Design Patterns

Chosen Design Patterns

For our project we have chosen the following three design patterns, Strategy, Singleton and Factory.
This choice allows us to easily add and tweak features as the project advances while keeping a working game for our users. This also allows for highly customizable games with multiple presets. Like adding extra maps.

Strategy

The Strategy pattern gives us the most control in switching between different kinds of logic; which is why we use this for collisions.
This allows us to write multiple collision classes, each with their own way of colliding objects.

Singleton

The Singleton pattern is useful for running single instances of certain classes. This allows a global access point for our board class and the logic of the game and no trouble with different instances.

Factory

The Factory pattern is very versatile; which is why we use this to produce multiple objects, such as the game board, the flippers, the ball and other physics objects.
This gives us a straightforward way to safely create these game-critical objects.

Implementing our Design Patterns

Strategy pattern was implemented in our Board class. Here we also load your config file which allows the user to make an unique game for himself.

Singleton pattern was implemented in our Score class, this allows us to easily add score to the game from different classes in our game, we use this to add score through the collisions class and the board class.

Factory pattern was used to support the feature of having multiple balls. For this we use our BallFactory class to create multiple balls which allows us to have as many balls as we deem necessary.