2D Mario Kart Madness

Manuel Guerra

Version: 1.0

Overview

The goal of this goofy kart racer game is to rack up as many points possible and make it to the other side of the screen in the fastest time possible. However, the drivers behind you and in front of you will try to prevent you from doing so.

Game Description

Objective – Get to the other side of the screen with the most points as possible and fastest time possible.

Gameplay – You're the main driver (Mario) trying to make it to the other side of the screen but the other drivers (Bowser and Toad) are trying to prevent that. You start off the race at the left side of the screen (with Toad) and you have to reach Bowser to finish the game. However, Bowser and Toad will be throwing items at you which will slow down your progress from reaching the other side. In order to dodge their attacks, you can either change lanes or block their items with your item (either in front or behind you) and you get points for successfully blocking them. If you get hit by an item Mario gets pushed back (how much depends on difficulty). If you use a mushroom then Mario thrusts forward temporarily. You can also throw your items forward only and if you hit Bowser you get points. If you use an item behind you then it is automatically discarded because Mario can't throw items backwards (he needs to focus on the road). The main driver receives an item every 7 seconds if they don't have one. When you finally reach Bowser then you've beat the game which will then display your score and time.

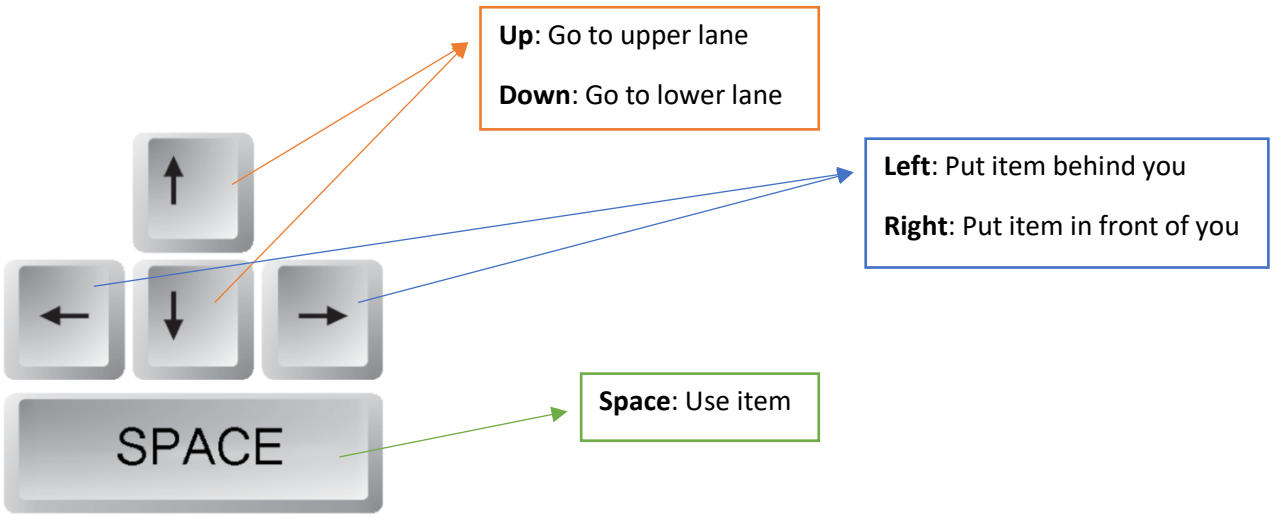
Controls – UP, DOWN, LEFT, RIGHT, SPACE

- UP – move up to the upper lane of the track
- DOWN – move down to the lower lane of the track
- LEFT – move shell item (if you have one) behind you
- RIGHT – move shell item (if you have one) in front of you
- SPACE – uses item (discards shell if it's behind you)

Assets Used

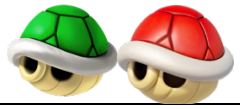
- Images
  - All item PNGs, character PNGs, background JPG, and controls PNG were all found on Google Images
- Sound files
  - battleMode.mp3 at https://downloads.khinsider.com/game-soundtracks/album/super-mario-kart-original-soundtrack
  - itemBox.mp3 at https://www.youtube.com/watch?v=a4eJjmAxUGA
  - marioNo.wav at https://themushroomkingdom.net/media/mk64/wav
  - marioYes.wav at https://themushroomkingdom.net/media/mk64/wav
  - welcomeN64.wav at https://themushroomkingdom.net/media/mk64/wav

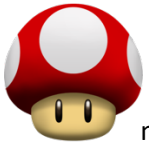
User Menu



You can only throw shells forward, if you use a shell behind you then it will be discarded!

Item	Attack with	Defend against
 green shell	+25 points	N/A
 red shell	+50 points	+25 points
 blue shell	+100 points	+50 points

Item	Can defend against
 green shell	
 red shell	
 blue shell	

 mushroom: gives player short, instantaneous boost

Any Difficulties Encountered and How They Were Overcome

The main difficulties encountered during this project were giving the illusion of a moving screen, making sure that each driver has their own item, and creating AI to interact with the player.

- Overcoming moving screen – Since I wanted to create an illusion that all drivers are driving to the right I needed to somehow scroll the background indefinitely. At first, I didn't know if this was possible but apparently you can use a for-loop and the copy() function to shift the background horizontally for every frame. Thus, an infinite scrolling background has been achieved.

- Overcoming item system – Making the item system wasn't too bad but it got tricky once I noticed that the two AI players couldn't share the same item function within the same class or else there could only be 1 item on the screen at a time (which we don't want). To fix this issue I needed to create an item function for Mario, Bowser, and Toad uniquely that way every driver has a chance to have/use an item at any given frame.

- Overcoming AI – This one was the trickiest. Since I wanted this to be a challenge for the player I needed to make AI that was both unpredictable and effective at taking down the main player. I used the Java.Random library to account for the unpredictability of the AI. The 2 main AI decisions that needed to be made were: deciding whether to move to the upper or lower lane and deciding when to throw the item. In order to calculate this decision, I made a formula such that for every certain period of seconds the AI would have a 50/50 chance of making that decision. (For example: for every 2 seconds there's a 50% chance the CPU will use their item if they have one).