Caleb Landis

CSCI437

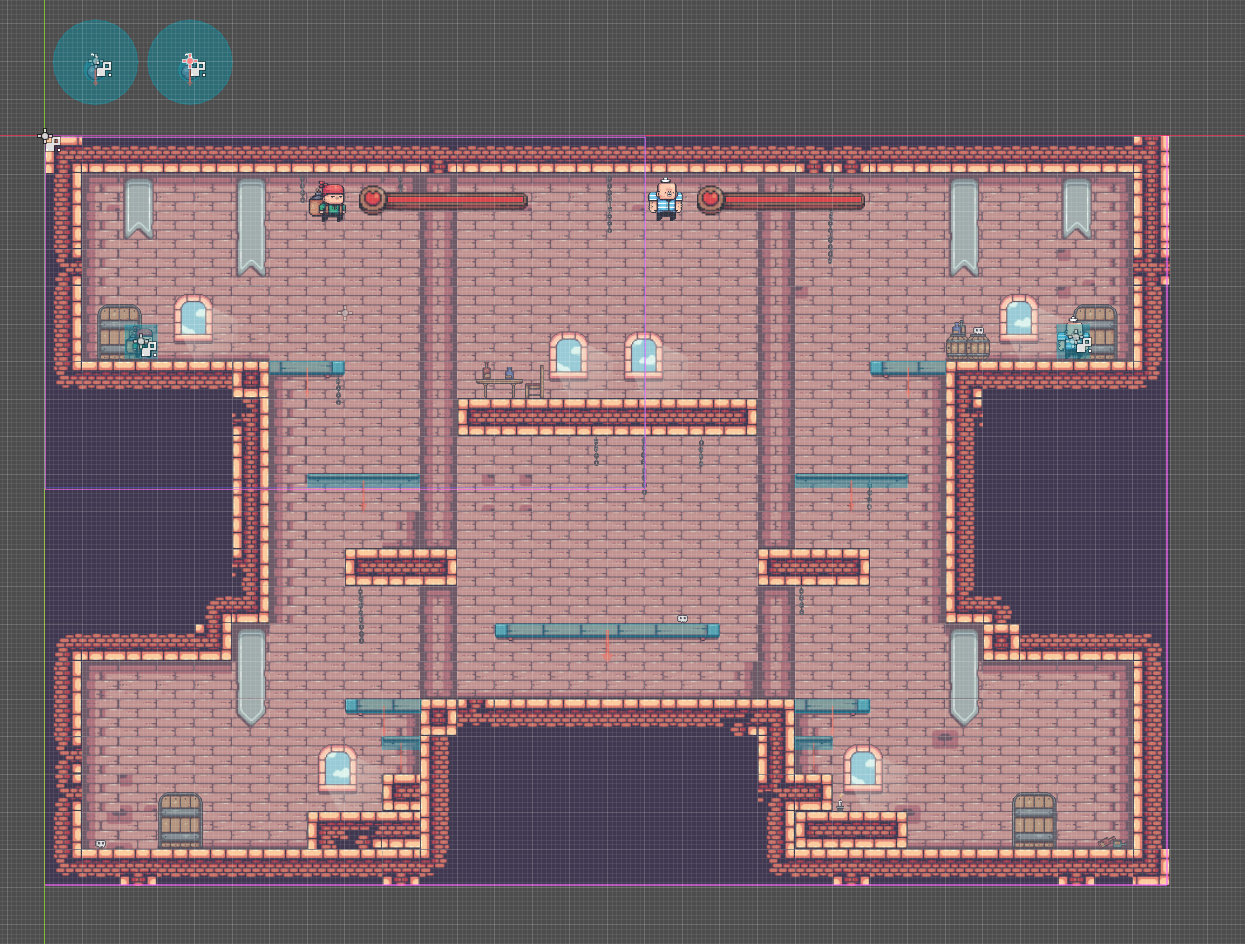
Final Project Documentation

**Bombs Away!**

**Game Design**

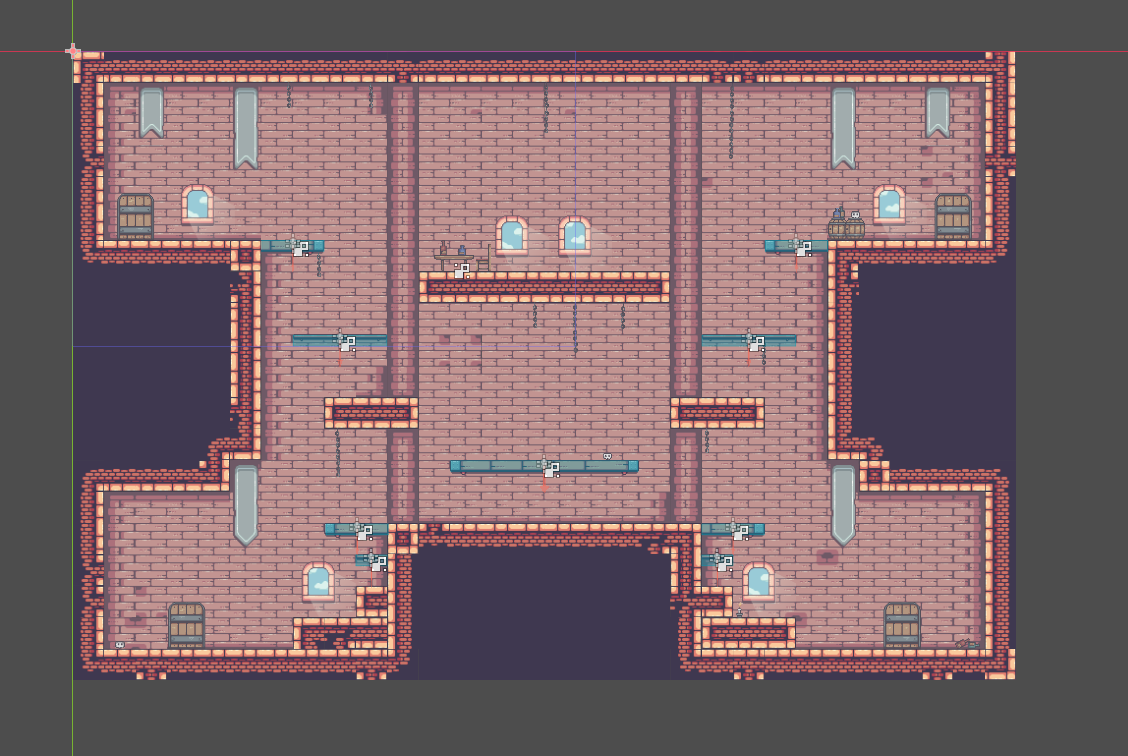
**Scenes**

**Castle\_Level.tscn**



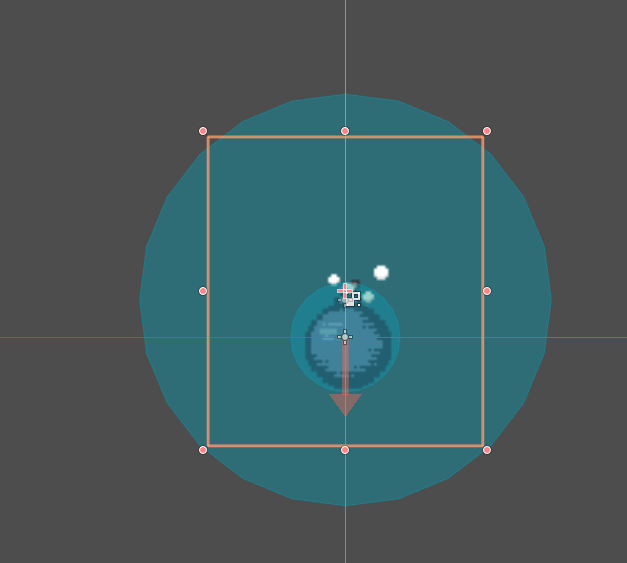
Castle\_Level.tscn is the main scene that encapsulates all the other scenes and sprites that make up the game. It holds the world tilemap scene, player animated sprites, animated bomb scenes, gui components, and the sounds.

