UDACITY

PROJECT

Classic Arcade Game Clone

A part of the Front-End Web Developer Nanodegree Program

PROJECT REVIEW

CODE REVIEW 5

NOTES

▾ js/app.js 4

```
// Whole-script strict mode syntax
'use strict';
// Enemies our player must avoid
var Enemy = function(x, y) {
    // Variables applied to each of our instances go here,
    // we've provided one for you to get started

    // The image/sprite for our enemies, this uses
    // a helper we've provided to easily load images
    this.sprite = 'images/enemy-bug.png';
    // enemy coordinate (x,y)
    this.x = x;
    this.y = y;
    // enemy moving speed
    this.speed = Math.floor(100 + Math.random()*100);
};

// Update the enemy's position, required method for game
// Parameter: dt, a time delta between ticks
Enemy.prototype.update = function(dt) {
    // You should multiply any movement by the dt parameter
    // which will ensure the game runs at the same speed for
    // all computers.
    if (this.x <= 505) this.x += this.speed * dt;
    else this.x = -30;
    if (player.x > this.x - 80 && player.x < this.x + 80) {
        if (player.y > this.y - 20 && player.y < this.y + 20)
```

AWESOME

Excellent! You're using one of the most comprehensive versions of the basic collision detection algorithm.

```
            player.reset();
    }
};

// Draw the enemy on the screen, required method for game
Enemy.prototype.render = function() {
    ctx.drawImage(Resources.get(this.sprite), this.x, this.y);
};

```

SUGGESTION

Developer-to-Developer tip

If you want to take your codes to the next level in future projects, why don't you try object inheritance? As you can see, in this project, for example, the `Enemy` and `Player` objects have common attributes/properties (in this case mainly the `x` and `y` properties), so why hardwiring or repeating yourself (remember the DRY - Don't Repeat Yourself - principle) when you can simply use available things?

This is simply a tip to further expand your knowledge of JavaScript and Object Oriented Programming, and it is not required to successfully pass this project.

Here are a couple of resources to start with, if you're interested of course!

http://javascriptissexy.com/oop-in-javascript-what-you-need-to-know/
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Inheritance_and_the_prototype_chain
http://www.crockford.com/javascript/inheritance.html

```
37 // Now write your own player class
38 // This class requires an update(), render() and
39 // a handleInput() method.
40 var Player = function() {
41     this.sprite = 'images/char-boy.png'
42     this.x = 200;
43     this.y = 400;
44 };
45
46 // reset player's location
47 Player.prototype.reset = function() {
48     this.x = 200;
49     this.y = 400;
50 };
51
52 // Update the player's position based on keyboard input
53 // if collision with an enemy, reset player's position
54 Player.prototype.update = function(dt) {
55     if (this.pressedKey == 'left' && this.x > 0) this.x -= 101;
56     if (this.pressedKey == 'right' && this.x < 400) this.x += 101;
57     if (this.pressedKey == 'up' && this.y > 0) this.y -= 83;
58     if (this.pressedKey == 'down' && this.y < 400) this.y += 83;
59     // reset the pressedKey
60     this.pressedKey = null;
61     if (this.y < -10) {
62         alert("You won!");
63         this.reset();
64     }
65 };
66
67 // Draw the player on the screen, required method for game
68 Player.prototype.render = function() {
69     ctx.drawImage(Resources.get(this.sprite), this.x, this.y);
70 };
71
72 Player.prototype.handleInput = function(command) {
73     this.pressedKey = command;
74 };
75
76 // Now instantiate your objects.
77 // Place all enemy objects in an array called allEnemies
78 // Place the player object in a variable called player
79
```

SUGGESTION

If you are interested in Game Development, perhaps the below references can help you to code games like a pro! :-)

References
http://codeincomplete.com/posts/2013/12/10/javascript_game_foundations_state_management/
http://stackoverflow.com/questions/18038502/how-to-code-a-html5-game-with-distinct-game-states
http://gamedevelopment.tutsplus.com/tutorials/finite-state-machines-theory-and-implementation--gamedev-11867
https://github.com/jakesgordon/javascript-state-machine

```
80 var player = new Player();
81 var allEnemies = [];
82 allEnemies.push(new Enemy(-10, 50));
83 allEnemies.push(new Enemy(-10, 140));
84 allEnemies.push(new Enemy(-10, 230));
85
86 // This listens for key presses and sends the keys to your
87 // Player.handleInput() method. You don't need to modify this.
88 document.addEventListener('keyup', function(e) {
89     var allowedKeys = {
90         37: 'left',
91         38: 'up',
92         39: 'right',
93         40: 'down'
94     };
95
96     player.handleInput(allowedKeys[e.keyCode]);
```

AWESOME

In general, excellent work in the file! :-)

```
97 });
98
```

README.md 1

js/resources.js

js/engine.js

index.html

css/style.css

RETURN TO PATH

Rate this review

Student FAQ