cs174A-dis1B-week0

January 10, 2020

1 CS-174A Discussion 1B, Week 0

@ Ali Hatamizadeh
@ ROLFE 3126 / Friday / 2:00pm- 3:50pm
@ https://github.com/ahatamiz/cs174a-1b-2020w

2 Outline

- About this course
- JavaScript and WebGL Basic
- Assignment 1

3 CS-174A Introduction

3.1 About Me:

- Ali Hatamizadeh, Ph.D student in Computer Science
- Office hours: Eng-VI 366, Friday 4:00 - 6:00 PM
- Email: ahatamiz@ucla.edu

3.1.1 Grading Policies

- 4 assignments (0 + 10 + 10 + 10): 30 pts
- Team project: 30 pts
- Midterm: 15 pts
- Final: 25 pts

(May change. Stay tuned till the next lecture)

4 JavaScript Basics

JavaScript can change HTML content

```
In [1]: %%html
        <p id="demo">JavaScript can change HTML content.</p>
        <button type="button" onclick='document.getElementById("demo").innerHTML = "Hello JavaS
```

```
<IPython.core.display.HTML object>
```

4.1 Let vs Var vs Constant

var: When you declare a variable with var, its scope is not limited to the block in which it is defined. It's limited to the function in which it is defined.

```
function start(){
for (var i=0;i<5;i++){

}

element.text(i)

}
```

```
In [2]: %%js

        function start(){
        for (var i=0;i<5;i++){
        }

        element.text(i);


        }

        start()
```

```
<IPython.core.display.Javascript object>
```

4.2 Let vs Var vs Constant

let and constant are block-scoped

```
function start(){
for (let i=0;i<5;i++){
}

element.text(i)

}
```

```
In [3]: %%js

        function start(){
        for (let i=0;i<5;i++){
```

```
        }

        element.text(i)

        }

        start()
```

```
<IPython.core.display.Javascript object>
```

4.3 Some additional points

When you use var outside of a function, it creates a global variable and attaches it to the window object in the browser.

When you use let to create a global variable, it is not attached to the window object.

4.4 Variables and Data types

JavaScript variables are containers for storing data values.

JavaScript variables can hold many **data types**: numbers, strings, objects and more:

```
var length = 16;                               // Number
var lastName = "Johnson";                      // String
var x = {firstName:"John", lastName:"Doe"}; // Object
```

```
In [4]: %%js
        var num = 16;                                    // Number
        element.text("The number is " + num)
```

```
<IPython.core.display.Javascript object>
```

```
In [5]: %%js
        element.text('here we go')
```

```
<IPython.core.display.Javascript object>
```

4.5 Objects

You define (and create) a JavaScript object with an object literal:

```
var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
```

4.6 Objects

Another way of creating object You define (and create) a JavaScript object with an object literal:

```
var person= new Object()
person.firstName="John"
person.age=50
person.eyeColor="blue"
```

```
In [6]: %%js
        var person= new Object()
        person.firstName="John"
        person.age=50
        person.eyeColor="blue"
        element.text(person['eyeColor'])
```

```
<IPython.core.display.Javascript object>
```

```
In [7]: %%js
        var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
        element.text(person.firstName + "'s age is " + person["age"]);  // two ways for accessi

        var name = new String("John");
        var name_2 = "John";
        element.text(name === "John");
```

```
<IPython.core.display.Javascript object>
```

4.7 Functions

A JavaScript function is a block of code designed to perform a particular task.

