CSC435: Web Programming

Lecture 10: JavaScript: Objects

Bei Xiao

American University

Feb 15, Monday, 2016

Activity Outline

- Exercises
- More on Event-driven Programming
- Form inputs
- JavaScript DOM
- Unobtrusive JavaScript
- JavaScript Object

Define Functions

```
function name() {
  statement ;
  statement ;
  ...
  statement ;
}
```

```
function myFunction() {
  alert("Hello!");
  alert("How are you?");
}
```

The above could be the contents of example.js linked to our HTML page.
Statements placed into functions can be evaluated in responses to user events.
To display results, you can use document.write()

Exercise 0: move the ball

Click button below to move the image to right

Click Me

Hint: You can access to ball's position as this:

```
 imgObj = document.getElementById('myImage');
  imgObj.style.position= 'relative';
  imgObj.style.left = '0px';

// to move the ball

parseInt(imgObj.style.left)
```

Accessing an Element: Document.getElementById

```
let name= document.getElementByID("id");
```

document.getElementById returns the DOM object for an element with a given id(note that you omit the # when

Exercise 0: move the ball

```
<script type="text/javascript">
        let imgObj = null;

        function init(){
          imgObj = document.getElementById('myImage');
          imgObj.style.position= 'relative';
          imgObj.style.left = '0px';
        }

        function moveRight(){
          imgObj.style.left = parseInt(imgObj.style.left) + 10 + 'px';
        }

        window.onload =init;

    </script>
```

Exercise 1: smallest

Write a function named **findMin** that accepts an array of numbers as a parameter and returns the smallest number in the array. For example, if an array variable named nums stored the following values:

```
let nums = [-1, 3.2, 12, 15, -4, 1, -12.5, 1, 8];
```

Then the call of findMin(nums) should return -12.5 since that is the smallest numerical value in the array.

You may assume that the array passed to your function is non-empty and contains only number types.

To display results, you can use document.write()

Or you can create a button to trigger your function

```
<button onclick=”findmin();">Click me!</button>
```

Exercise 2: reverse a number

- Write a JavaScript function that reverse a number. E.g: 25368 -> 86352

- Step 1: Create a simple .html file.
- Step 2: Write a function and save it as ReverseString.js
- Step 3: link the .js into the <head></head> in your .html

- First change the number to string.
- Then n.split(“”).reverse().join(“”);

- Again, you can create a button to trigger the function or use alter.

Exercise 2: reverse a number
e.g. 123->321
123.456>654.321

Solution 1 Negative numbers?

```
function rev_num()
{
let num = prompt("Enter the number to be reveresed :", " ");
let z = num.split("").reverse().join("");
let rev = Number(z);

document.write("The given number is : " +num+ " <br/> The reversed number is : " +rev+ "\n");
}
```

Solution 2 (does it work with decimal numbers? handle negative numbers?)

```
function rev_num() {
   num  = prompt("Enter the number to be reveresed :", " ")
   n = parseInt(num);

   for(let r = 0; n; n = Math.floor(n / 10)) {
      r *= 10;
      r += n % 10;
   }
   document.write("The given number is : " +num+ " <br/> The reversed number is : " +r+ "\n");
}
```

There are many other solutions, what is yours?

Exercise 3: guess a number

- Write a JavaScript program where the program takes a random integer between 1 and 10, the user then prompted to input a guess number. If the user input matches the guess number, the program will display “good work”, otherwise, it will display “not matched”.

Hint: use alert() function to pop out text message.

Use Prompt() for user input. E.g.

```
var gnum = prompt('Guess the number between 1 and 10
```

Exercise 4: split string

- Write a JavaScript function to split a string and convert it to an array of words.

- alert(string_to_array(“ Monday is blue”) ;

Output: “Monday”, “is”, “Blue”

Exercise 5: show today's date

- Write a JavaScript function to display today's day in the following format:
- Today is Thursday. It is 5:30pm.

```
let  d = new Date();        //get the date
let  day = d.getDay();  // get the day of the date
let  hour = d.getHours(); // get hours
let  minuets = d.getMinuets(); //get minuets
```

You can use document.write or alert to display the message.

Consider makes the days into an array of strings. Again array starts with 0.

