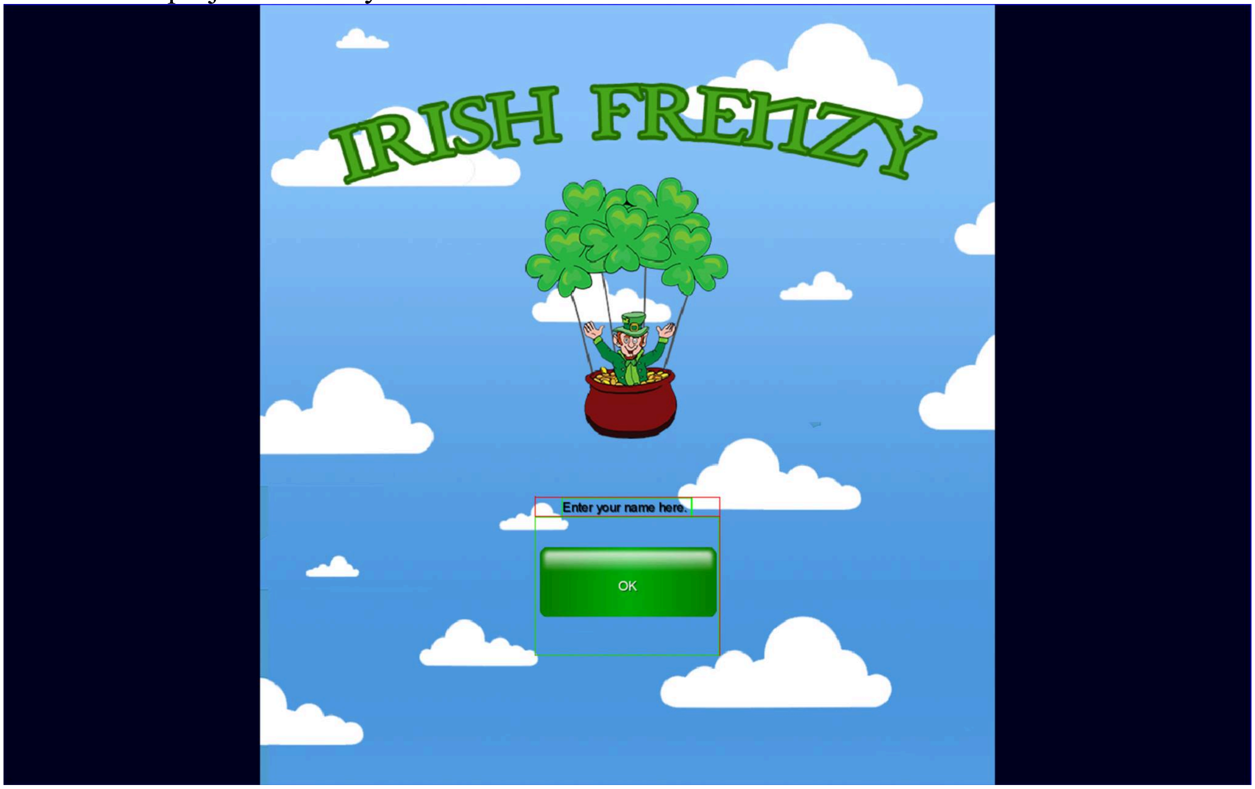
Irish Frenzy – User Manual and Documentation

