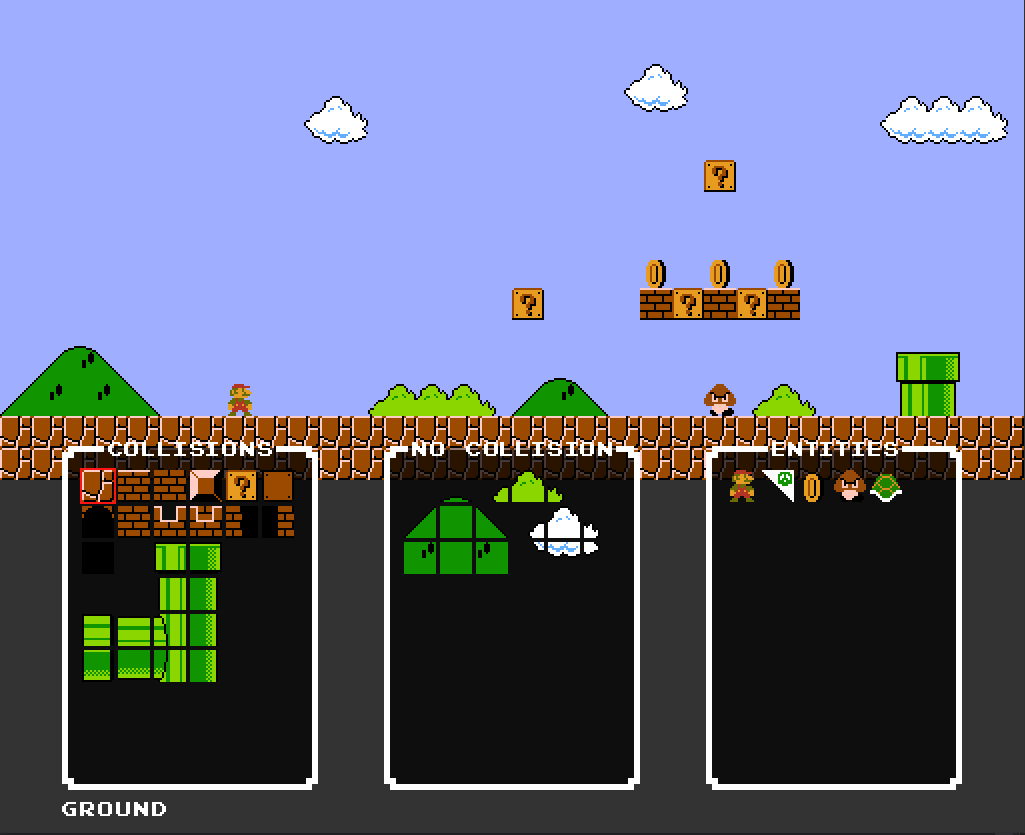
Right Click (On game level canvas) Erase Sprite (*Level editor*)