Exercise 5 (take home): show dates until Christmas

- Write a JavaScript to display how many dates until Christmas of 2018.
- You might find the following function useful:

```
var  d = new Date();        //get the date
var  n = d.getFulLyear();   // get the year of the date
var  d = d.getMonth();      // get the month of the
date
d.setFullYear(2020, 10,3); // Tue Nov 03 2020 11:17:37
GMT -500 (EST)
```

- Expected month from 0 to 11. 11 will be December, 12 will be start of the next year

- JS complete date references:
- http://www.w3schools.com/jsref/jsref_obj_date.asp

Form Element: <input>

```html
<!-- 'q' happens to be the name of Google's required paramter -->
<input type="text" name="q" value="Colbert Report" />
<input type="submit" value="Booyah!" />
```
HTML

Colbert Report | Booyah!

output

Input element is used to create many UI controls (an inline element that must be self-closed)

`name` attribute specifies name of query parameter to pass to server.

type can be `button, checkbox, file, hidden, password, radio, reset, submit, text`,…

`value` attribute specifies control's initial text.

Text fields: <input>

```html
<input type="text" size="10" maxlength="8" /> NetID <br />
<input type="password" size="16" /> Password
```

HTML

NetID

Password Log In

output

- input attributes: `disabled, maxlength, readonly, size, value`
- size attribute controls onscreen width of text field
- maxlength limits how many characters user is able to type into field

Form Elements: Text boxes: <textarea>

```
<textarea rows="4", cols="20">
Type your comments here.
</textrea>
HTML
```

- Initial text is placed inside textarea tag (optional)
- Required `rows` and `cols` attributes specify height/width in characters
- optional readonly attribute means text cannot be modified

Use InnerHTML to add text

```
<button onclick="addText();">Click me!</button>
<span id="output">Hello </span>
```
HTML

```
function addText() {
  var span = document.getElementById("output");
  span.innerHTML += " bro";
}
```
JS

Click me! Hello

output

- can change the text inside most elements by setting the `innerHTML` property

JavaScript Programming: Resources

Review programming basics: using variables, arrays, loops, if-statements, and functions

Go over some JavaScript tutorials - there are many great ones!

Mozilla's JavaScript Basics Tutorial

Basic interactive tutorials:
LearnJS

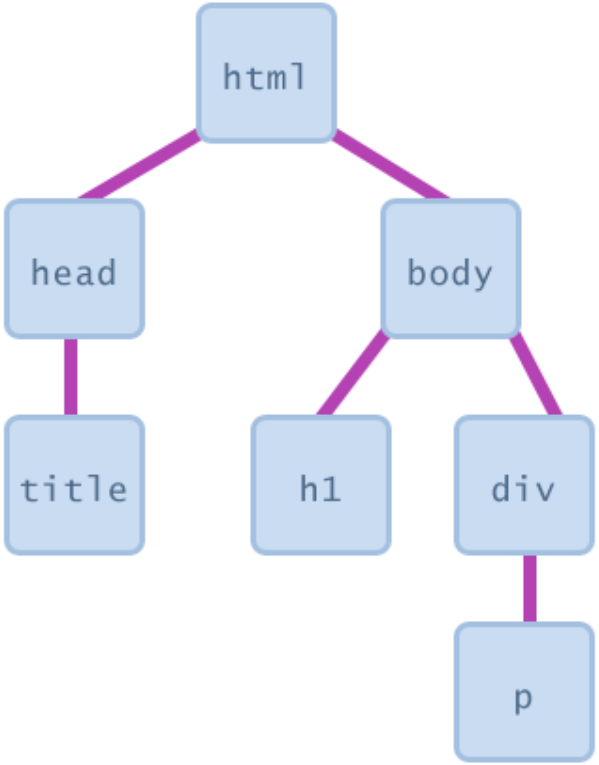
Check out cool examples of JavaScript on the web!

