Javascript Final Project

Dog Volleyball

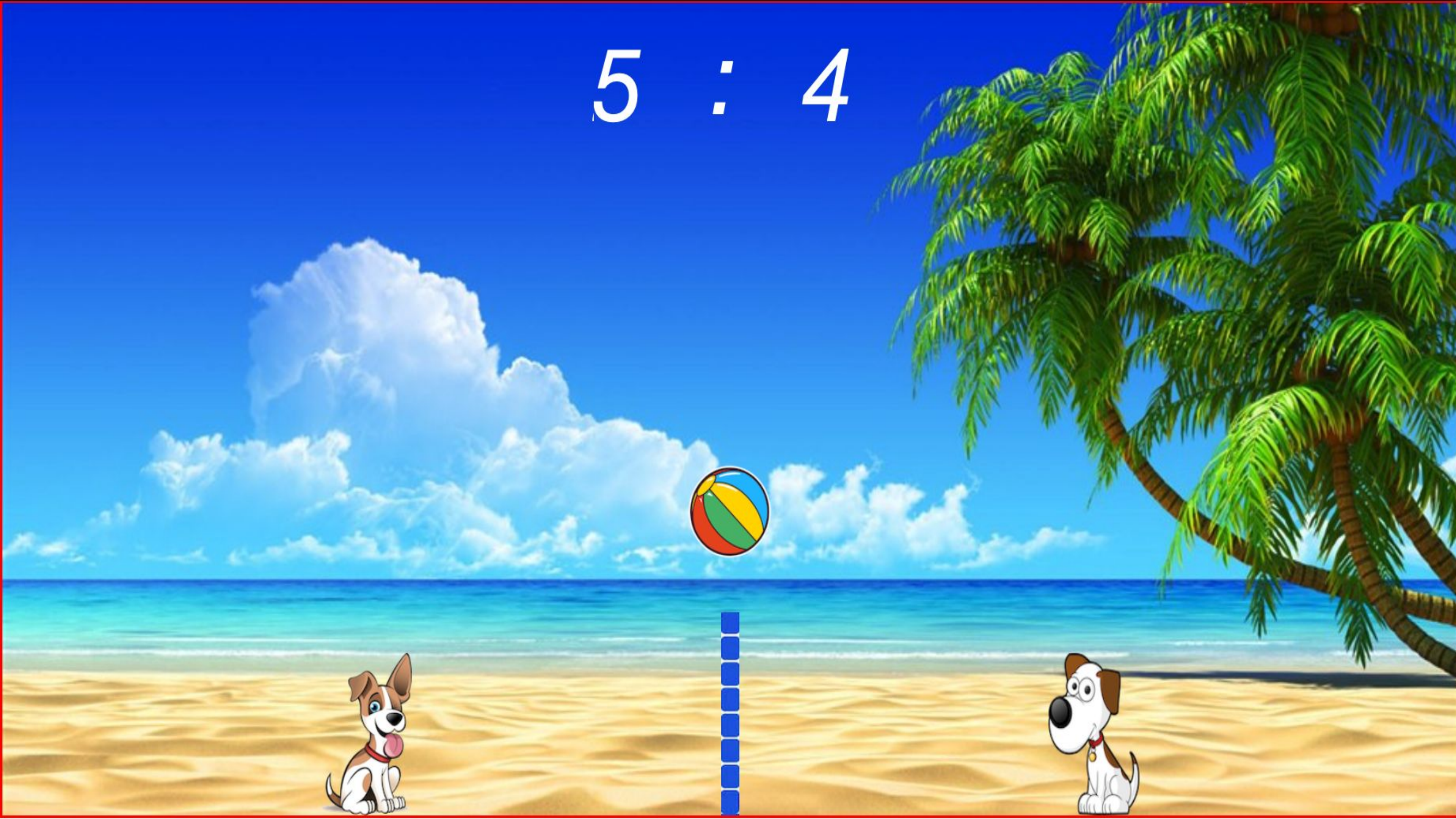
0716050 吳泓毅

Goals

- Motivated by the Classic Game, Pikachu Volleyball.
- Rebuild the Game by Myself.

