GUI based game: Racket

Nikita Melentjevs, Julian Michelsen

January 11, 2019

1 requirements

Create a Gui based game using racket, accepting user needs and requirements, visually pleasing functional, bug free final product.
refining of the problem and posing possible solutions.

- Create a character for the player to control?
- Create a level scape for the player to navigate?
- Create some way to level up and increase difficulty?
- Create enemies of some sort/obstacles?
- Create an initial menu screen for the player to choose options and such.

Possible Solutions:

- portal game
- shooter game
- geometry dash

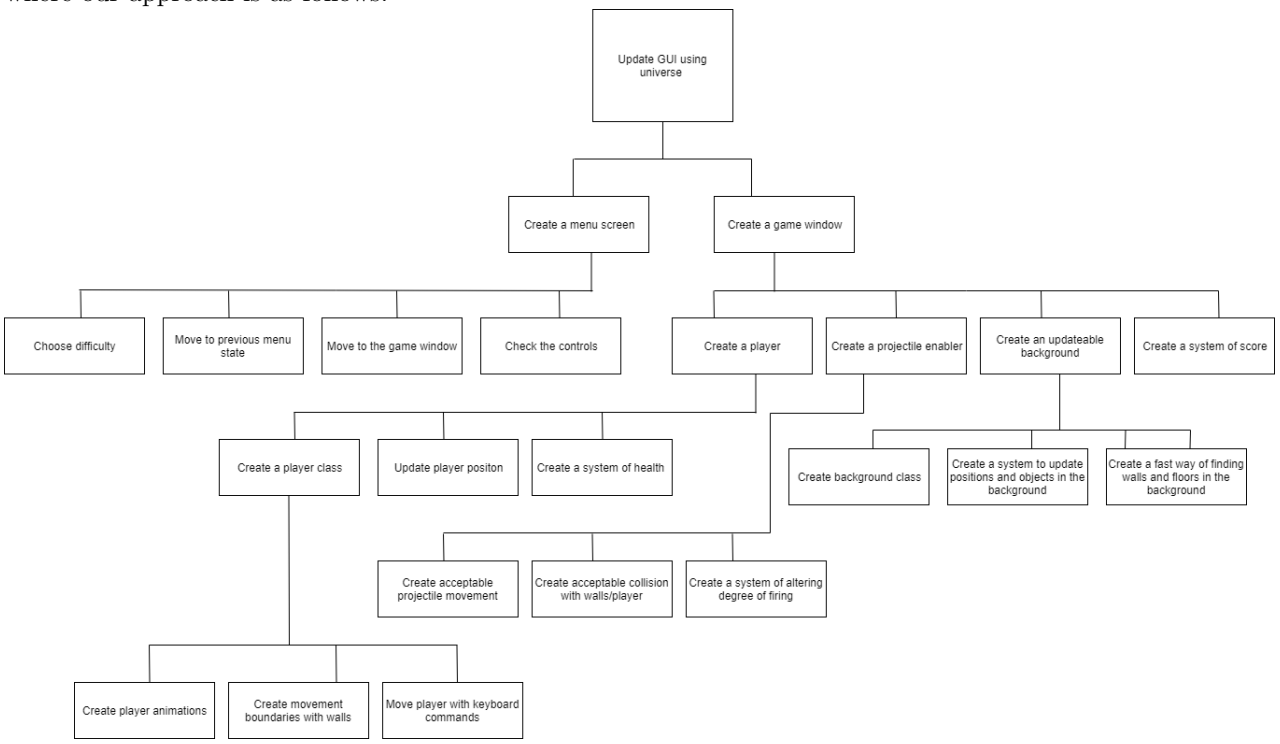
2 Design

Our game is an open-loop system as it relies on the input of the user to create any sort of desired output:

- Our input system is the keyboard where keyboard commands are entered, as well as the cursor and cursor clicks on the initial menu screen.
- The controller is the update function, where time is taken into consideration and the user inputs are taken into consideration.
- The response is the movement of the player on the GUI.