Dennis Music Video
JavaScript Memroy Game
Robby Leonardi Mario-style Resume
Species in Pieces
Rainbow Worm
OMFGDOGS

Document Object Model (DOM)

- *a set of JavaScript objects that represent each element on the page*

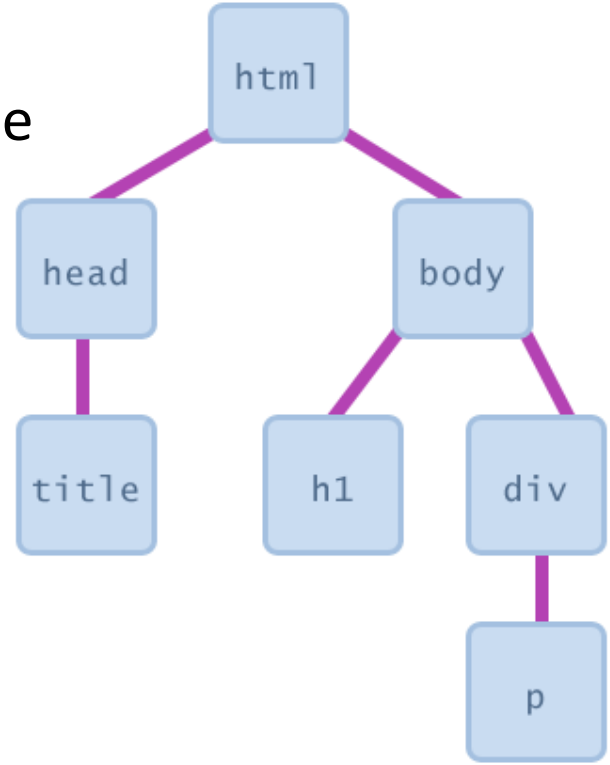
```
<html>
  <head>
    <title> ... </title>
  </head>
  <body>
    <h1> ... </h1>
    <div>
      <p> ... </p>
    </div>
  </body>
</html>
```



Document Object Model (DOM)

- *a set of JavaScript objects that represent each element on the page*

- each tag in a page corresponds to a JavaScript DOM object
- JS code can talk to these objects to examine elements' state
  - e.g. see whether a box is checked
- we can change state
  - e.g. insert some new text into a `div`
- we can change styles
  - e.g. make a paragraph red



Document Object Model (DOM)

How to get DOM elements in JS

1. Ask them by id: document.getElementById(…)
2. Query for them with CSS style selectors:

    document.querySelector(…)

    document.querySelectorall(…)

3. Make new ones! document.createElement(…)

Getting a DOM element in JavaScript

```
<p id="october"></p>
                                                    html
```

```
Let pTag = document.getElementbyId("october")
                                                    JS
```

What is inside a DOM object?

For starters, the HTML attributes. This is HTML:

```
<img src="images/puppy.png" alt= "A fantastic puppy photo">
                                                    HTML
```

Has a DOM object (let's call it puppyImg) with these two properties:

- puppyImg.*src* -- set by the browser to images/puppy.png
- puppyImg.*alt* -- set by the browser to "A fantastic puppy photo"

Accessing Properties of a DOM object (Example)

```
<p>See our <a href="sale.html" id="saleslink">Sales</a>
today!</p>
<img id="icon" src="images/borat.jpg" alt="Borat" />
<caption class="photo user-upload">Beauty.</caption>  HTML
```

```
let icon    = document.getElementById("icon");
let theLink = document.getElementById("saleslink");
let caption = document.querySelector("caption");
JS
```

Property	Description	Example
tagName	element's HTML tag	icon.tagName is "IMG"
className	CSS classes of element	caption.className is "photo user-upload"
src	URL target of an image	icon.src is "images/borat.jpg"
href	URL target of a link	theLink.href is "sale.html"

innerHTML property

- All DOM elements have a property called innerHTML that has the contents of the HTML tag as a string.

```
<ul id="dr-seuss">
    <li>Thing 1</li>
    <li>Thing 2</li>
</ul>
```

HTML

```
let elm = document.getElementById("dr-seuss");

//elm.innerHTML: "\n <li>Thing 1</li>
                        <li>Thing 2</li>  \n"
```

JS

Modifying DOM Elements (Example

```
<a id="fb-link" href=http://www.facebook.com> Facebook </a>
                                                    HTML
```

Before the JavaScript runs, we'd see:

Facebook

And after we run this JavaScript:

```
let link = document.getElementById("fb-link");
link.innerHTML = "MySpace is back in a really big way.";
```

We 'd see

My space is back in big way.

Unobtrusive JavaScript

Unobtrusive JavaScript

- What is unobtrusive JavaScript?
  - HTML with NO JavaScript code inside the tags
  - Uses the JS DOM to attach and execute all JavaScript event handlers.
- Allows separation of web site into 3 major categories.
  - Content(HTML) – what is it?
  - Presentation (CSS) – how does it look?
  - Behavior (JavaScript) – how does it respond to user interactions.

