Deadwood Design Patterns

We should probably start this report by saying that many of the design patterns we learned about in class weren't implemented in this assignment. The reason being that most of the code for the assignment was already written from Assignment 2 and could be re-used for Assignment 3 without making any modifications, and at the time of writing the Assignment 2 code, we hadn't yet learned about many of the design patterns. However, this doesn't mean we didn't use any design patterns, so here's the few that were implemented.

While implementing the MVC pattern, we made sure that there was no direct communication between the data class, like Player, and the view class, like GUIView (Which is the View class for both players and the board). Following the pattern, all communication and modification of attributes was done through a controller class, like PlayerController. Implementing this pattern was useful for us when we converted to the GUI version of the game.

For the Singleton pattern, we made several of our classes singletons because we only ever needed one instance of them. We did this for Bank, SetUp, and XMLParser.

For the Flyweight pattern, we only loaded in the various images for the game once, and then had all the GUI elements that used that image refer to the single instance of it. We can't actually confirm whether this approach was successful, because we don't know exactly how JavaFX's GUI element constructors work, whether they make shallow copies or entirely new ones for every instance of an object.