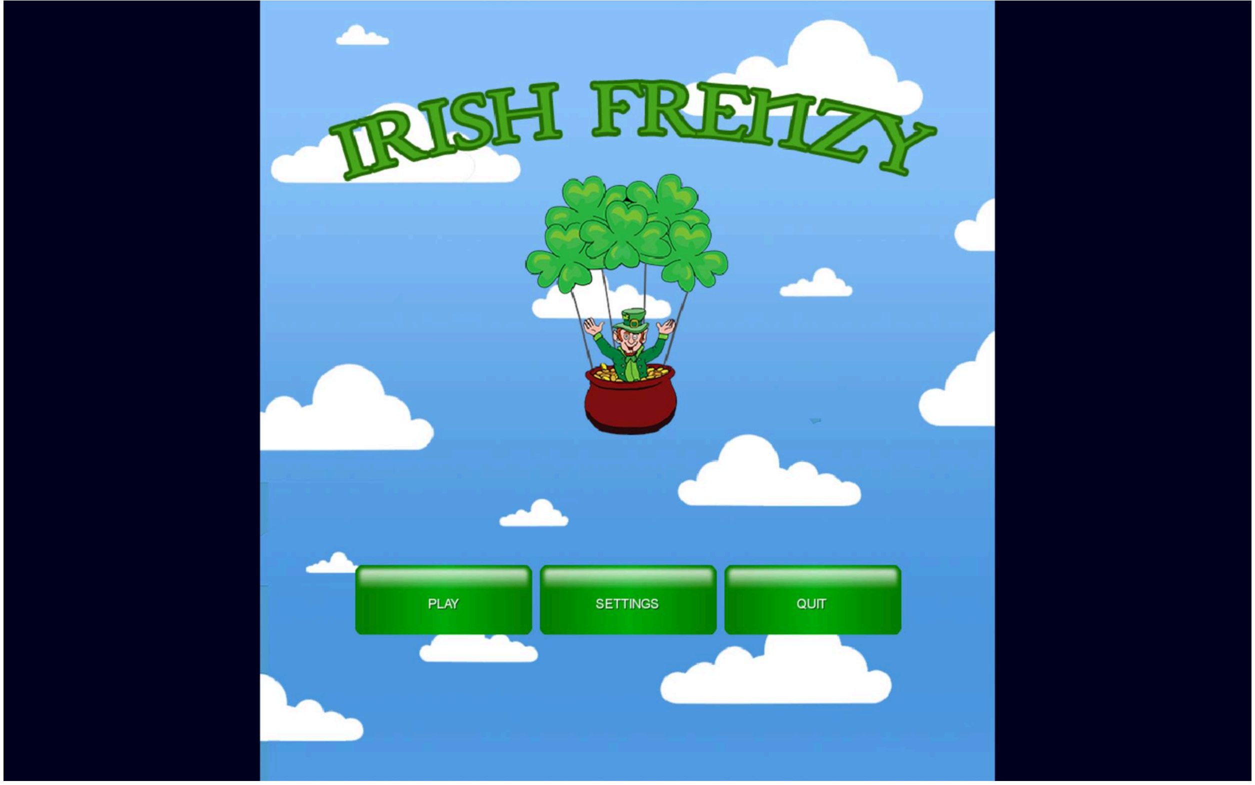
William Schiela, Anthony Janocko, Gabriel Duemichen, Adam Cole

User Manual

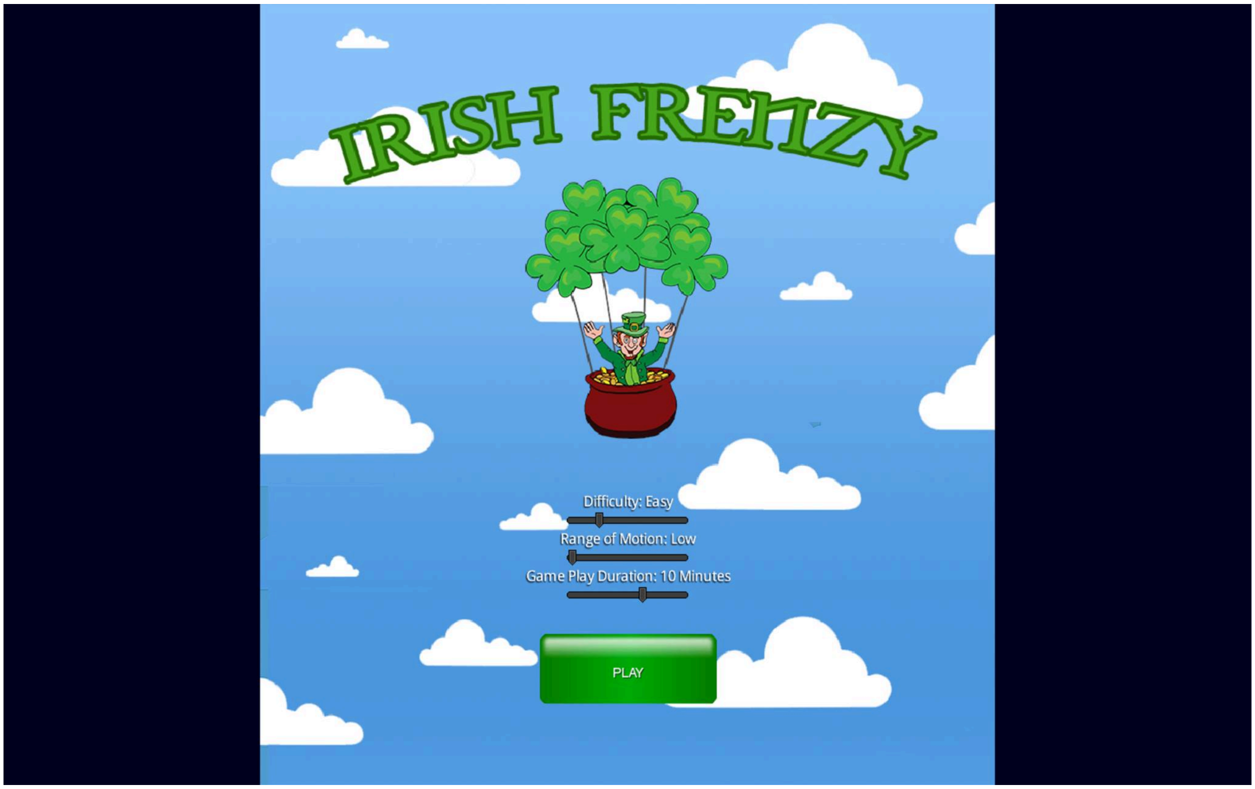
Login Screen – Upon starting the application, the first screen that appears is a login screen. Click on the text box to enter your name, and then click OK. The name you enter will be used along with the current timestamp to name the research data file – according to the naming convention "username-yyyy-mm-dd_hh-mm-ss.ssss.csv" – which is saved in the android/assets folder in the project directory.



