CS 3710 Vocational Languages

October 15, 2012

JavaScript

- JavaScript is a language designed for web browser.
  - Developed by Brendan Eich at Netscape in 1995
- The language is related to Java in name only
  - The name was part of a marketing deal between Sun and Netscape.
- There are various implementations available
  - Spidermonkey interactive shell interface
  - Rhino: http://www.mozilla.org/rhino/
  - Browser JavaScript

JavaScript Language

- It is a scripting language:
  - JavaScript is embedded in HTML and interpreted to produce outputs for the browser
- It is an interpreted (vs. compiled) language
  - Loading a program runs the program
- It is an un-typed or loosely typed (vs. strong or statically typed) language
  - No need to specify the type of a variable in the program.
  - The interpreter resolves the variable types and makes necessary adjustments to execute the program.

This Course

- We’ll talk about the language concepts:
  - C-style syntax
  - Dynamic typing
  - First-class functions, inner functions
  - Objects without classes
  - Prototypal inheritance

- We won’t focus on:
  - HTML, DOM, CSS
  - COMP 3715 – Network Computing with WEB Applications
### JavaScript Language Design

- Functions based on Lisp/Scheme
  - First-class higher-order functions
    ```javascript
    function addx(x) {
      return function addy(y) { return x+y;
    }
    ```
- Objects based on Smalltalk/Self
  ```javascript
  var pt = {x: 10, move:function(dx)
  {this.x += dx}}
  ```

### Using JavaScript

- JavaScript can be embedded in HTML documents – in the `<head>`, in the `<body>`, or in both.
  ```javascript
  <html><body><pre>
  <script>
  document.writeln("Hello World!");
  </script></pre></body></html>
  ```
- JavaScript can reside in a separate page
  ```html
  <html><body><pre>
  <script src="lect1.js"></script></pre></body></html>
  ```

### Running JavaScript 1.6

- Firefox 2.0+
- Google Chrome 1.0+
- Safari 3.0+
- Opera 9.5+
- Develop applications that run on popular web browsers.

### JavaScript Basics

- JavaScript is case sensitive
  - HTML is not case sensitive; onClick, ONCLICK, … are HTML
- Statements terminated by returns or semi-colons (;)
  - `x = x+1;` same as `x += 1`
  - Semi-colons can be a good idea, to reduce errors
- "Blocks"
  - Group statements using `{…}`
  - Does not introduce a separate scope, unlike other languages
- Variables
  - Define a variable using the `var` statement
  - Define implicitly by its first use, which must be an assignment
    - Implicit definition has global scope, even if it occurs in nested scope?
A small set of basic type

- **Boolean**
  - Two values: true and false
- **Number**
  - 64-bit floating point, similar to Java double and Double
  - No integer type
  - Special values NaN (not a number) and Infinity
- **String**
  - Sequence of zero or more Unicode characters
  - No separate character type (just strings of length 1)
  - Literal strings using ’ or ” characters (must match)
- **Special values**
  - null and undefined
  - typeof(null) = object; typeof(undefined) = undefined

JavaScript Syntax – Variables and Literals

- **Declaration**
  - Explicit: var i=12;
  - Implicit: i=12;
- **Variable Scope**
  - Global
    - Declared outside functions
    - Any variable implicitly defined
  - Local
    - Explicit declarations inside functions

JavaScript Syntax - Variables and Literals

- Variables can hold any valid type of value:
  - Number ... var a = 7;
  - Boolean ... var b = true;
  - Function ... [Discussed Later]
  - Object ... [Discussed Later]
  - Array ... var arr= new Array();
  - String ... var s = “abc”;
  - and can hold values of different types at different times during execution,

JavaScript Operator

<table>
<thead>
<tr>
<th>Comparison Operators</th>
<th>Arithmetic/ assignment Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;</td>
<td>+ concatenation for string</td>
</tr>
<tr>
<td>&lt;</td>
<td>-</td>
</tr>
<tr>
<td>&lt;=</td>
<td>/</td>
</tr>
<tr>
<td>&gt;=</td>
<td>%</td>
</tr>
<tr>
<td>==</td>
<td>=</td>
</tr>
<tr>
<td>===</td>
<td>==</td>
</tr>
<tr>
<td>!=</td>
<td>!</td>
</tr>
<tr>
<td>!==</td>
<td>!=</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>&amp;&amp;</td>
<td>-</td>
</tr>
</tbody>
</table>

- == and !== are type-converting comparison.
- === and !== are strict comparison, where the compared variables have to be the same type and the same value to be true.
- NaN is not equal to anything, including NaN.
- Positive and negative zeros are equal to one another.
- Two objects are strictly equal if they refer to the same Object.
- Null and Undefined types are == (but not ===).
Control and Loop Statements

- **if statement**
  
  ```javascript
  if ( boolean statement ) {
    ...
  } else { ...
  }
  ```

- **switch statement**
  
  ```javascript
  switch ( a ) {
    case 1: //statement
    case "two": //statement
    default: //statement
  }
  ```

- **for**
  
  ```javascript
  for(var i = 0; i < 10; i++)
  ```

- **do {} while ()**

- **while() {}**


JavaScript Functions

- Declarations can appear in function body
  - Local variables, “inner” functions

- Parameter passing:
  - Basic types passed by value, objects by reference

- Call can supply any number of arguments, the extra arguments are ignored
  - `functionname.length`: # of arguments in definition
  - `functionname.arguments.length`: # args in call

- Closures and Curried functions
  ```javascript
  functionCurAdd(x){return function(y){return x+y}};
  ```