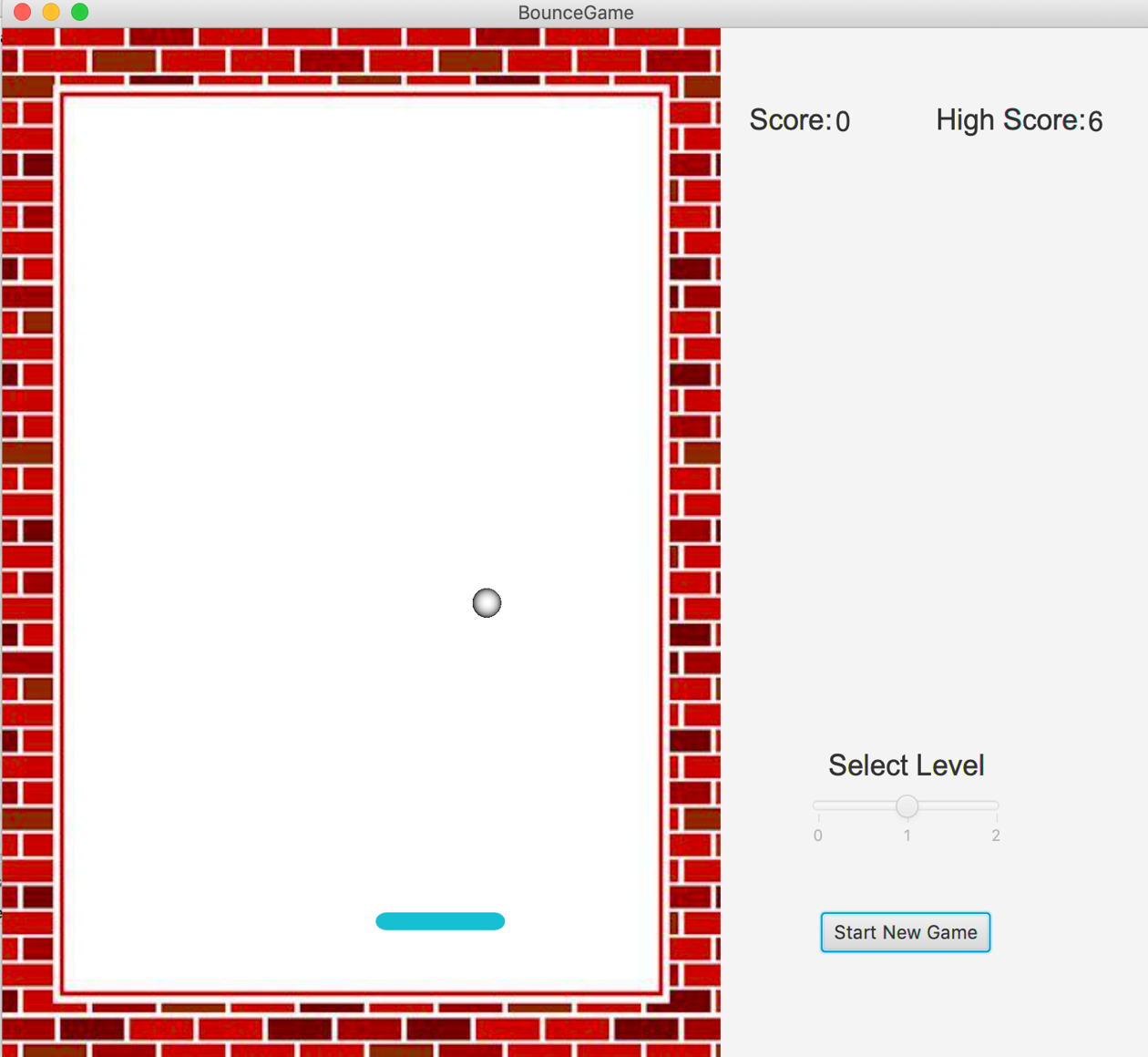
Bounce Game

**Design**:

I created a game using JavaFX. JavaFX is a Java library used to creating GUI Applications.

