CSC435: Web Programming

Lecture 12: JavaScript: Objects

Bei Xiao

American University

March 2, Friday, 2018

Future lecture plan

Lectures	Content	homework
Feb 26 (today)	Unobtrusive JS OOP in JavaScript Functions	Homework 3 Due Thursday
March 1 (Friday)	OOP, Form validation.	
March 5 (Tuesday)	DOM, Events and Timers	
March 8 (Friday)	More JS, Mid-term review	
March 12-17	Spring break	Homework 4 (UI control is out
March 19	Mid-term exam	In-class exam (HTML, CSS, JS)
March 22	Ajax, Fetch, Json	Homework 4 is due
March 27	jQuery	

Activity Outline

- JavaScript Forms and events.
- JavaScript Objects
- OOP Exercises.

- Friday: Quiz 2 (in-class requires turn in on blackboard). 15 mins.

Take-home reading

Introduction to JavaScript (must read). Many readings are required!!

JavaScript and Browser, DOM:

http://eloquentjavascript.net/13_browser.html

http://eloquentjavascript.net/14_dom.html

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

Objects and arrays:
http://eloquentjavascript.net/04_data.html

JS skeleton

```html
<!-- in the <head> block -->
<script src="path/to/javascript/file.js"
type="text/javascrpt"></script>
```

HTML

```javascript
(function() {

  // set-up code that doesn't involve the DOM
  //   (e.g. setting up initial values, arrays, etc.)

  window.onload = function() {
    // phew! your code goes here
  };

  //function definitions go here

})();
```

JS

Homework 3: Grading

```
<body>
        <h1>Grade Calculator</h1>

        <div id="assignments">
            <div class="hw">
                HW <input class="earned" type="text" size="2" />
                 / <input class="max" type="text" size="2" />
            </div>

            <div class="hw">
                HW <input class="earned" type="text" size="2" />
                 / <input class="max" type="text" size="2" />
            </div>

            <div class="hw">
                HW <input class="earned" type="text" size="2" />
                 / <input class="max" type="text" size="2" />
            </div>
        </div>
```

Homework 3: JS

```
window.onload = function() {
 document.getElementById("compute").onclick = computeGrade;
 document.getElementById("clear").onclick = clearClick;
};




function computeGrade() {
 var earned = 0;
 var earnedInputs = document.querySelectorAll(".earned");
 alert(earnedInputs[0].value);

}

}
```

Homework 3: Grading

```
<body>
        <h1>Grade Calculator</h1>

        <div id="assignments">
            <div class="hw">
                HW <input class="earned" type="text" size="2" />
                 / <input class="max" type="text" size="2" />
            </div>

            <div class="hw">
                HW <input class="earned" type="text" size="2" />
                 / <input class="max" type="text" size="2" />
            </div>

            <div class="hw">
                HW <input class="earned" type="text" size="2" />
                 / <input class="max" type="text" size="2" />
            </div>
        </div>
```

Homework 3: getDocumentByID?

```
<!DOCTYPE html>
<html>
<body>

Name: <input type="text" id="myText" value="Mickey">

<p>Click the button to change the value of the text field.</p>

<button onclick="myFunction()">Try it</button>

</body>
</html>
```

```
<script>
function myFunction() {
  document.getElementById("myText").value = "Johnny Bravo";
}
</script>
```

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 “ strict” mode

```
“use strict”;

   ….. Your code ……
```

- Writing "use strict"; at the very top of your JS file turns on strict syntax checking:
- Shows an error if you try to assign to an undeclared variable

- Stops you from overwriting key JS system libraries

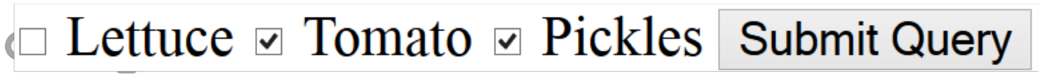
- Forbids some unsafe or error-prone language features

- You should *always* turn on strict mode for your code in this class!

