Classic Arcade Game Clone

▼ js/app.js 10

```
1  let score=0;
```

SUGGESTION

It is a good practice to include the "use strict"; tag in your JavaScript files.

'use strict' is helpful because it forces you to write better code by preventing functions with bad syntax from e
requiring you to declare a variable before using it. Sometimes when you implement 'use strict' you may find t
doesn't work. That's because you have some sloppy coding mistakes that need to be fixed. It's like having a ve
master sitting on your shoulder making sure that everything is correct before you can move on.

```
2
3  class character {
4    constructor(x,y,sprite) {
5      this.x = x;
6      this.y = y;
7      this.sprite = sprite;
8    };
9
10   render() {
11     ctx.drawImage(Resources.get(this.sprite), this.x, this.y);
12   };
13 };
14
15 // Enemies our player must avoid
16 class Enemy extends character {
```

AWESOME

Love the way you have used inheritance concepts 💥

```
17   constructor(x,y,sprite) {
18     super(x,y,sprite='images/enemy-bug.png');
19     this.speed = 100 + Math.floor(Math.random() * 512);
```

AWESOME

Nice job randomizing enemy speed 🎉

```
20
21   };
22
23   update(dt) {
24     // You should multiply any movement by the dt parameter
25     // which will ensure the game runs at the same speed for
26     // all computers.
27     this.x += this.speed * dt;
28     if (this.x>550){
29       this.x = -100;
```

AWESOME

Nice job playing around with the 'this' keyword 💥

```
30      this.speed = 100 + Math.floor(Math.random() * 512);
31    }
32    if ((player.y >= this.y - 50 && player.y <= this.y + 50) && (player.x >= this.x - 50 && player.x <=
```

AWESOME

Collisions functionality works completely fine 💫 ⭐

```
33      player.x = 200;
34      player.y = 300
35    }
36  };
37 };
38
39
40
41
42 // Now write your own player class
43 // This class requires an update(), render() and
44 // a handleInput() method.
45 class Player extends character {
46   constructor(x,y,sprite) {
47     super(x,y,sprite='images/char-boy.png');
```

SUGGESTION

You can also give a user multiple character options to choose from 💡

```
48   };
49
50   update() {
51     if (this.x > 400) this.x = 400;
52     if (this.x < 0) this.x = 0;
53     if (this.y > 380) this.y = 380;
54     if (this.y < 0){
55       this.y = 380;
56       this.x = 200;
57       score += 1;
```

AWESOME

Good job keeping track of the score 🎉

```
58       swal({
```

AWESOME

Amazing job using sweet alert 💥 🌟
Love this functionality 😄

```
59         allowEscapeKey: false,
60         allowOutsideClick: false,
61         title: 'Congratulations! You Won!',
62         text: 'Your ' + (( score> 1) ? "scores are " : "score is ") + score +' .\n Woooooo!',
63         type: 'success',
64         confirmButtonColor: '#02ccba',
65         confirmButtonText: 'Play again!'
```

AWESOME

Always nice to give a PLAY AGAIN option to the user.
Definitely adds to the user experience ✨ 💫

```
66       }).then(function (isConfirm) {
67         if (isConfirm) {
68           // do nothing as the player will be back by default;
```

AWESOME

Nice job resetting the game ✨

```
69         }
70       })
71
72     }
```

```
};

handleInput(key) {
  switch (key) {
    case 'right':
    this.x += 101;
    break;
    case 'left':
    this.x -= 101;
    break;
    case 'down':
    this.y += 90;
    break;
    case 'up':
    this.y -= 90;
    break;
    // default:
  }
};
};


// Now instantiate your objects.
// Place all enemy objects in an array called allEnemies
// Place the player object in a variable called player

let allEnemies=[], ePosition=[60,140,220];
let player = new Player(200,380,50), e;

ePosition.forEach(function(y){
  // debugger
  ss = 100 + Math.floor(Math.random()*512);
  e = new Enemy(0,y,ss);
  allEnemies.push(e);
});

// This listens for key presses and sends the keys to your
// Player.handleInput() method. You don't need to modify this.
document.addEventListener('keyup', function(e) {
  var allowedKeys = {
    37: 'left',
    38: 'up',
    39: 'right',
    40: 'down'
  };

  player.handleInput(allowedKeys[e.keyCode]);
});

```

▸ README.md 2

▸ js/resources.js

▸ js/engine.js

▸ css/style.css

▸ index.html

RETURN TO PATH