Project 3
Team 16

Design Patterns

We will be making the arcade game Breakout. This single player game has the user moving a paddle to try and keep a ball from touching the floor. There are blocks on the ceiling that can be destroyed when the ball touches them. The objective of the game is to clear out all the blocks before the ball touches the ground. For this project, there are 3 core pieces that the game revolved around, ball, bricks, and paddle. The ball needs to be in constant motion and checking for contact with the other objects.

When deciding on the best route to take for completing this game, we decided to use 2 primary design patterns. The first being the Builder pattern. We did not use any classes therefore, each of the game functions had to be written on their own and assembled together. Each of our core pieces had their own set of priorities and had a function for each one. The paddle piece used an EventListener in order to check the player's mouse location. Then the drawPaddle function would be changing the paddles location and calling a hit function to check for the balls location. A separate hit function was used to check which bricks were not broken. An array would keep track of these and use drawBricks to put them on the board.

The other design pattern we used was Composite pattern. All of these functions and objects that have been previously mentioned can be traced back to the moveBall function in some way. The draw functions for the ball, bricks, and paddle are all called here in order for the changes to take place in real time with the ball's movements. Except for the start function and the brick array function, the rest are derived from the moveBall function. Are reasoning for this was due to this function being called every 5ms in order to simulate smooth ball movement. The Composite pattern helped us centralize where all of our functions were taking place and utilize the Builder pattern to keep the process simple.