Menu Screen – Following the Login Screen you will be taken to the main menu. The Menu Screen has 3 buttons: click "Play" to be taken to the Game Screen and start the game, "Quit" to close the research data file and quit the application, or "Settings" to be taken to the Setting Screen and customize the gameplay settings.



Setting Screen – The Setting Screen has slider bars to allow the user to calibrate the gameplay to their individual ability. Use the "Difficulty" slider to adjust the gameplay difficulty from Very Easy, Easy, Normal, Hard, and Very Hard. This affects the scrolling speed, frequency of obstacles, and duration for which some obstacles (like wind) appear. Adjust the "Range of Motion" slider from Low, Medium, and High to customize the gameplay to the user's individual range of motion limitations. The lower the Range of Motion, the more easily the character can move left and right. Use the "Duration" slider to adjust the gameplay duration from 1 minute to 15 minutes.



Difficulty: Easy

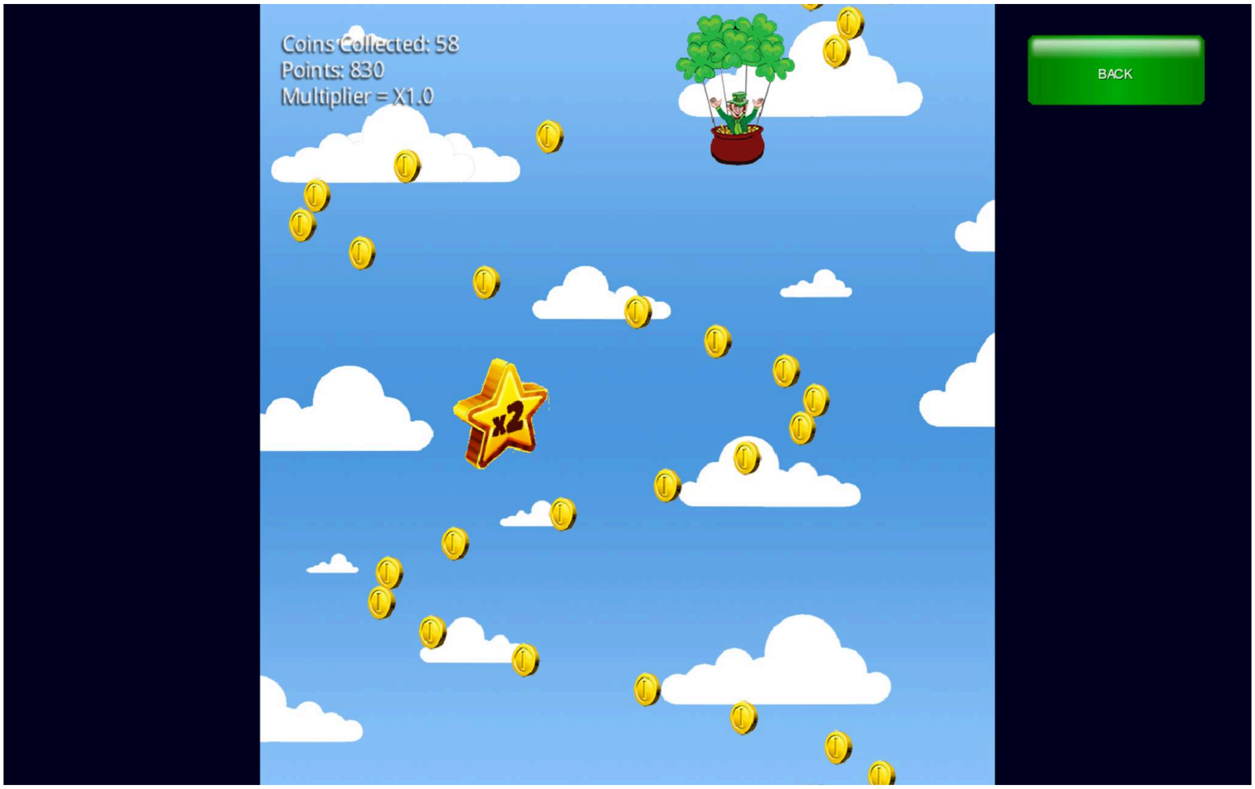
Range of Motion: Low

Game Play Duration: 10 Minutes

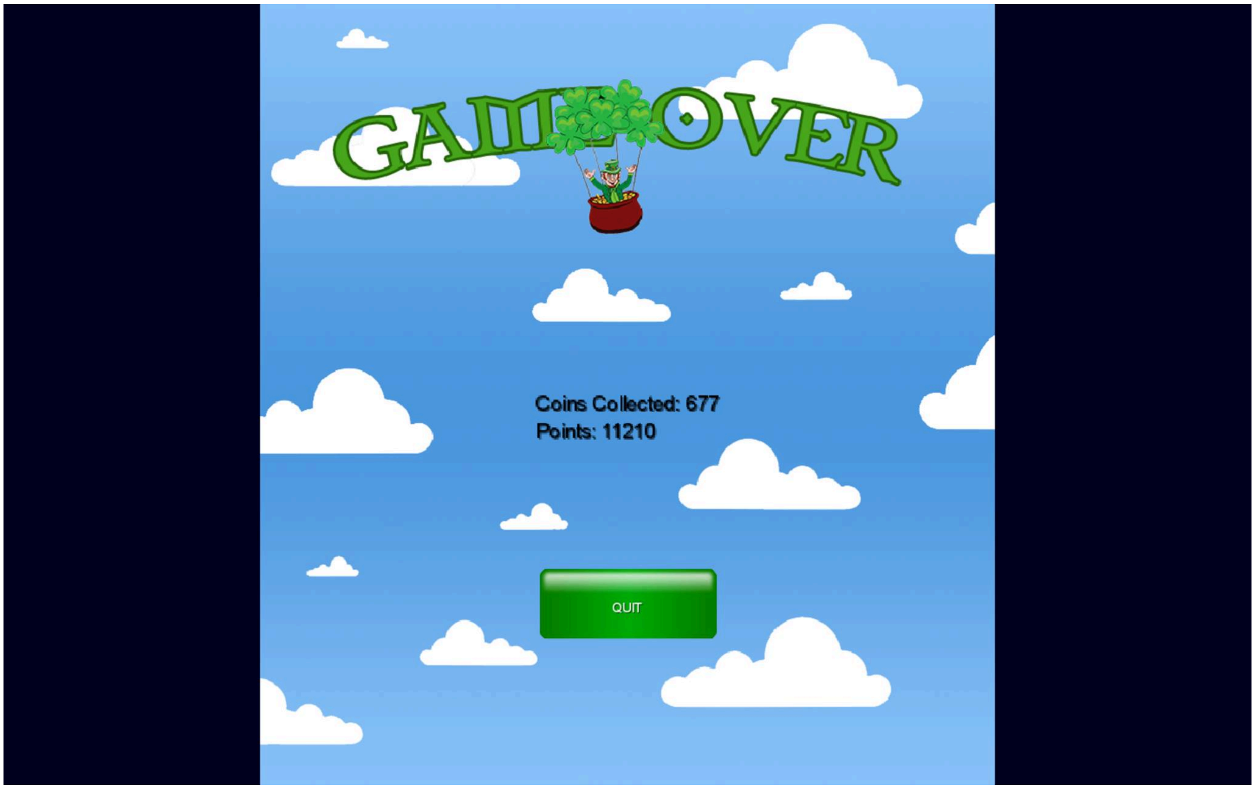
Game Screen – The Game Screen features the main character falling through the sky. Coins appear in a sinusoidal path with amplitude that varies randomly every half-period, while power-ups and obstacles appear at random positions with varying frequencies. The player can move the user left and right with the arrow keys to collect coins, gain power-ups, and avoid obstacles. The player's current score, number of coins collected, and current point multiplier is displayed in the top left corner of the screen. Click the Back button to pause the game and return to the Menu Screen.

