COS 426: Precept 1

JavaScript

Nora Willett
Outline

• Outline
  – Programming tips for JavaScript
  – Python server
  – GUI
JavaScript

• JavaScript is
  – an interpreted language.
  – object-based.
  – case sensitive.
  – widely used and supported.
  – accessible to the beginner.
Variables

• A variable can be:

  var a = 0;
  console.log(typeof a); // → number

  var a = "Hello world!";
  console.log(typeof a); // → string

  var a = ["Hello", "COS", 426];
  console.log(typeof a); // → object

  var a = true;
  console.log(typeof a); // → boolean
  // can also be null or undefined
Variables

- can be an array of object:

```javascript
var journal = [
    {events: ["work", "ice cream", "pizza", "running", "television"],
     squirrel: false},
    {events: ["weekend", "cycling", "break", "peanuts", "beer"],
     squirrel: true},
    ];
console.log(journal[0].events[1]); // → ice cream
for (var prop in journal[0]) {
    console.log(prop);
    console.log(journal[0][prop])
} // → events
// → ["work", "ice cream", "pizza", "running", "television"]
// → squirrel
// → false
Variable scope

- In JavaScript, instead of braces, functions are the only things that create a new scope

```javascript
var a = 1;
{
  var a = 2;
}
console.log(a); // → 2
```

```javascript
var a = "outside";
var f = function() {
  var a = "inside f";
};
f();
console.log(a); // → outside
```
Function variables

- Function variables act as names for a specific piece of the program
  
  ```javascript
  var Sqr = function( x ) { return x * x; };
  ```

- Function Declaration
  
  ```javascript
  function sqr( x ) { return x * x; }
  ```

  + not part of regular top-to-bottom flow of control
  + can be used by all the code
Special functions

- `alert()` to display a message box
- `confirm()` to display a confirmation box
- `prompt()` to display a prompt box
- `open()` to open a new window
- `close()` to close a window
- `write()` write a string to the Web page
- `console.log()` outputs a message to the Web Console
Debugging

- `console.log()` is your friend
- `console.log()` is your friend
- `console.log()` is your friend
- `console.log()` is your friend
- `console.log()` is your friend
- `console.log()` is your friend
- `console.log()` is your friend
- `console.log()` is your friend
- `console.log()` is your friend
### == VS ===

### will return false for them all, however == will:

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>' ' == '0'</td>
<td>false</td>
</tr>
<tr>
<td>' ' == 0</td>
<td>true</td>
</tr>
<tr>
<td>0 == '0'</td>
<td>true</td>
</tr>
<tr>
<td>false == 'false'</td>
<td>false</td>
</tr>
<tr>
<td>false == '0'</td>
<td>true</td>
</tr>
<tr>
<td>false == undefined</td>
<td>false</td>
</tr>
<tr>
<td>false == null</td>
<td>false</td>
</tr>
<tr>
<td>null == undefined</td>
<td>true</td>
</tr>
<tr>
<td>' \t\r\n ' == 0</td>
<td>true</td>
</tr>
</tbody>
</table>
Objects

• PROTOTYPE

```javascript
Array.prototype.myUpperCase = function() {
    for (i = 0; i < this.length; i++) {
        this[i] = this[i].toUpperCase();
    }
}

var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.myUpperCase();
document.write(fruits);
// → BANANA, ORANGE, APPLE, MANGO
```
Objects

```javascript
var testOne = function () {};
testOne.prototype = function () {
    var me = {}, privateVar = 2;
    me.aMethod = function () {
        return privateVar;
    };
    me.publicVar = "foo bar";
    me.bMethod = function () {
        return this.publicVar;
    };
    return me;
};
for (var i = loopCount; i>0; i--)
{
    new testOne();
}

loopCount=1,000,000:
TestOne takes 17ms, while test Two test 43ms. WHY?

var testTwo = function () {
    var me = {}, privateVar = 2;
    me.aMethod = function () {
        return privateVar;
    };
    me.publicVar = "foo bar";
    me.bMethod = function () {
        return this.publicVar;
    };
    return me;
};
for (var i = loopCount; i>0; i--)
{
    new testTwo();
}
```
JavaScript Demo
JavaScript Help

http://www.w3schools.com/js/
JavaScript: The Good Parts

Most programming languages contain good and bad parts, but JavaScript has more than its share of the bad, having been developed and released in a hurry before it could be refined. This authoritative book scrapes away these bad features to

More »
Simple HTTP server

• Open up a terminal and type:
  – $ cd /home/yourdir
  – $ python -m SimpleHTTPServer

• That's it! Now your http server will start in port 8000. You will get the message:
  – Serving HTTP on 0.0.0.0 port 8000

You can access it via
  http://127.0.0.1:8000/yourhtml.html
Dat.Gui

- A lightweight graphical user interface for changing variables in JavaScript.
- Link for tutorial (no need to learn how to use it)
  
  http://workshop.chromeexperiments.com/examples/gui
QUESTIONS?