A JavaScript function is executed when "something" invokes it (calls it).

```
In [8]: %%js
        function myFunction(p1, p2) {
          return p1 * p2;   // The function returns the product of p1 and p2
        }

        var a = 3;
        var b = 4;
        element.text("The product of a and b is " + myFunction(a,b))
```

```
<IPython.core.display.Javascript object>
```

4.8 Factory Functions

A factory function creates an object

```
function createCircle(radius,location){

    return {
        radius:radius,
        location:location,
        visible:true,
        draw: function(){element.text('Here we go : draw')}
    }

}
```

```
In [9]: %%js
        function createCircle(radius){

            return {
                radius:radius,
                visible:true,
                draw: function(){element.text('draw')}
            }

        }

        const circle1=createCircle(1)
        circle1.draw()
```

```
<IPython.core.display.Javascript object>
```

4.9 JavaScript this Keyword

" this " in JavaScript refers to the object that is executing the current function

If the function is part of an object (in other words is a method of that object): " this " refrences the object itself

Otherwise, " this " refers to the global object (which is window object in browsers)

```
In [10]: %%js

         const video ={
             title:'a',
             play(){
                 element.text(this)
             }
         }

         video.play()
```

```
<IPython.core.display.Javascript object>
```

```
In [11]: %%js

         const video ={
             title:'a',
             play(){
                 element.text(this)
             }
         }

         function playVideo(){
             element.text(this)
         }

         playVideo()
```

```
<IPython.core.display.Javascript object>
```

4.10 Constructor Functions

A constructor function also creates an object. Be aware of the naming conventions:
Camel notation: oneTwoThree (Used for naming factory functions)
Pascal notation : OneTwoThree (Used for naming constructor functions)

```
function Circle(radius,location){
    //this here is an empty object to which we add radius and location
    //JavaScripts objects are dynamic. Once created, we can add aditional methods to them.
    this.radius=radius
    this.location=location
    this.visible=true
    this.draw=function(){element.text('Here we go : draw')}

}
```

```
In [12]: %%js

         function Circle(radius){
             //this here is an empty object to which we add radius
             //JavaScripts objects are dynamic. Once created, we can add aditional methods to t
             this.radius=radius
             this.visible=true
             this.draw=function(){element.text('Here we go : draw')}

         }
```

```
         const circle1=new Circle(1)
         circle1.draw()
```

```
<IPython.core.display.Javascript object>
```

4.11 Object Methods

```
In [13]: %%js
         var person = {
           firstName: "John",
           lastName : "Doe",
           id       : 5566,
           fullName : function() {
             return this.firstName + " " + this.lastName;
           }
         };
         element.text(person.fullName())
```

```
<IPython.core.display.Javascript object>
```

4.11.1 Object Constructors

```
In [14]: %%js
         function Person(first, last, age, eye) {
           this.firstName = first;
           this.lastName = last;
           this.age = age;
           this.eyeColor = eye;
         }

         var myFather = new Person("John", "Doe", 50, "blue");

         element.text("My father is " + myFather.firstName + " " + myFather.lastName + ".")
         //element.text(myFather)
```

```
<IPython.core.display.Javascript object>
```

4.12 JavaScript Classes

A class is a type of function, but instead of using the keyword `function` to initiate it, we use the keyword `class`, and the properties is assigned inside a `constructor()` method.

4.12.1 Class Definition

Use the keyword `class` to create a class, and always add a `constructor` method.
The `constructor` method is called each time the class object is initialized.

```
class Car {
  constructor(brand) {
    this.carname = brand;
  }
}
mycar = new Car("Ford");
```

4.13 A Quick Way to Learn JS

- https://www.w3schools.com/js/

5 WebGL Basics

WebGL (Web Graphics Library) is a JavaScript API for rendering interactive 3D and 2D graphics within any compatible web browser without the use of plug-ins. WebGL does so by introducing an API that closely conforms to OpenGL ES 2.0 that can be used in HTML5 `<canvas>` elements.

- Demo 0: Clearing with colors. How to clear the rendering context with a solid color.
- Demo 1: Simple color animation. A very basic color animation.

5.1 ... and some other projects

- WebGL Wather
- Make me pulse wish 2017

6 Assignment 1: Environment Setup and Creating a Simple Rectangle

Posted on CCLE (Week 1)