2.1 Top Down Approach

Our game is being created with top down design:
where our approach is as follows:



2.2 Use Cases

- Create movement through keyboard clicks
- Fire a projectile through keyboard clicks
- Create a background that looks reasonable
- Create enemies to avoid
- Create a system of health
- Create a system of score
- Create animations for the player
- Create an acceptable leveling system

2.3 UML Class Diagram

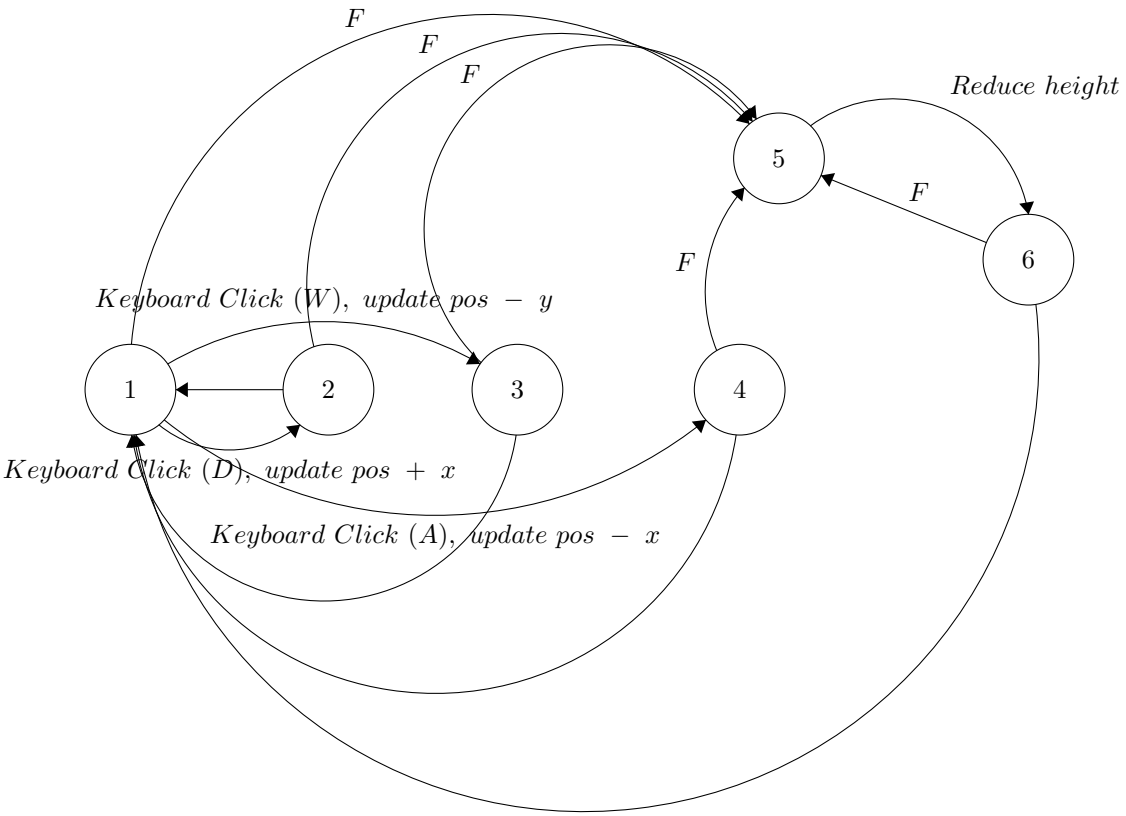
Write your subsection text here.

2.4 Different Data Structures

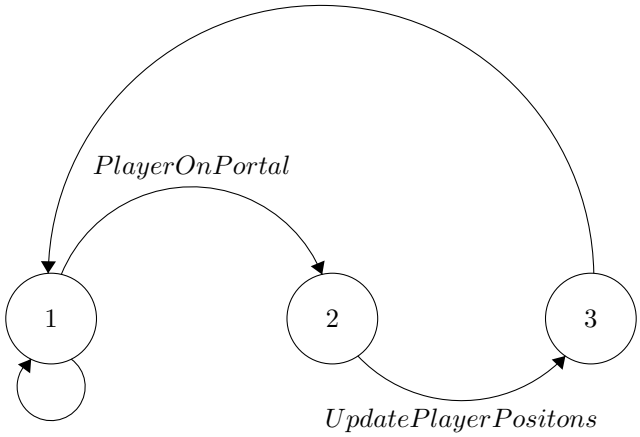
Lists, Hash tables, sets

2.5 FSM's

Movement through Keyboard Click: where F represents a check-falling function
Blank line represents time passing.