**Castle\_World.tscn**



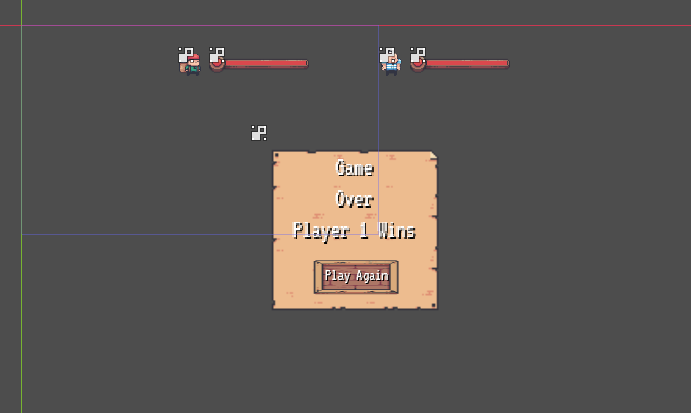
Castle\_World.tscn contains the tile map data for the world which is an imported map from the Tiled map editor. This scene also keeps track of all the collisions for the walls, floors, and platforms. I also added the background prop sprites that the player doesn’t interact with to this scene as well.

**Bomb.tscn**



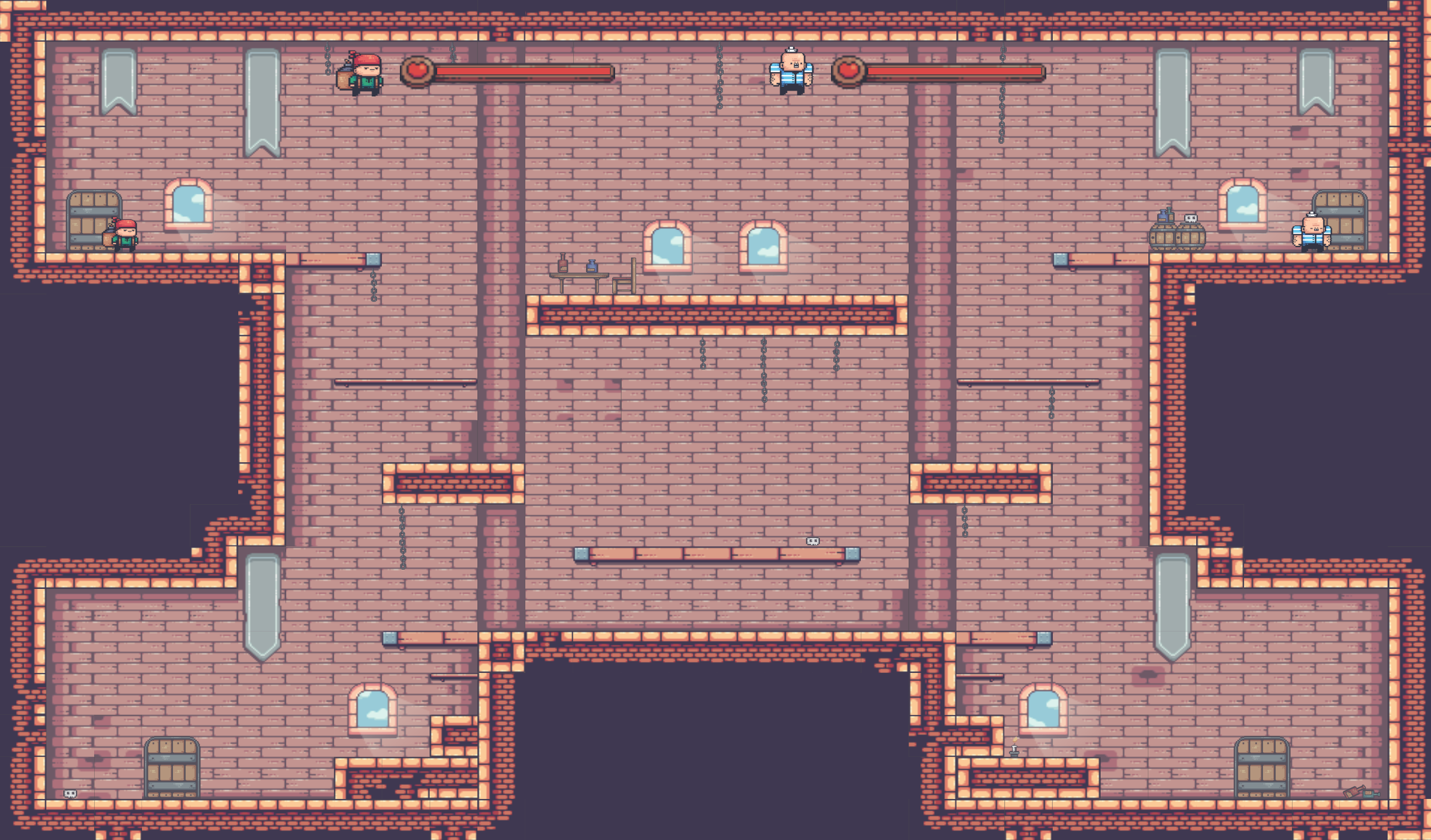
Bomb.tscn abstracts a lot of methods with how the player interacts with the bomb instances. The scene has an animated bomb sprite that changes animations from a lit bomb to an explosion once the bomb’s explode() function is called. The bomb also has two collision areas, one for detecting how to interact with the world, and the other for establishing a radius in which the opposing player will be damaged if within.

**PlayerHUD.tscn**

****

This scene houses the GUI components that helps indicate player state. The two meters at the top are health bars which fill less of the meter as the player loses health. The component in the middle is the game over menu which only becomes visible when one player reaches 0 health. Pressing Play Again will reset player, bomb, and gui positions, and the Game Over menu will once again become invisible.

**Sprite Interactions and State Transition**

****

At the start of the game, both players start equal distance from the wall at the top left and right corners separately. Both players’ bomb scene instances are loaded in, but they are set to not visible. Player sprites can collide physically with one another, but they cannot collide physically with the bombs. Once placed, the bombs can physically interact with the worlds’ collisions such as the floors and platforms. When the player sets a bomb, the bomb is set to visible, dropped from the players’ location, and falls to the floor.



When the enemy player is near the player’s bomb, the player can press the detonate button to have the bomb explode. A player cannot detonate a bomb without setting it first. If the enemy player is within the bomb’s explosion radius, the enemy player will take 10 damage. Each player starts at 100 health, and they are not damaged by their own bomb. When a player takes damage, their health bar will get lower.



If a player is damaged, the game will check the player’s health. If their health is 0, the Game Over menu becomes visible, and the game ends. If the player presses the Play Again button, the game will reset all states as mentioned earlier.



The state transitions throughout the game follow a simple path as shown below:

Players move around -> Player sets bomb -> Player detonates bomb -> Player health reaches 0

-> Game Over menu becomes visible and game pauses -> Player clicks Play Again

-> game resets

**Software Engineering Plan**

Since I am working alone on this project, my plan is to create everything in the order of which has the highest dependencies to run, which also follows the path of how the game plays and transitions states. The plan is as follows:

1. Create world map in Tiled map editor; import into godot.
2. Create collisions for walls, floors, and platforms.
3. Create Player sprites with animations, movement, and sounds all handled within their own script.
4. Create bomb scene so players can place, detonate, and be damaged by bombs.
5. Create health bars to indicate how much health each player has.
6. Design and implement game over menu to show who has won the game and reset all instances so game can be played again.

**How to Play**

To download the game, the source code can be found at my github repository for the project:

For some reason at the last second, godot did not want to properly build my game even though it had been working before me finishing the project. Therefore, for now, the game will have to be ran in debug mode through the godot game engine editor. Once the game window as loaded in, the fun automatically begins.

**Movement – Player 1 – Keyboard or Gamepad 1:**

Walk left -> “a” key or left on left joystick

Walk right -> “d” key or right on left joystick

Jump -> “w” or “a” on xbox and “x” on playstation

Set bomb -> “c” or “x” on xbox and “square” on playstation

Detonate bomb -> “v” or “b” on xbox and “circle” on playstation

**Movement – Player 2 – Keyboard or Gamepad 2:**

Walk left -> “left arrow” key or left on left joystick

Walk right -> “right arrow” key or right on left joystick

Jump -> “up arrow” or “a” on xbox and “x” on playstation

Set bomb -> “o” or “x” on xbox and “square” on playstation

Detonate bomb -> “p” or “b” on xbox and “circle” on playstation