Steven Eisemann

10/17/18

Chip's Challenge Reflection

The most significant change to my design from the last submission is that I moved the singleton pattern out of the Level class, in fact this class was removed completely, and made the GameGrid class the singleton. In addition to this, I utilized the observer pattern for keys, circuits, and doors, and the strategy pattern to generalize picking up keys and circuits. I added a Teleporter class to make it easier to add multiple teleporters in a level.

If I were to start from scratch, there is not too much I would change I think. If anything, I would clean up how my observers are observing strategies and try to split up the strategy for collectibles and the actual collectibles themselves. Observing strategies makes the UML diagram rather confusing I feel. I would also try to find a way to change how levels are loaded. Right now, I start the game at level one in the same way we began the Columbus game. When level 2 needs to start, I clear the game grid, remove the player ImageView, redraw the game grid, and then re-add the player ImageView. This all is happening in the runGame method and feels bad. It also makes it fairly un-scalable.

Besides those concerns, I don't think my design was too messy. Making the GameGrid a singleton made giving all the classes a reference to the game grid much easier than passing the same reference around everywhere. Using the strategy pattern to pick up items also helped simplify things as I was able to just "pick up" the object in the Chip class. Since I could not think of a good way to fit in observer, my game plays slightly differently than the original. When a key is picked up, all doors of that color open, since the doors are observing the matching key. I think this is okay though, since having a "yellow" key should allow all yellow doors to be opened.