Current Progress

- Ball can move smoothly
- Dogs can go left, right, up, and down smoothly
- Dogs can’t jump
- Collision Detection works


5 : 4

Development

- HTML + CSS + JavaScript
- Use VS Code
- Demo on Safari 14.0.3

Structure

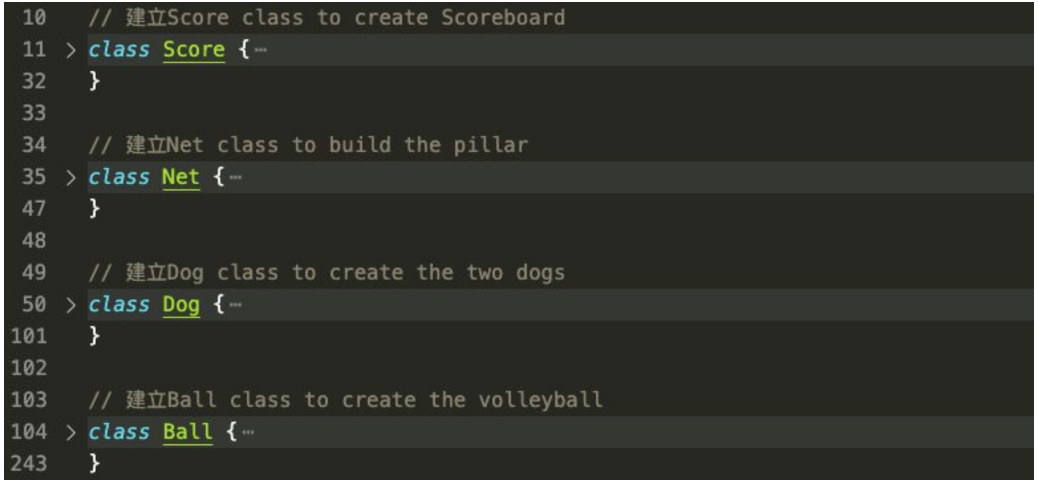
- HTML
- CSS
- JS
- Images

About My Project

- 1 Ball, 1 Net, 2 Dogs, a ScoreBoard
- Dog on the Left Serve the Ball First
- Dog Winning Last Round Serve the Ball this Round
- 11 Points to Win

Objects

- Classes for the 4 Types of Objects

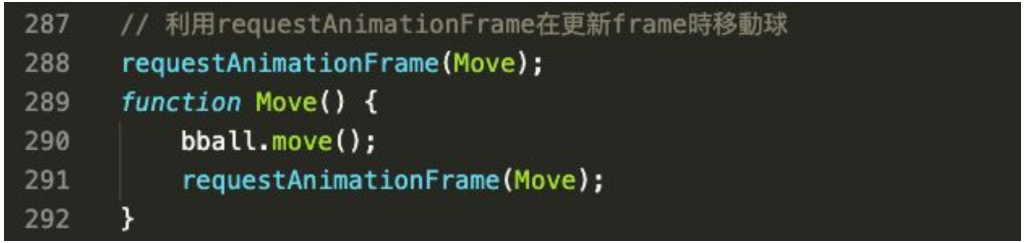


```
// 建立Score class to create Scoreboard
class Score {⋯
}

// 建立Net class to build the pillar
class Net {⋯
}

// 建立Dog class to create the two dogs
class Dog {⋯
}

// 建立Ball class to create the volleyball
class Ball {⋯
}
```

Move of Ball

- Use requestAnimationFrame to make the ball move smoothly.
- requestAnimationFrame()
    - To Make Objects Move Smoothly.
    - Update the Screen Frame by Frame.

- Recursive Call to let the ball keep moving



```
// 利用requestAnimationFrame在更新frame時移動球
requestAnimationFrame(Move);
function Move() {
    bball.move();
    requestAnimationFrame(Move);
}
```