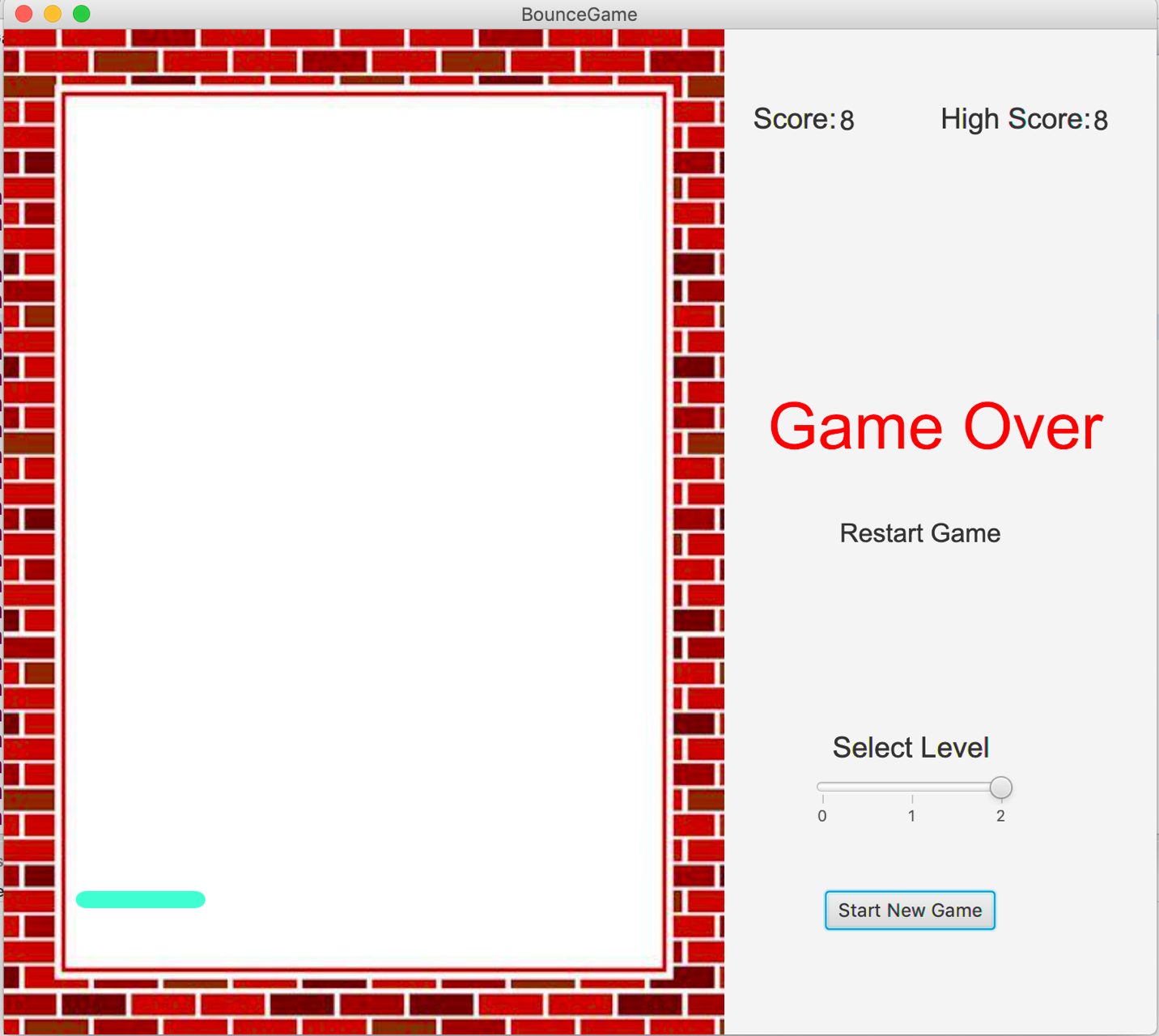
UI Design:



There are 2 main components in this game:

* Ball -> Created using Sphere Node available in JavaFX library.
* Paddle -> Created using Rectangle Node available in JavaFX library.

The remaining components in this game are Labels, Slider, Button and an Image of a Wall.



Game Play:

Ball starts from a random X position on top of the Pane and continues to travel downwards by bouncing of the walls until it hits the paddle on the bottom. Once it hits the paddle it bounces back starts travelling towards top of the Pane until it the top part of the wall. Game stops when player misses the ball with the paddle. Number of times the player can hit the ball with the paddle gives the player a score.

Game starts when player hits on Start New Game button. This resets the score and high score of the game. When starting a new game player can select level of the game where highest level denotes highest velocity of the ball.

Once the ball goes below the paddle the Game stops and player can restart the game. During restart only the score gets reset high score will be maintained. Restart will only take the velocity of the previous game and slider level will not reflect for the restart.

API Design:

BounceGame.java -> This class have the main method and all the game logic to perform various actions based on the user Input.

Paddle.java & Ball.java-> Each class represents the corresponding objects in the game and extend **Rectangle** and **Sphere** classes from JavaFX library respectively.

Components.java -> This class is used as a factory class for all the components in the game in order to reuse the code.

In BounceGame.java:

**start()** -> Sets the stage with all the required components in order to display in the scene.

**setRestartEvent(), setPaddleMovement(), setStartButtonEvent(), addSliderValueEvent()** -> These functions set event handling for restart, paddle movement, start and slider selection respectively.

**getNewTimeLine()** -> Creates a new timeline of events and returns so that start and stop actions can control the actions in the game.