Nafisa Ahmed

Lubna Lakhwala

Alison Price

Skanda Rao

Myia Samuels

Jason Wong

28 Jan 2016

“The Adventures of Red Ball” Final Report

The main goal of our game is for a player to successfully complete three minigames as the Red Ball character. Our three minigames are the maze, platform game, and battleship. We, as a team, were able to complete this code.

One of the first things we had to do was plan the execution of the code. The lead programmers mainly took on that job along with the QA tester. The graphics designer designed the Red Ball, who is our main character, and the other backgrounds to the games. The UI and the project manager started on the flowcharts.

When the player first opens the game, a beautiful start screen will appear, giving them three options. The player can immediately start the game, read the instructions, or view the credits. The instructions are very basic, not including any controls as they vary from game to game. From the instructions and the credits, they have the option of returning to the start screen.

Once the game is started, the player is given instructions specific to the maze game such as the controls, as well as the number of lives for all three minigames (15). When they press start again, the maze appears. The ball appears in the center of the maze, and the player has to rotate the maze itself (which is constantly shifting down) to guide the ball through the maze. If the ball hits a wall, the character loses one life and is returned to the center.

If at any point in the three games, the player loses all 15 lives, they are shown the game over screen. The screen offers them the ability to return to the start screen so that they can try playing the whole game again.

However, if they are able to defy all odds and finish the maze, they will go to the instructions to the platform game. This will bring them to the game itself. In this game, they are required to move the ball to the door on the right by jumping along the platforms using the arrow keys. If they try to jump through a platform from directly underneath, they are reflected back down. Touching the bottom of the screen causes the ball to lose one life and returns it to the top left, where it originally begins. The door can be entered by pressing down when on its platform.

After they go through the door, the player will see the instructions for the final mini game: the battleship game. Then they will be faced with the actual game. The objective here is to guess the alignment of the battleships randomly generated by the computed within a certain number of moves (50). If they do not do so, they lose a life, and the computer respawns the battleships. The game is complete after they finish this level, presenting them with the end screen to congratulate them, to finally inform them of the story (defeating the evil rectangle and saving the world), and to offer them the option of starting again. It also notifies the player of the table tennis bonus game.

The code to make this game is mostly made up of classes.

The main problem during coding was the merge conflicts, mainly because Jason was the only member with experience with coding in groups after Google Code-in.

The table tennis bonus game created by Skanda was not originally included with the other games as it is a 3D game. Instead, it was included as a folder within the game’s folder, but we have managed to convert it into a separate class and add all the images with the other games. Once the player has completed the battleship game, they will be sent into the start screen of the bonus table tennis game. The screen will have a difficulty level selected on top (by default normal) and the start screen on the bottom. The difficulty changes the speed of the ball and the size of the player’s paddle. On starting the game, the player is presented with the instructions. A timer counts down until the game, giving them 30 seconds to read. They also have the option of skipping to the game with a skip button at the bottom right.

In the game, there is a long green table set up. There is no net in this version, and the paddles can only move along the plane perpendicular to the board along the edges. The computer is on the far side with a dark red paddle, and the player has a yellow paddle left unfilled to allow them to see the table through it. The red ball begins with the opponent in every round, and a timer counts down three seconds. When the ball is released, it bounces along the table, deflected by the paddles. If it falls off of the board, the paddle opposite of the side it last touched gains a point (for example, if it hits the player’s side and falls off, the computer gets a point).

The player’s paddle is controlled with the mouse while the computer moves its paddle towards the ball when the ball approaches. The ball deflects at wider angles when the player hits it with the left or right of the paddle. The player’s controls seem to inherently make it more difficult for the player. However, there are techniques to winning, such as making use of the wild angles to make the ball go from one corner to the diagonal opposite and dodging the ball if it is heading out.

Points appear for the player on the left as yellow balls, and for the computer on the right as dark red balls. The first to ten wins. If the player wins or loses, they are presented with a screen that tells them so, then presents a button to redirect them to the start screen.

The inspiration for the game was the initial creation of a mustachioed red ball as an adventurous character. The theme of having “shape” characters was continued with the idea of having a rectangle be the red ball’s enemy, but in the end this character did not play much of a role in the game.  Due to the larger size of this team, it was decided that the game would be comprised of several mini-games so everyone could code a lot.

We are proud of many aspects of our code.  We are mainly proud of the fact that we managed to make a complex, functioning code.  The majority of our group members have little coding experience, so we were unsure of our ability to make a complex code.  However, we learned new coding techniques and skills as we went along, and came up with a great final product.

We are, however, especially proud of a few things.  First, we are proud of ourselves for learning how to handle merge conflicts properly and efficiently.  Since we were all simultaneously working on the same code, we had to resolve several merge conflicts throughout this project.  Having to repeat this process several times made us better and better at fixing the conflicts, especially with the help of Jason who knows GitHub well.

We are also especially proud of the fact that we encountered a very small amount of bugs during this project.  Any problems we had were quickly and efficiently solved.  This was definitely something to be proud of, for it meant that our code had been put together well.

The only external sources we used were pictures for either our characters or backgrounds.  We used a maze image for our maze game, and Myia created the platform game in Photoshop.  We also used pictures like eyes and mustaches to create the characters in our game. Overall, we would all agree that this code was a success.