Design Pattern Descriptions

Pattern Name: Observer Pattern	
Class Name	**Role in Pattern**
Key	Observable
Door	Observer
Purpose: The door observes the Key so that it knows when the door is able to be unlocked	

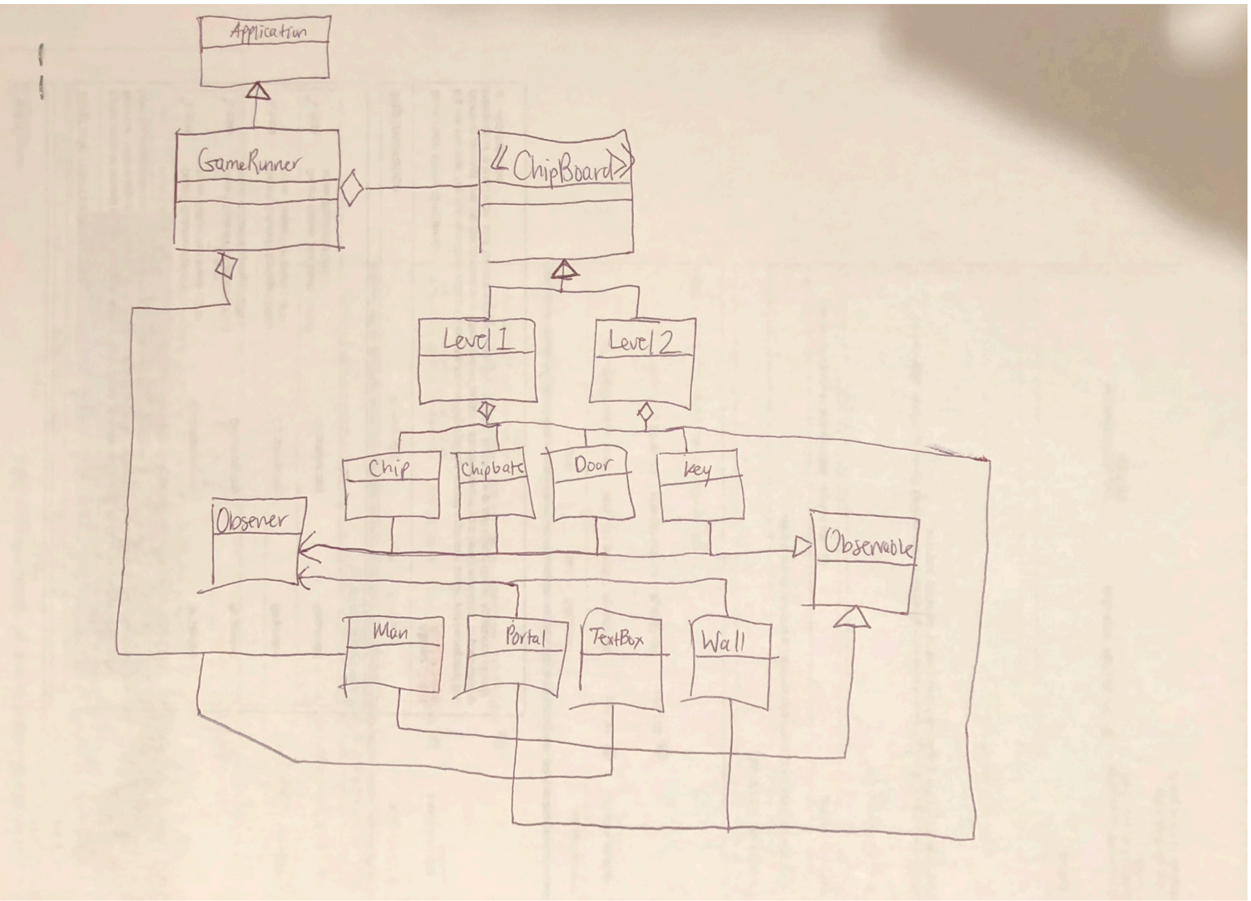
Pattern Name: Singleton Pattern	
Class Name	**Role in Pattern**
Portal	Singleton
Purpose: We only ever want one instance of the portal to exist, so we make it a singleton to ensure that this happens	

Pattern Name: Strategy Pattern	
Class Name	**Role in Pattern**
ChipBoard	Abstract Class
Level1	Implementation 1
Level2	Implementation 2
Purpose: Each layer of the game board is an implementation of ChipBoard… this allows us to customize the layout each time while having a similar foundation.	

Design Discussion

I am quite pleased with the way my game turned out, and it taught me a lot about different design patterns. I chose to use the observer method quite profusely because so much content within the game relied on the actions of other elements. Additionally, because this game contained multiple levels, I wanted to ensure that they all had a similar, yet customizable structure, so I chose to use the strategy pattern. As for implementation of the switch of levels, I wish I had more time to figure all this out, since I’m not sure how I would implement 3+ levels given my current design. Finally, an important aspect was making sure that singular instances of the elements existed (e.g. the man, portal, etc.) so using singleton design pattern ideas helped ensure that the elements on the board were the only instance and that the data they were producing was valid.

Updated UML



Changes to this diagram include the addition of observer patterns, additional functionality classes, and overall a better understanding of the program at hand.