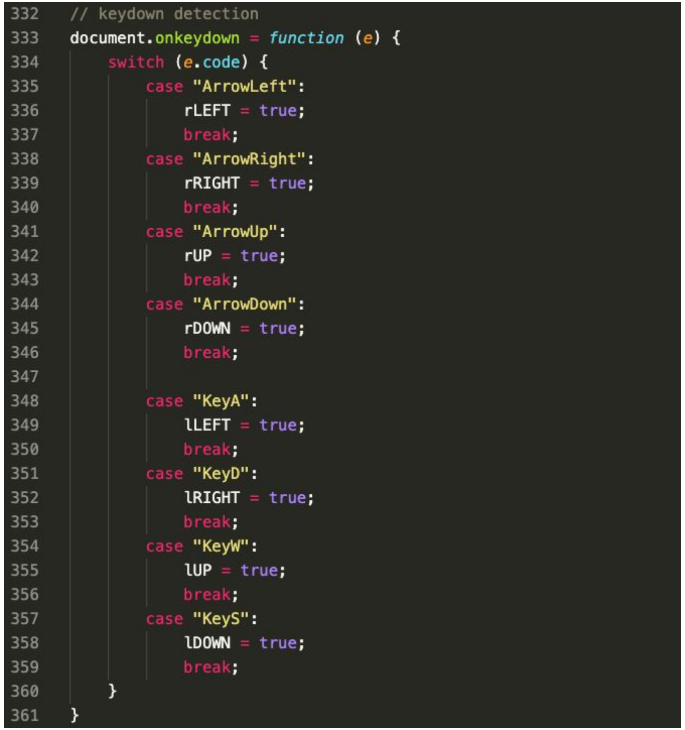
Keyboard Triggers Move of Dogs

- Move() function to move smoothly with requestAnimationFrame().
- Use Boolean variables to control Move().
- Use onkeydown to detect keyboard down.
    - determine direction and set such boolean to true to trigger Move().
- Use onkeyup to detect keyboard up.
    - determine direction and set such boolean to false to stop Move().

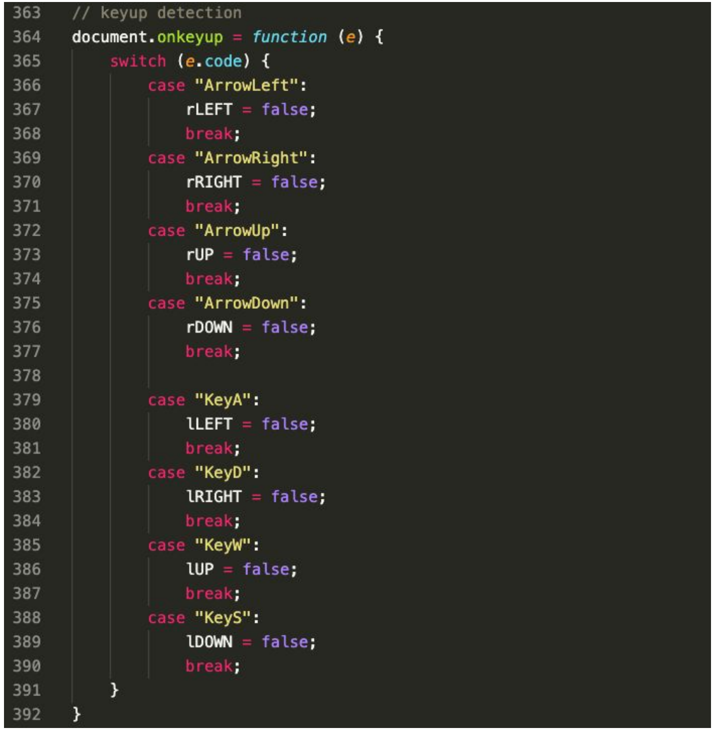
Functions

- onkeydown
  - once a set key is pressed down and holded, it triggers.
- onkeyup
  - once a set key is not being pressed, it triggers.