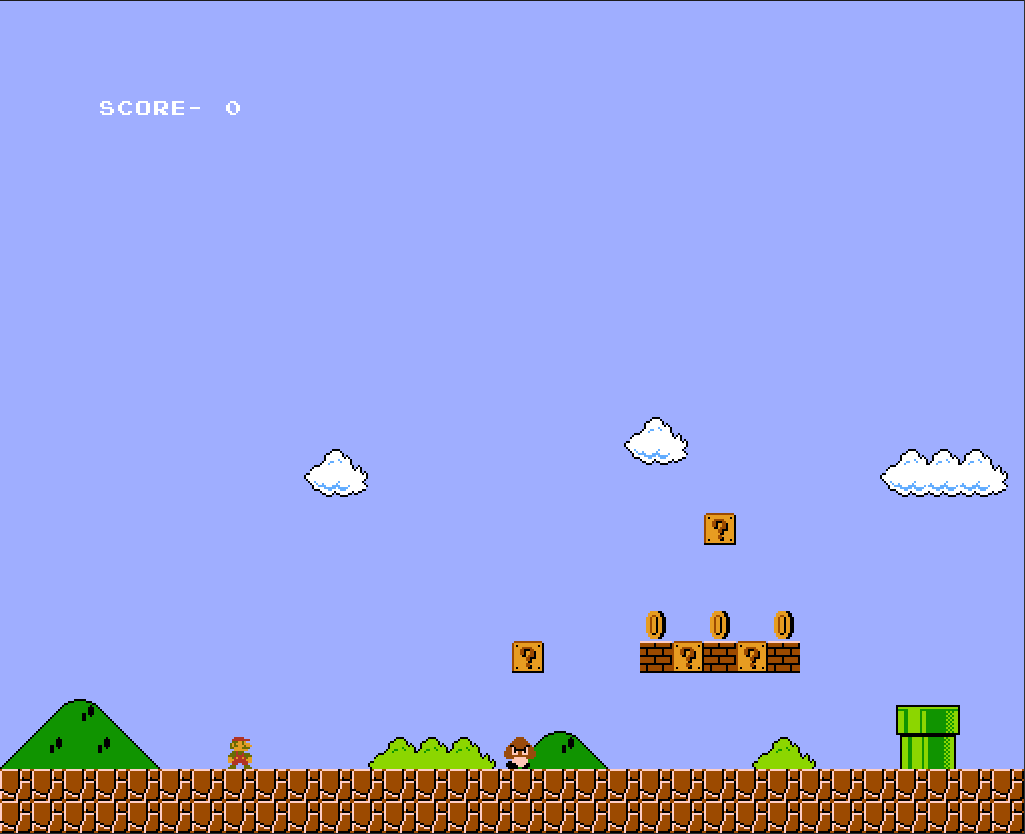
Middle click & drag Pan camera (*Level editor*)

**Screenshots**









**Test Plan**

**Bug 1**

*Entity getting stuck in floor after falling*

To fix this bug, I reset the entity’s Y position to the nearest multiple of 16 (size of tiles) when colliding down. The bug was happening because the entity would move into the ground between frames because of the gravity being applied. It would then register as colliding left and right when in the ground, so could not move. It is not noticeable for left or right collision because there is not a constant force being applied and it only checks for collision in the direction the player is walking, meaning if the player is in the wall to the right slightly, they will still be able to walk out to the left.

**Bug 2**

*Enemies updating when off screen*

I had a problem with enemies updating when not on the screen, which meant if there was a hole in the floor (like in the original SMB 1-1 level), Goombas and Koopas would just fall through the hole long before the player reached them. To fix this, every time I go to update an Entity, I check if it is within a certain distance of the player first. If it is within the distance of the ScreenWidth/2, it will be updated.

**Bug 3**

*Koopa jittering when walking*

This bug occurred because the Koopa is a different sized sprite to all of the other Entities. All other Entities are 16x16 in size, but the Koopa sprite is 16x24. This was related to the first bug in the report, where I had to round it to the nearest multiple of 16. It would be rounded to half a tile off the floor, fall, collide with the ground, and repeat. To fix this bug, I made the sprite 16x32 and have some clear space in the sprite above the Koopa. This has one drawback which is that the collision on the Koopa is slightly bigger than the Koopa, but it is not noticeable in game.

**Bug 4**

*Selecting sprite in level editor would also draw a sprite under the button*

I had a problem when first implementing UI, when selecting a different sprite which was above the level canvas, it would draw a sprite in that location as you clicked. To fix this I implemented a system which stopped other input if another button had been registered. So, when a UI button is clicked, it does not check for any other mouse input on that frame and does not draw the sprite.

**Bug 5**

*Standing near the edge of the map would crash the game*

This bug was because of an optimisation I made for collision. When each Entity checked every tile for collision, the game slowed down very fast. Instead I just access the surrounding tiles and check collision on them. Although when trying to directly access tiles (to select them for collision, as opposed to just looping through all of the existing tiles), the game would crash as it would try to access tiles which did not exist (outside of the map) due to a vector out of bounds error. To fix this I added a try-catch block around each tile access attempt, if it fails, the software does not crash and the collision for that (non-existent) tile just is not checked.