Checkboxes:<input>

yes/no choices that can be checked and unchecked (inline)

```
<input type="checkbox" name="lettuce" /> Lettuce
<input type="checkbox" name="tomato" checked="checked" />
Tomato
<input type="checkbox" name="pickles" checked="checked" />
Pickles  HTML
```



- none, 1, or many checkboxes can be checked at same time
- when sent to server, any checked boxes will be sent with value on:
  - http://webster.cs.washington.edu/params.php?tomato=on&pickles=on
- use checked="checked" attribute in HTML to initially check the box

document.querySelectorAll

```
var inputs =
document.querySelectorAll("input[type='checkbox']"
);
for(var i = 0; i < inputs.length; i++) {
    inputs[i].checked = true;
}
```

```
var checked =
document.querySelectorAll("#checks
input[type='checkbox']:checked");
for (var i = 0; i < checked.length; i++) {
  str+=checked[i].value + " ";
}
```

document.querySelectorAll

```
elementList = document.querySelectorAll(seletors);
```

```
<p  class="example">A paragraph with class="example"</p>
HTML
```

```
// Get all <p> elements in the document
var x = document.querySelectorAll("p");
// Set the background color of the first <p> element
x[0].style.backgroundColor = "red";
JS
```

Return value: a list of the elements within the document (using depth-first pre-order traversal of the document's nodes) that match the specified group of selectors. The object returned is a NodeList.

document.querySelectorAll

```
elementList = document.querySelectorAll(seletors);
```

```
<h2 class="example">A heading with class="example"</h2>
<p class="example">A paragraph with class="example".</p>
<p class="example">Another paragraph with class="example".</p>        HTML
```

```
var x = document.querySelectorAll("p.example");
x[0].style.backgroundColor = "red";                                     JS
```

Return value: a list of the elements within the document (using depth-first pre-order traversal of the document's nodes) that match the specified group of selectors. The object returned is a NodeList.

docucment.querySelector

```
elementList = document.querySelector(seletors);
```

```
<h2 class="example">A heading with class="example"</h2>
<p class="example">A paragraph with class="example".</p>
<p class="example">Another paragraph with class="example".</p>        HTML
```

```
var el = document.querySelector(".example");                    el.
.style.backgroundColor = "red";
//In this example, the first element in the document with the class "example" is
returned:
```

Return value: A Element object representing the first element in the document that matches the specified set of CSS selectors.

Unobtrusive styling

```
function okayClick() {
  this.style.color = "red";        // <-- bad style
  this.className = "highlighted"; // <-- better style
}
```
JS

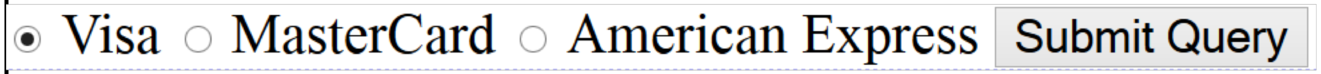
```
.highlighted{color:red;}
```
CSS

- Well-written JavaScript code should contain as little CSS as possible
- Use JS to set CSS classes/IDs on elements
- Define the styles of those classes/IDs in your CSS file
- We will discuss this in another class

Radio buttons: <input>

sets of mutually exclusive choices (inline)

```
<input type="radio" name="cc" value="visa" checked="checked" />
Visa
<input type="radio" name="cc" value="mastercard" /> MasterCard
<input type="radio" name="cc" value="amex" /> American Express
```
HTML



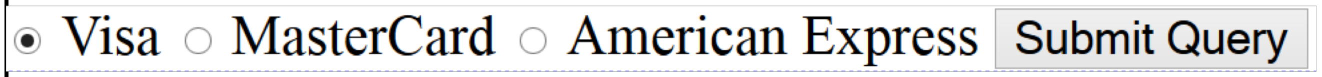
output

- grouped by name attribute (only one can be checked at a time)
- must specify a value for each one or else it will be sent as value on

  Set the “Visa” to be checked
  document.getElementById(”visa").checked = true;
  document.querySelector("input[namecc]:checked").value

Text Labels: <label>

```
<label>
  <input type="radio" name="cc" value="visa" checked="checked"> Visa
</label>
<label>
  <input type="radio" name="cc" value="mastercard"> MasterCard
</label>
<label>
  <input type="radio" name="cc" value="amex"> American Express
</label>
```
HTML


● Visa ○ MasterCard ○ American Express

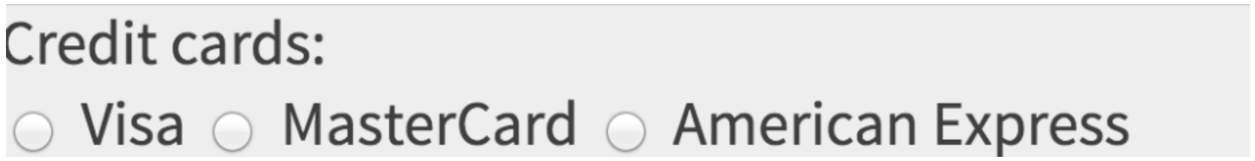
output

- Associates nearby text with control, so you can click text to activate control
- Can e used with check boxes or radio buttons.
- Label element can be targeted by CSS rule.