Keyboard Detection

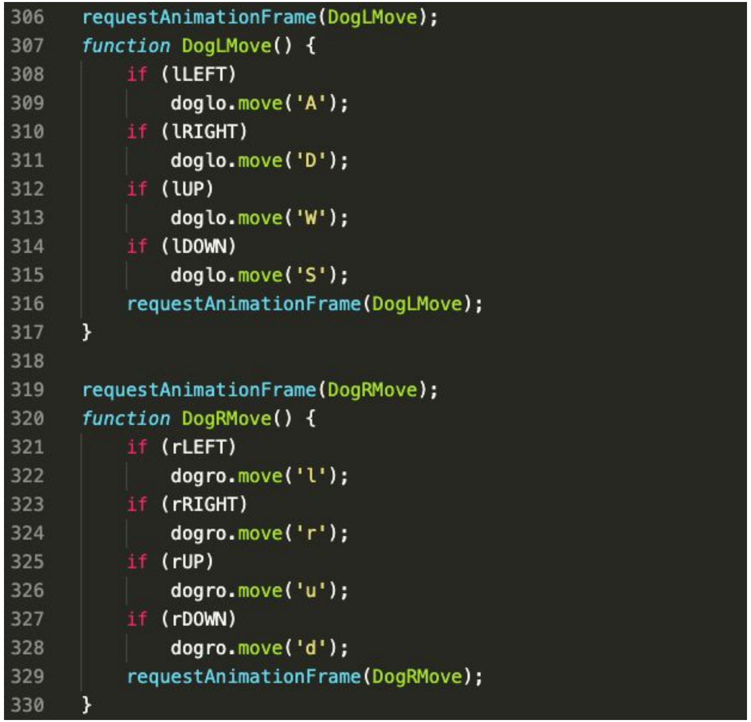


```
// keydown detection
document.onkeydown = function (e) {
    switch (e.code) {
        case "ArrowLeft":
            rLEFT = true;
            break;
        case "ArrowRight":
            rRIGHT = true;
            break;
        case "ArrowUp":
            rUP = true;
            break;
        case "ArrowDown":
            rDOWN = true;
            break;

        case "KeyA":
            lLEFT = true;
            break;
        case "KeyD":
            lRIGHT = true;
            break;
        case "KeyW":
            lUP = true;
            break;
        case "KeyS":
            lDOWN = true;
            break;
    }
}
```



```
// keyup detection
document.onkeyup = function (e) {
    switch (e.code) {
        case "ArrowLeft":
            rLEFT = false;
            break;
        case "ArrowRight":
            rRIGHT = false;
            break;
        case "ArrowUp":
            rUP = false;
            break;
        case "ArrowDown":
            rDOWN = false;
            break;

        case "KeyA":
            lLEFT = false;
            break;
        case "KeyD":
            lRIGHT = false;
            break;
        case "KeyW":
            lUP = false;
            break;
        case "KeyS":
            lDOWN = false;
            break;
    }
}
```

Dog Move



```
requestAnimationFrame(DoglMove);
function DoglMove() {
    if (lLEFT)
        doglo.move('A');
    if (lRIGHT)
        doglo.move('D');
    if (lUP)
        doglo.move('W');
    if (lDOWN)
        doglo.move('S');
    requestAnimationFrame(DoglMove);
}

requestAnimationFrame(DogRMove);
function DogRMove() {
    if (rLEFT)
        dogro.move('l');
    if (rRIGHT)
        dogro.move('r');
    if (rUP)
        dogro.move('u');
    if (rDOWN)
        dogro.move('d');
    requestAnimationFrame(DogRMove);
}
```

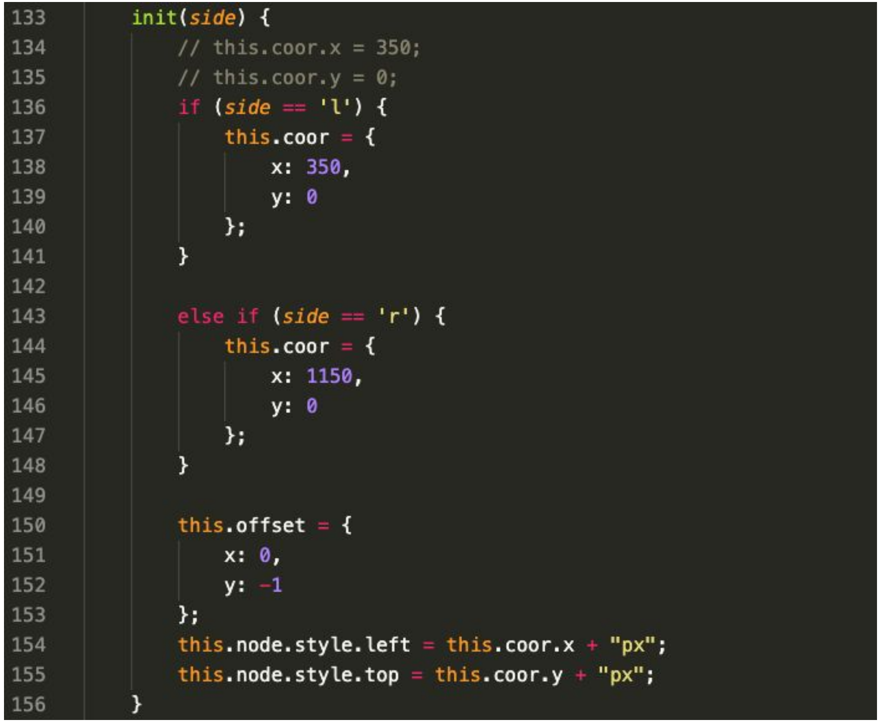
Functions in Ball Class

- init()
  - for initialization of the ball position and offset.
- move()
  - for detections of events to determine the position and offset of the ball.