Quote: Breaking up is hard to do. But in web design, separation can be a good thing. Content, style, and behavior all deserve their own space.

Obtrusive Event Handler (bad)

```
<button onclick="okayClick();">OK</button>
```
HTML

```
// called when OK button is clicked

function okayClick() {
   alert("booyah");
}
```
JavaScript

OK

This is a bad style (HTML is cluttered with JS code)

Goal: remove all JavaScript code from HTML body

Solution: attach an event handler in JavaScript Code

```
let objectName.onevent = function()
```
JS

```
<button id="ok"">OK</button>
```
HTML

```
let okButton = document.getElementById("ok");
okButton.onclick = okayClick;
```
JS

It is legal to attach event handlers to elements' DOM objects in your JavaScript code.
Notice that you do not put parentheses after the function's name

This is better style than attaching them in the HTML.

When does my code run?

```
<html>
  <head>
    <script src="myfile.js" type="text/javascript"></script>
  </head>
  <body> ... </body>
</html>
```

HTML

```
let x = 3;
function f(n) { return n + 1; }
function g(n) { return n - 1; }
x = f(x);
```

JavaScript/myfile.js

- Your file's JS code **runs the moment** the browser loads the script tag
  - Any variables are declared immediately
  - Any functions are declared but not called, unless your global code explicitly calls them
- At this point in time, the browser has not yet read your page's body
  - None of the DOM objects for tags on the page have been created yet

A failed attempt at being unobtrusive

```
<html>
  <head>
    <script src="myfile.js" type="text/javascript"></script>
  </head>
  <body>
    <div><button <em>id="ok"</em>>OK</button></div>
    (... more html ...)        HTML
```

```
let btn = document.getElementById("ok");
btn.onclick = okayClick;          // this is bad: btn is null at this point
                                  JavaScript/myfile.js
```

- Problem: global JS code runs the moment the script is loaded.
- Script in head is processed before page's body has loaded.
- No elements are available yet or can be accessed yet via the DOM
- We need a way to attach the handler after the page is loaded.

The window.onload Event

```
function functionName() {
  // put code to initialize the page here
}

// instruct window to run the function when the page has loaded:
  window.onload = functionName;  // notice no () after function name
```

- There is a global event called window.onload event that happens once everything in the page is loaded.

- If you attach a function as a handler for window.onload, it will run at that moment.

Exercise 1: unobtrusive JS event

- Write a HTML with a button “OK”
- Write a unobtrusive JavaScript that when the user click the OK button, then the page pop out ”Booyah”.
- Check with your neighbor to see if your code is truly Unobtrusive.

- <button id=“ok”>OK </button>

Hint: wrote two functions

function pageLoad(){} // load the page and event handler

Function okayClick(){} // alert

Exercise 2: Modify the codes (movingimage.html) in the exercise folder in blackboard to make it unobtrusive

Modifying DOM Elements (Example

```
<a id="fb-link" href=http://www.facebook.com> Facebook </a>
                                                    HTML
```

Before the JavaScript runs, we'd see:

Facebook

And after we run this JavaScript:

```
let link = document.getElementById("fb-link");
link.innerHTML = "MySpace is back in a really big way.";
```

We 'd see

My space is back in big way.

JavaScript objects

Object	Properties	Methods
	car.name = Fiat	car.start()
	car.model = 500	car.drive()
	car.weight = 850kg	car.brake()
	car.color = white	car.stop()

```
var  car {
name: “Fiat”,
model: “500”,
color: “white”,
Weight: “850kg”};
```

```
// retrieval
car.name  //”Fiat”

car[name] // “Fiat”
```

Object: construction and retrieval

- *An object is a container of properties, where a property has a name and a value.*

Construction

```
Var flight {
 airline: "Oceanic",
Number: 815,
Departure:{
    IATA: "SYD",
    time: "2004-09-22 14:55",
    city: "Sidney"
};
Arrival: {
    IATA: "LAX",
    time: "2004-09-23 10:42",
    city: "Los Angeles"
}

};
```

Retrieval

```
flight.departure.IATAL // "SYD"

flight[airline] // "Oceanic"

// use || to fill in default
value

Var status = flight.status ||
"unknown";

flight.equipement  //undefined

flight.equipment.model //throw
"TypeError"
```

Object: update

- *A value in an object can be updated by assignment. If the property name already exist in the object, the property value is replaced:*