Grouping input: <fieldset> <legend>

group of input fields with optional caption

```
<fieldlset>
<legend> Credit Careds: </legends>
  <input type="radio" name="cc" value="visa" checked="checked"> Visa
</label>
<label>
  <input type="radio" name="cc" value="mastercard"> MasterCard
</label>
<label>
  <input type="radio" name="cc" value="amex"> American Express
</fieldset>                    HTML
```

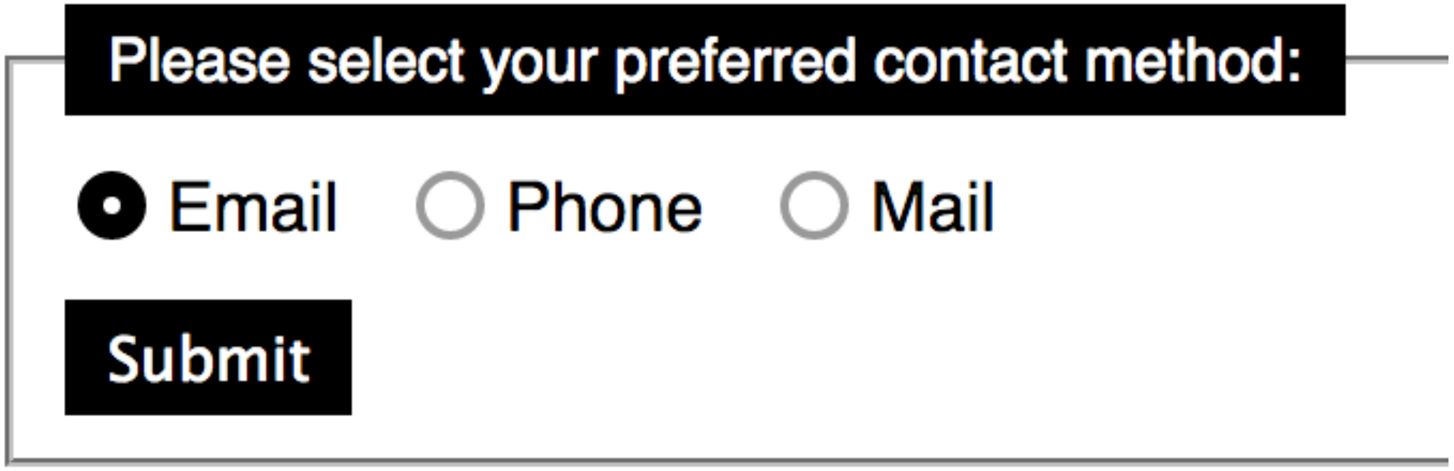

Credit cards:

○ Visa ○ MasterCard ○ American Express

Fieldset group related input fields, adds a border; legend supplies a caption.

Style Form Elements

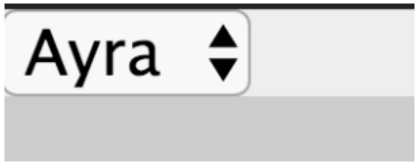
```
input {
  border: 2px solid #999;
}
Input: checked{
    border: 6px solid: black;}
```
CSS



Drop-down list: <select>,<option>

menus of choices that collapse and expand (inline)

```
<select name="favorite-character">
  <option>Rob</option>
  <option>John</option>
  <option selected="selected">Ayra</option>
  <option>Sansa</option>
</select>
```

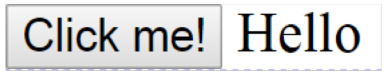


- Option element represents each choice
- Select optional attributes: disabled, multiple, size
- Optional selected attribute sets which one is initially chosen

The innerHTML property

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

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



output

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

simple computations Lecture 11 Exercise folder

- Write a dropdown menu use
- <input class="numberInput" type="text">
- <select id = ‘input1t”>
  <option value =“square”>square
  <option value =“cube”>cube
  <option value =“factorial”>factorial
  </select>

  Use <input id =“input1”> to ask users to input a number.

```
function pageLoad() {
   let input =
document.querySelector('.numberInput');
   let para = document.querySelector('p’);
   input.onchange = outputNumber;
}
 function outputNumber(){
    //Your code goes here
}
```

- To allow users to compute various functions of the number entered in the input area such as square, cube, or factorial.
- Display the results in the browser when they select the method.
- Hint: let input = document.querySelector(‘.numberInput’)
- let output = document.getElementByID(‘output’)

JavaScript Properties and methods

Properties: example

array.length

myString.length

Math.max

Methods: example

array.push()

string.toUpperCase()

JavaScript objects

Object	Properties	Methods

	car.name = Fiat
	car.model = 500
	car.weight = 850kg
	car.color = white
		car.start()
		car.drive()
		car.brake()
		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;