init(side)

- When a round is over, the ball has to drop from the air on the side of the last winning side.
- Who serves the ball this round is according to the result of the last round.

init(side)



```
init(side) {
    // this.coor.x = 350;
    // this.coor.y = 0;
    if (side == 'l') {
        this.coor = {
            x: 350,
            y: 0
        };
    }

    else if (side == 'r') {
        this.coor = {
            x: 1150,
            y: 0
        };
    }

    this.offset = {
        x: 0,
        y: -1
    };
    this.node.style.left = this.coor.x + "px";
    this.node.style.top = this.coor.y + "px";
}
```

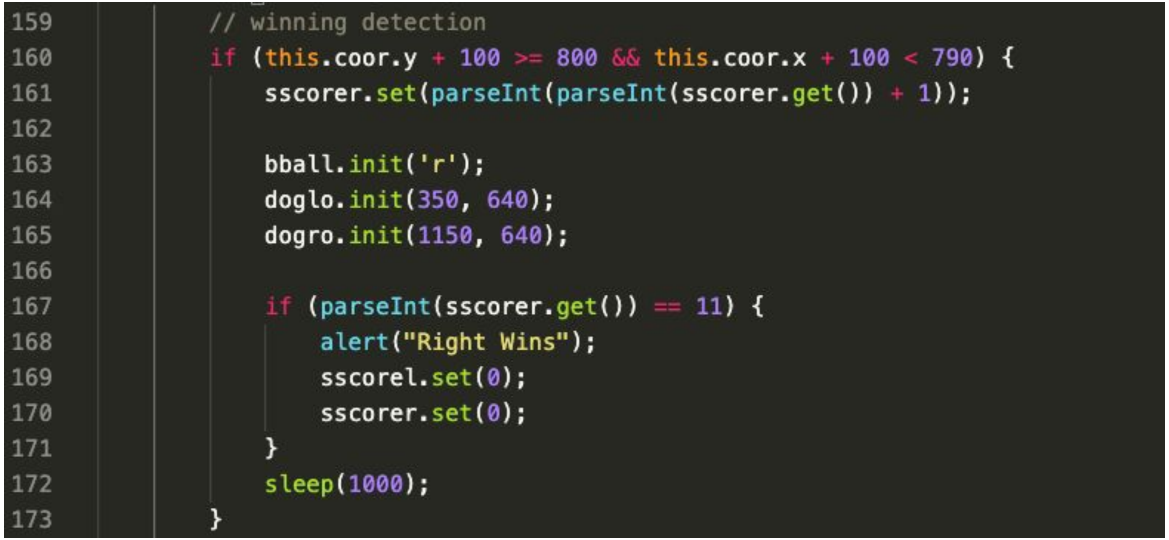
move()

- Winning Detection
- Collision Detection between the Dogs and the Ball
- Collision Detection between the Ball and the Boundaries and the Net

Winning Detection

- Round Win
  - add points to scoreboard.
  - reset ball position and offset and dogs positions.
- Game Win
  - alert winning message and reset scoreboard.

Winning Detection



```
        // winning detection
        if (this.coor.y + 100 >= 800 && this.coor.x + 100 < 790) {
            sscorer.set(parseInt(parseInt(sscorer.get()) + 1));

            bball.init('r');
            doglo.init(350, 640);
            dogro.init(1150, 640);

            if (parseInt(sscorer.get()) == 11) {
                alert("Right Wins");
                sscorel.set(0);
                sscorer.set(0);
            }
            sleep(1000);
        }
```

TODO...

- Make JUMP more Reasonable to follow Physical Principles.
- Add POWER HIT
- Design Better UI

Problems met

- Wanted to Use PixiJS to Render the Game.
    - can make the ball rotates easily
    - can add different dog postures easily

Remarks

- Feel good to make a game by myself.
- Not sophisticated.
- Need to Modify if time’s enough.

THANKS
For
Listening