Introduction to JavaScript

Outline

- What is JavaScript?
- Using JavaScript
  - Adding JavaScript to HTML
  - Sample Usages
- JavaScript Basics
- JavaScript Functions

Introduction

- JavaScript is a programming language for use in HTML pages
- Invented by Brendan Eich in 1995 at Netscape Corporation (LiveScript)
- JavaScript is a full-featured programming language (has nothing to do with Java!)
- JavaScript programs are run by an interpreter built into the client's web browser (not on the server)

JavaScript Uses

- dynamically modifying an HTML page
- responding to user action
- validating user input
- communicate with server
- storing/restoring data
- animation and drawing
- playing audio and video, …

Outside Web Pages

- Gadgets
  - Desktop apps, Browser extensions
- Tools
  - Adobe Acrobat, Open-Office, Flash
- Server-side scripting
  - Node.js
- Game development
  - DX-Studio, Re-Animator

Using JavaScript

Inline Script

- JavaScript can be inserted into HTML pages by using the <script> tag

```
<!DOCTYPE html>
<html>
<body>
.
.
<script>
   document.write("Hello World!");
</script>
.
.
</body>
</html>
```

Where to Put Inline Scripts

- You can have any number of scripts
- Scripts can be placed in the <body> or in the <head>
  - In the <head>, scripts are run before the page is displayed
  - In the <body>, scripts are run as the page is displayed
- In the <head> is the right place to define functions and variables that are used by scripts within the <body>

External Scripts

- Scripts can also be loaded from an external file
- Useful for complicated scripts or set of functions that are used in different pages

```
<script src="mine.js"></script>
```

Sample Usages

- Responding to Events

```
<button type="button" onclick="alert('Hi!')">
Click Here!</button>
```

- Modifying HTML Content

```
element = document.getElementById('box');
element.innerHTML = 'Hello World!';
```

- Modifying HTML Style

```
element = document.getElementById('box');
element.style.color = '#f30';
```

JavaScript Basics

Statements

- Each statement is optionally ended with a semicolon
  - It is good practice to add semicolons
- Statements can be grouped together into blocks, enclosed by { }
- Comments are delimited with // and /* */ as in Java and C++

Variables

- JavaScript has variables that you can declare with the optional var keyword
- Variables declared within a function are local to that function
- Variables declared outside of any function are global variables
  - Note: variables declared without var are made global

```
var str = "Hello";
```

Basic Data Types

- string
- number
- Boolean
- null
- undefined

Operators

- JavaScript has Java/C-like operators
  - Arithmetic (+, - , *, /, %)
  - Assignment (=, +=, -=, *= /=, %=, ++, --)
  - Comparison (<, >, <=, >=, ==)
  - Logical (&&, ||, !)
- Notes:
  - + also does string concatenation
  - === checks both value and type

Type Conversion

- In expressions involving a string, the + operator and a number, numbers are converted to strings

```
res = 'The total is ' + 12;
res = 12 + 'is the total';
```

- With other operators, strings are converted to numbers

```
a = 20 - '32';    // -12
b = 1 * '5.4';    // 5.4
c = 3 * 'a2';     // NaN
```

Type Conversion (cont'd)

- The following values are treated as false
  - null
  - undefined
  - 0, NaN
  - ' ' (empty string)
- Anything else is treated as true

Control Structures

- JavaScript has C-like structures
- Conditionals
  - if, else
  - switch, case
- Loops
  - for, while
  - (can break and continue)

Arrays

- Arrays are indexed from 0
- Special version of for works with arrays

```
var colors = ['red', 'geen', 'yellow'];
for (var i in colors) {
   document.write('<div style="background-color:'
                  + colors[i] + ';">'
                  + colors[i] + '</div>\n');
}
```

JavaScript Functions

Functions

- You can define functions using the function keyword
- Functions can return a value using the return keyword (or return undefined by default)

```
function test(n) {
   var a = 2 / n;
   return a;
}
```

Function Arguments

- JavaScript is very flexible with function arguments
- A function can be called with more or less arguments than the number of declared parameters
- Too few arguments: leaves parameters undefined

```
function show(x, y) {
  document.write(x + '\n');
  document.write(y);
}
show('hello'); // prints hello undefined
```

Function Arguments

- All the arguments to a function can be accessed through the arguments pseudo-array

```
function show() {
  for (var i = 0; i < arguments.length; i++) {
    document.write(arguments[i]);
  }
}

show('a', 'b', 'c', 'd', 42);
```

Default Parameters

- Here is a common idiom for making default parameter values

```
function show(x, y) {
  x = x || '';
  y = y || 0;
  document.write(x + '=' + y);
}
```

- If a is not false then **a || b** evaluates to **a**
- Otherwise **a || b** evaluates to **b**

Anonymous Functions

- Functions don't have to have names
- Functions can be assigned to variables or object properties

```
function f() {};

var f = function() {};  // equivalent to above
```

eval

- The eval function evaluates a string as if it were JavaScript code
- The evaluation environment is the same as that in which eval is called

```
var str = '2 * 4 + 5';
var x = eval(str);

code = 'alert(1)';
eval(code);
```

Simple User Interaction

- JavaScript has some built-in functions providing simple user interaction
  - alert(msg): alerts the user that something has happened
  - confirm(msg): asks the user to confirm (or cancel) something
  - prompt(msg, default): asks the user to enter some text

```
alert('The email is not correct!');

confirm('Are you sure you want to do that?');

prompt('Enter you name');
```

References

- W3Schools
  - http://www.w3schools.com/js
- Eloquent JavaScript
  - http://eloquentjavascript.net/
- Internet Programming by Pat Morin
  - http://cg.scs.carleton.ca/~morin/teaching/2405/
- Dr. Zarrabi-Zadeh Slides