FIT2102 Programming Paradigm
Assignment 1
Report

Name: Lai Ying Ying
Student ID: 3052 6361

Summarise the workings of the code and highlight the interesting parts.

1. A ball, player's paddle, cpu's paddle and text elements are created in the html file.

2. Code out the logic using observables for:
- controlling the player's paddle using the mouse
- moving the ball in a certain direction and change the velocity when it hits the paddles, ceiling or wall
- tracking the ball so the cpu's paddle can follow the ball
- keep track of the score so it will end the game when the score is 7
- restart the game when the player presses 'Enter' by calling the pong() function again.

3. Documentation

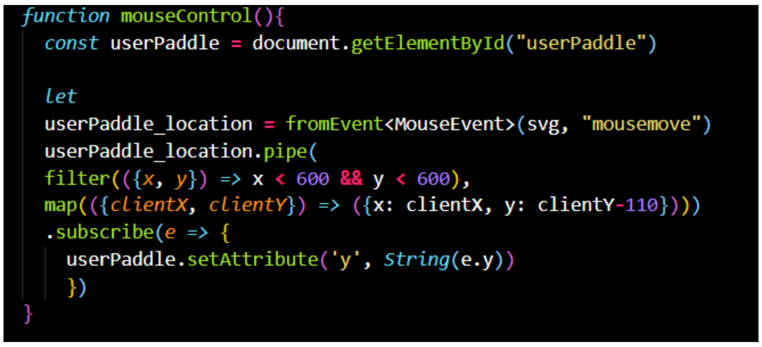
Design decisions and justification

1. All the function for the features of the game are encapsulated inside the pong() function as it is easier to restart the game by calling the function again.

2. Observables are used for the other logics so that it can observe the changes and pass in the observer to subscribe() to execute the observable. For example, instead of declaring a mutable variable for keeping track of the score, an observable is used instead. This can prevent using mutable data.

Explain how you tried to follow the FRP style

1. map() and filter() are composed using pipe to map the array of values and filter the value according to the logic required. This can help greatly is reducing the use of if and else statement, and also define complex transformations from simple, reusable elements.



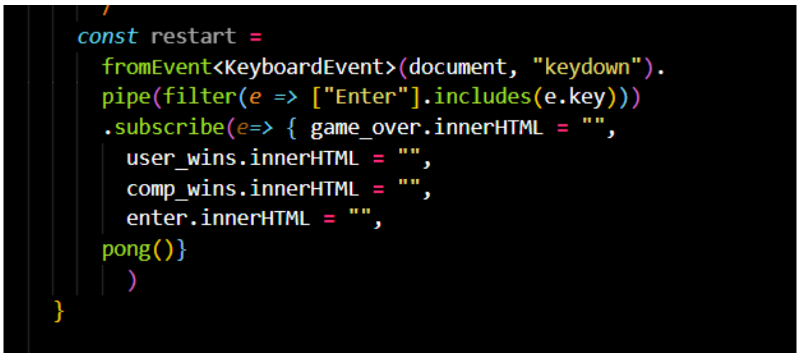
```
function mouseControl(){
  const userPaddle = document.getElementById("userPaddle")

  let
  userPaddle_location = fromEvent<MouseEvent>(svg, "mousemove")
  userPaddle_location.pipe(
  filter(({x, y}) => x < 600 && y < 600),
  map(({clientX, clientY}) => ({x: clientX, y: clientY-110})))
  .subscribe(e => {
    userPaddle.setAttribute('y', String(e.y))
    })
}
```

2. ‘event listeners’ are added to the html elements, which invoke the specified functions when the event fires. For examples, moving the user’s paddle using mouse and pressing ‘Enter’ to restart the game.

<u>How you manage state throughout your game</u>

1. When either one of the scores reach 7, the user will press ‘Enter’ if he decides to restart the game, the pong() function will be called and another game will start.



```
    const restart =
      fromEvent<KeyboardEvent>(document, "keydown").
      pipe(filter(e => ["Enter"].includes(e.key)))
      .subscribe(e=> { game_over.innerHTML = "",
        user_wins.innerHTML = "",
        comp_wins.innerHTML = "",
        enter.innerHTML = "",
      pong()}
        )
  }
```