Construction

```
Var flight {
 airline: "Oceanic",
Number: 815,
Departure:{
    IATA: "SYD",
    time: "2004-09-22 14:55",
    city: "Sidney"
};
Arrival: {
    IATA: "LAX",
    time: "2004-09-23 10:42",
    city: "Los Angeles"
}

};
```

update

```
flight['airline'] = 'wow'
// if the object doesn't have
the property name, the object
is augmented:

flight.equipment = {
    model:'Boeing 777'
};

flight.status = 'overdue'
```

Object: reference

```
var Stooge = {
“first-name”:”Jeremy”,
“second-name”:”Howard”
}
var x = stooge;
x.nickname = ‘Curly’;
var nick = stooge.nickname;

//nick is ‘Curly’ because x and stooge
are references to the same object
```

Object: function construct with "this"

```
function  person(firstname,lastname,age,eyecolor)
{
    this.firstname=firstname;
    this.lastname=lastname;
    this.age=age;
    this.eyecolor=eyecolor;
}


// new instance
myFather=new person("John","Doe",50,"blue");
```

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/this

Object: adding method

```
myFather.name = function () {
    return this.firstName + " " + this.lastName;
};
```

Object: quiz

Which is the following is a valid way to create a direct instance of an object?

a. myObject.create ();

b. myObject = new Object;

c. myObject = new Object();

Object: quiz

- What is the the output of the following code after “alert”?

```
function person (firstname,lastname,age,eyecolor)
{
this.firstname=firstname;
this.lastname=lastname;
this.age=age;
this.eyecolor=eyecolor;
}

myFather = new person("John","Doe",50,"blue");
var x =myFather;
x.job = “Teacher”;
var profession = myFather.job;
alert(profession);
document.writeln("father's firstname is ",
  myFather.firstname,"<br>");
```

Using “reference”

- Add code to the code in the last slide and print:

- `my father 's nickname is Johny` using document.writeln

Demo: show info

In a JavaScript, create an object.
Create a property called “info” and assign a string.

Write a function (object method) myFunct() that alert the “info” value of the .info property to the browser.

you can say: “I am a new shinny object”

Create a instance of the method of the object by calling myFunct()

Create a button uses onClick to evoke the method. How do you display the “info” to the browser?

Enumeration of object

```
for (var key in object ) {
  print(object[key]);
}

var obj = {first: “prop1”, second:
  “propr2”, 3: “proper3”}

for (var key in obj) {
  s += key + “:” + obj[key] + “ “;
}
document.write(s);
```

Object: exercise 1

- Write a JavaScript program to list the property of the following sample object:

- var student = {
  Name: “Jenny Klein”
  Class: “ Senior”
  AUID: “ 31635”
  Hobby: “writing code”
  };

  Sample output: name, Class, AU ID, Hobby
  Hint: write a function to output the list of property. E.G. you can use string.push() to append to an empty array and then print out the array.

Object: exercise 2

- Write a JavaScript program to display the reading status (i.e. display book name, author name, and reading status) of the following books.

```
var library = [
  {
    title: 'Bill Gates',
    author: 'The Road Ahead',
    readingStatus: true
  },
  {
    title: 'Steve Jobs',
    author: 'Walter Isaacson',
    readingStatus: true
  },
  {
    title: 'Mockingjay: The Final Book of The Hunger Games',
    author: 'Suzanne Collins',
    readingStatus: false
  }];
```

Exercise: input number

- Create a simple UI input field
- Ask the user to input numbers between 1-10
- If the input number is not within the range,
- Tell them the input is not valid.
- If the input number is within the range,
- Tell them the input is valid.

Exercise: input number

- Create HTML element with ID:

```
<p>Please input a number between 1 and 10:</p>
<input id="numb”>
<button type="button"
onclick="myFunction(">Submit</button>
<p id="demo"></p>
```

- In JavaScript, create a function to validate the number:

a) Get the element of the input field by id.

```
var numb  =  document.getElementById(“num”).value
```

b) see if the number is a number IsNan() and whether it is between 1 and 10.

c) report the results to the browser using

```
document.getElementById("demo").innerHTML = text;
```

Take-home reading and exercise

Introduction to JavaScript (must read):

https://developer.mozilla.org/en-US/docs/Web/JavaScript/A_re-introduction_to_JavaScript

DOC model:

https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction