Assignment 3 Report

Adam Atbi, Max Shankey

Respective Branches:
Adam - assign_3_adam
Max - Max’s_A3_Refactoring_Loops

1)
Entities being spawned:
Doors - COMMIT
Fires Commit
Coins - COMMIT
Doors, fire and coins were all being spawned in a way which meant that each one was spawned in at a specific coordinate and were done individually. This had a bad smell to it as it felt there must be an easier way to do this. For example the fires which are on pretty much every level made up over 40 lines of code, the way I simplified it down to made it less than ten. This was done by instead creating each instance of fire in a loop and using an array to store the positions, and what level they were on. The way doors and coins were spawned was also changed in order to simplify them. Refactoring how these instances were assigned their values reduced the number of lines in the game panel by a significant amount.

2)
Collision repeat code - COMMIT

In the Collision.java file there were 5 instances of repeated code determining the distance between 2 points using the pythagorean theorem. The repeated code is bad for readability and is not immediately clear without close examination. I created a function to return the distance between two points using the same formula which I can clearly call in place of the formula itself.

3)
Re factored collectibles into a super class to remove dead code and reduce repeated variables - COMMIT
Every single collectible had X and Y cords associated with every instance of them, instead of adding a private variable to each one of these things and making some kind of distinction. I decided to add a super class which all collectibles would inherit these things from. This removed repeated variables as well as removing some dead code in the process.

4)
Grass tile name - COMMIT

Simple fix of a confusing variable name. The original game aesthetic planned to have a brighter look and green grass as the base playing surface. When we decided on the darker style we changed the asset to a dark blue square but neglected to change the variable name from grass to a more appropriate one.
To rectify this I changed the name from grass to ground which is a more accurate description.

5)
Only draw assets in game jframe boundaries and other optimizations - COMMIT

As in any game one of the most computationally taxing aspects is graphics rendering. The method of level rendering we settled on is for all of the levels to exist next to each other horizontally at all times. By shifting them left and right we can reveal the appropriate window when the level changes.
One large inefficiency is that the original system would draw the asset to the window irrelevant of if it is visible in the window. By adding a check that allows only visible tiles to be drawn, we cut down on the number of assets to be displayed by a factor of 7.
Another efficiency optimization was to make a call to gamePanel once in the higher createWholeMap method and pass it into the lower draw method instead of calling repeatedly. In the old system each screen refresh would require almost 5500 calls to GamePanels getCurrentWindow, 3 for each asset.