Balance it!

Authors: Rohan Parikh, Akshat Adsule
Revision: 5/6/22

Introduction:
Our program is a game in which users attempt to balance various objects through their phones. Our game will be level-based with increasing difficulty and skill required. To increase the difficulty, we would have more storage spaces and more items to sort. Our program is aimed at people who have injuries or conditions in their hands that impede their fine motor skills. Through our game, users can practice their fine motor skills and regain their abilities. The rules are to categorize the various items into the same-colored bins. If an item is dropped or miscategorized, the user has lost and the game will reset. Once a user has successfully categorized all the items, they will win and move on to a more difficult level. Tilting the phone would change the surface angle which would move the blocks from side to side- trying to get it into the box. The final level will have the most storage boxes and most things to store and balance. The main target audience for our program is people with arthritis and this program will aid them in improving their hand-eye coordination along with other traits.

Instructions:

1) Find the runnable Jar file in the dist folder
   a) Alternatively, you may build the jar yourself by running `./gradlew assemble`
2) Run the jar
   a) This should just work by double-clicking the jar file, but you may need to run the `java -jar` command
   b) Alternatively, you can run `./gradlew run` to run the project
3) Profit
   a) Once the program launches, you should be greeted with a QR code. Scan the code with your phone, and you should see a level select screen. **For best results, hold your phone sideways from this point**. From here you can select a level and play it. The objective of the game is to ‘bounce’ the same colored circles into the same colored boxes. If you drop one or put a circle in the wrong box, the level will reset. You can control the angle of the balance beam by rotating your phone horizontally.
      i) If you have an iPhone, you may need to use Chrome to open the URL

Features List (THE ONLY SECTION THAT CANNOT CHANGE LATER):
Must-have Features:
- Mobile-based controller. Controlling the game through a physical device is an integral part of our game. With our goal of helping those with limited mobility, a mobile-based controller is expressly crucial for such a goal. There will be an option to play specifically on the computer rather than having a phone controller.

- Networking. It would be much easier for the user if the process of playing the game was as simple as possible. Thus, having a network connection between the device and the computer is important as it is the most seamless way of handling inter-device communication (compared to something like having a USB connection)
- Physics. Our game is based on balancing which is fundamentally physics-based. Being able to do this with accuracy and precision will make our game intuitive and grounded.
- Levels. We need to make several levels in order for the players to have options. Also, the same level over again could get repetitive and boring- even if this is designed to aid people with disabilities- so it would be a must-to-have level.
- Being able to skip. If a level is too difficult for someone, they should have the ability to skip the current level and move on to the next.

Want-to-have Features:
- Being able to go back to previous levels. If a user would like to practice a certain skill again at a level, they should be able to do so.
- Stars with time- 3 stars for under a certain time, lower stars if time was higher. Assigning star values will motivate the user to complete the level in a faster time and improve their fine motor skills.
- Save feature. A save feature would be necessary if the players want to have a way to get back to the level they were at previously and not restart the whole game every time.
- Handling networking errors effectively. Networking errors are bound to happen (such as client disconnects), and our game ideally should be able to handle such errors gracefully.
- Fancy UI. Ideally, our UI should be cohesive and easy to use.

Stretch Features:
- Multiplayer aspect with competition-based levels/leaderboards. This would be a very nice feature to have (and is entirely possible with Firebase), it seems out of the timeframe we have for this project.
- 3D graphics and physics. While this is possible with Processing, we do not believe that we currently have the scope nor the time to fully develop 3D physics and handle that in processing.
- Level builder. The level building would introduce a whole new UI and flow we would have to design and implement and is thus out of the time frame for this project.

<u>Class List (basic for now, will expand once we start coding)</u>:
- Main - Starts the program
- Drawing surface- for the screen attributes and some methods including the draw method, mouse pressed.
- LevelSuperclass - Contains data about the current level
- Level 1- Contains data and attributes for level 1
- Level 2- Contains data and attributes for level 2

- Level 3- Contains data and attributes for level 3
- Level 4- Contains data and attributes for level 4
- Blocks class- Contains information about the actual objects that the user has to move to the balancing mechanism
- BalanceBeam - Contains data for a balance beam on the current level which the user control by tilting their phone
- GamePiece- Contains data for any game piece on the level
- Generator- Represents the ball generator. The generator drops down 2 different colored balls, red and blue which the user needs to put in the respective boxes
- Hopper- Contains data for the storage system for the balls
- Physics Circle- Represents the circle with physics, has attributes for a circle that match up with physics realistically
- Physics Constants- Contains data for base physics values including friction, gravity, and damping
- Physics Line- Represents the line with physics, has attributes for a line that matches up with physics realistically
- Physics Rectangle- Represents the line with physics, has attributes for a rectangle that matches up with physics realistically
- Physics Shape- Represents a general shape with physics, has attributes for a rectangle that matches up with physics realistically
- GameContext- Represents the current state of the game and is shared between several other classes
- GameSessionListener- Listens to the current session and updates the GameContext with the new data using the data from the user's phone
- Session- Represents the game session between the computer and the client
- SessionSnapshot- Represents a snapshot of data passed between the client and the app
- MainScreen- Contains data and represents the main title menu with 4 buttons that lead to their respective level number

Credits:

Developers

- Akshat: Networking, level design, and game code.
- Rohan: Physics, game code, and screens.

Libraries

- Processing
- Firebase
- QRGen
- Java Standard Library