Summary of power-ups/power-downs and obstacles:

- *2X Multiplier*: Doubles the point value of coins for a period of time
- *Rainbow*: Causes a pot of gold worth 50 coins to appear.
- *Poison Bottle*: Inverts the player's controls and halves the point value of coins for a period of time.
- *Cannonball*: Prevents player from collecting coins for a period of time.
- *Wind*: Pushes the character towards the side of the screen. This is meant to be implemented on the crank controller as an applied force in the direction of the wind.

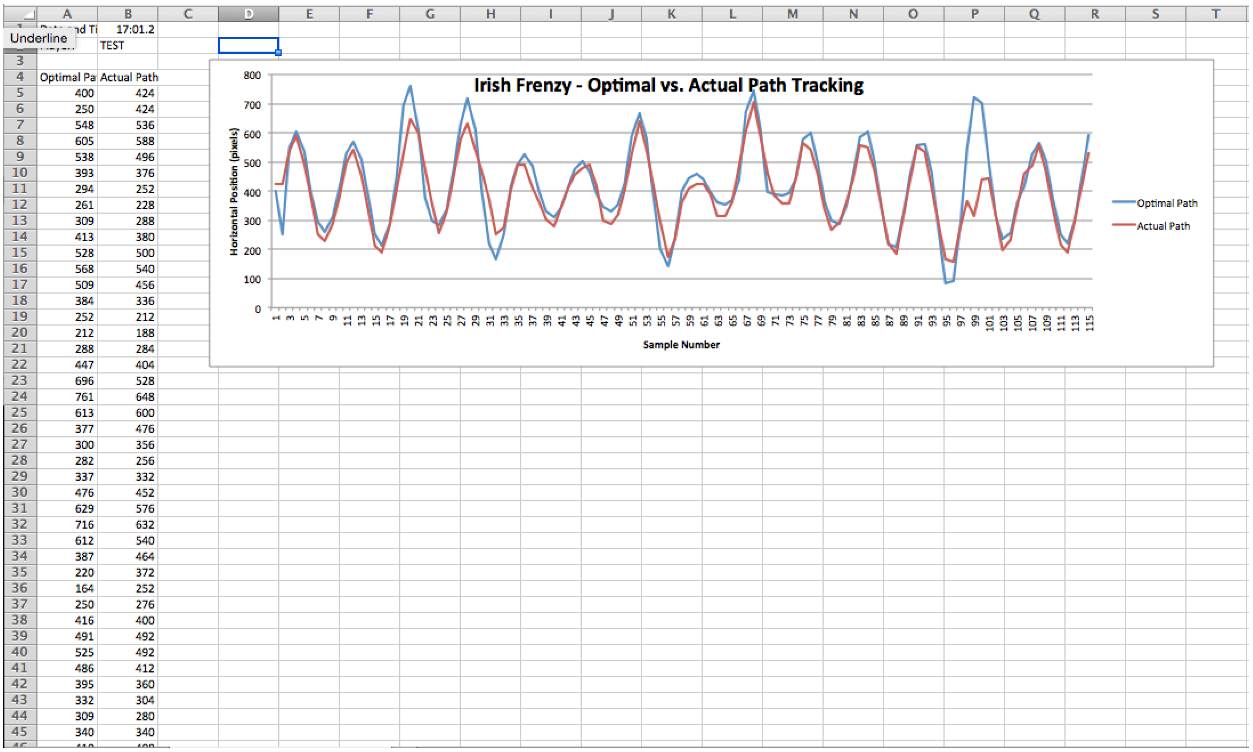

Coins Collected: 58
Points: 830
Multiplier = X1.0

Game End Screen – The Game End Screen displays the player’s total points and number of coins collected during the duration of the game. The back button allows the user to return to the Menu Screen.