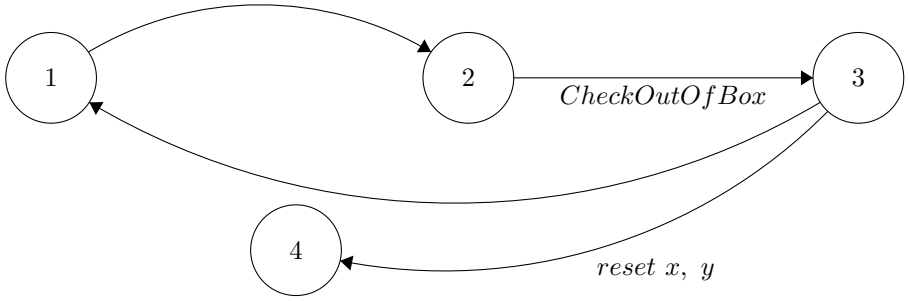


Portal Relocation:



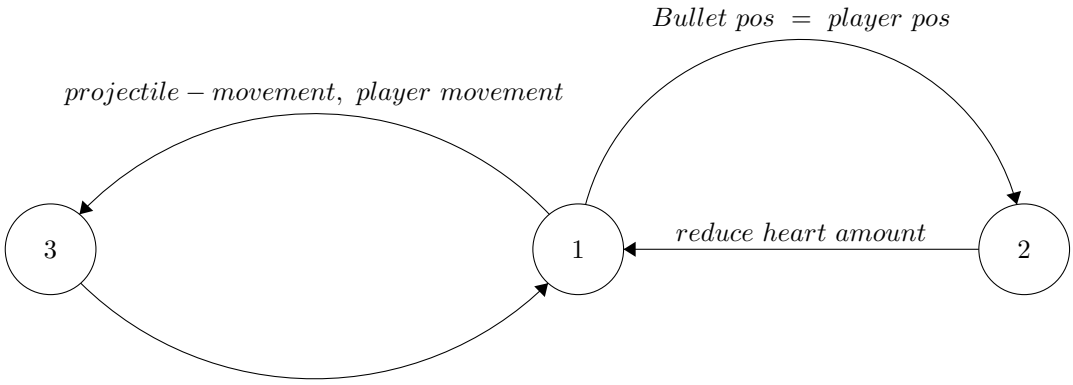
Projectile updating after being fired:

$x = x + 3,\ y = y((tan(piAngle/180) * 3 * (-1))$



2.6 EFSM's

Life reduction EFSM with movement and projectile movement:



2.7 Human Interface

The color background that we have decided to use makes it very easy to distinguish between the walls and the background. The bullet color and the enemy color was chosen to be red as it is the most commonly used color for dangerous objects, so that the player instinctively understands that they should avoid the bullets without being told to. The star in the interface is also a universal symbol

for an object that should be collected. Otherwise player may not know whether or not to move towards it.
The hearts are placed at the top of the screen to be easily visible by the player.

3 Implementation

3.1 Pre-conditions and Post-conditions

Write your subsection text here.

3.2 Specification vs Implementation

Write your subsection text here.

3.3 Side Effects

The lifebarchange funcion not only changes the amount of hearts the player currently has, but also the bullet class field has-hit, this is an example of where side effects can be very useful, when trigger-checks are updated so that they cannot be triggered again. The change function alters almost all the classes in

the program, as well as our hash-table that is used for the movement and portal gun rotation.

4 Testing and Maintaining

4.1 Test Cases and Assertions

- test cases for movement
- test cases for teleporting through portals
- test cases for

4.2 User Feedback

survey-link here

Animations	Color Scheme and Design	Fun and Overall Playability
cell4	cell5	cell6
cell7	cell8	cell9

4.3 Subsection Heading Here

Interface requirements from survey results + fix.

5 Conclusion

Write your conclusion here.