Coins Collected: 677
Points: 11210

Output data – Upon quitting the game using the quit button, the game outputs the user performance data to the android/assets folder. This data file is named using the username of the patient and the date. The data file includes username, date, the actual path of the user and the optimal path. Here is an example of the data file and a quick analysis of the performance. As you can tell from the graph, this user was very competent and was able to follow the optimal path for most of the gameplay:



	A	B
1	Date and Ti	17:01.2
2		TEST
3		
4	Optimal Pa	Actual Path
5	400	424
6	250	424
7	548	536
8	605	588
9	538	496
10	393	376
11	294	252
12	261	228
13	309	288
14	413	380
15	528	500
16	568	540
17	509	456
18	384	336
19	252	212
20	212	188
21	288	284
22	447	404
23	696	528
24	761	648
25	613	600
26	377	476
27	300	356
28	282	256
29	337	332
30	476	452
31	629	576
32	716	632
33	612	540
34	387	464
35	220	372
36	164	252
37	250	276
38	416	400
39	491	492
40	525	492
41	486	412
42	395	360
43	332	304
44	309	280
45	340	340

Technical Documentation

`GameState.java` – This class is the first object created when the game is executed. It handles the instantiation of all objects directly related to the game state, such as the Character, OptimalPath, CoinPath, ObstaclePath, PowerPath, Wind, and DataFile. It also defines the default settings for the game, such as difficulty, scroll speed, range of motion, and duration, and controls properties of the game's appearance, such as the aspect ratio. Once the game's state objects have been instantiated, it sets the current screen to the Login Screen, and the game window appears.

Dependencies:

- `Character.java`: This class handles the current state and appearance of the character. In particular, this class keeps track of the character's current position on screen and updates that position according to key-presses made by the user. The character's `update()` method calls `shiftCharacter()` to move the character left or right, depending on the last key-press and the poisoned-state of the character. The Character class also has a PowerContainer to hold the character's currently-applied power-ups.
- `OptimalPath.java`*:* This class computes and randomizes the optimal path along which coins spawn and creates reference points along the optimal path with which to compare the character's actual position. For each reference point, the OptimalPath writes the optimal position and the character's actual position to the .csv DataFile for research purposes.
- `CoinPath.java`*:* This class handles the spawning and updating of coins and pots of gold along the optimal path.
- `ObstaclePath.java`*:* This class handles the spawning and updating of obstacles at random horizontal positions on the screen.
  Related classes:
  - `ObstacleFactory.java` – creates obstacles
  - `Obstacle.java` – superclass for all obstacles' shared functionality
  - `Cannonball.java` – represents the cannonball obstacle
- `PowerPath.java`*:* This class handles the spawning and updating of power-ups and power-downs.
  Related classes:
  - `PowerFactory.java` – creates power-ups/power-downs
  - `Power.java` – superclass for all powers' shared functionality
  - `PoisonBottle.java` – represents the poison bottle power-down
  - `Times2Multiplier.java` – represents the x2 multiplier power-up
  - `Rainbow.java` – represents the rainbow power-up
- `Wind.java`*:* This class handles the appearance and effects of wind clouds as they appear periodically throughout gameplay.
- `DataFile.java`*:* This class handles opening, writing data to, and closing the .csv research data file, which is saved to the android/assets folder once the user quits the game.

`LoginScreen.java` – This is the class that creates and displays the Login Screen and gets the username entered by the user, storing it in a static variable to allow access by GameState when it instantiates the DataFile.

`MainMenu.java` – This class creates and displays the Menu Screen, including buttons to proceed to the Game Screen and play the game, proceed to the Setting Screen and customize the gameplay settings, or quit the game and exit the application.

`SettingScreen.java` – This class creates and displays the Setting Screen, including sliders to adjust the game difficulty, gameplay duration, and user's range of motion. When a slider's state is changed, this class updates the corresponding game state variables in GameState.

`GameScreen.java` – This is the class containing the primary game loop. Along with GameState, it handles all the game logic during gameplay, updating GameState and rendering the relevant game objects onscreen.

`GameEndScreen.java` – This is the class that handles displaying the user's final score and number of coins collected at the end of gameplay.