CHAPTER 6
JAVASCRIPT
PART 1
OVERVIEW OF JAVASCRIPT

JavaScript is an implementation of the ECMAScript language standard and is typically used to enable programmatic access to computational objects within a host environment.
OVERVIEW OF JAVASCRIPT

• Originally developed by Netscape, as LiveScript

• Became a joint venture of Netscape and Sun in 1995, renamed JavaScript

• JavaScript and Java are only related through syntax
  – JavaScript is dynamically typed
  – JavaScript’s support for objects is very different

• JavaScript be embedded in many different things, but its primary use is embedded in HTML documents
OVERVIEW OF JAVASCRIPT

• JavaScript is a scripting language that enables web developers/designers to build more functional and interactive websites.

• **JavaScript** is a subset of Java

• Differences between Java and JavaScript:
  – Java is a **compiled language**
  – JavaScript is an **interpreted language**
Hey, I need your help with something. You know JavaScript right?

No, I know Java.

Huh, what's the difference?

Think of it like the difference between English and American.

One is an ancient language, slow to change and stands alone.

And the other?

It's easier to understand, but has no class.

03/06/2008

Written by Terry Richards

Drawn by John Batlis
OVERVIEW OF JAVASCRIPT

• The JavaScript Programming Language
  – JavaScript
  • Two formats:
    – Client-side
      » Program runs on client (browser)
    – Server-side
      » Program runs on server
      » Proprietary to web server platform
WHAT JAVASCRIPT CAN DO?

- **JavaScript gives HTML designers a programming tool**
- **JavaScript can put dynamic text into an HTML page**
  
  A JavaScript statement like this: `document.write("<h1>" + name + "</h1>")` can write a variable text into an HTML page
- **JavaScript can react to events**
  
  A JavaScript can be set to execute when something happens, like when a user clicks on an HTML element
- **JavaScript can read and write HTML elements**
  
  A JavaScript can read and change the content of an HTML element
- **JavaScript can be used to validate data**
  
  A JavaScript can be used to validate form data before it is submitted to a server-save the server from extra processing
JavaScript Syntax

• The HTML `<script>` tag is used to insert a JavaScript into an HTML page.

```html
<html>
<body>
<script type="text/javascript">
document.write("JavaScript is not Java");
</script>
</body>
</html>
```

The `<script>` tags tell the browser to expect a script in between them.

This part that writes/display the actual text
Where to put your scripts?

1. Scripts in `<head>`.
2. Scripts in `<body>`
3. Using an External JavaScript
4. In both the body and the head section.
Where to put your scripts? <head>.

<html>

<head>
<script type="text/javascript">
function message()
{
alert("This alert box was called with the onload event");
}
</script>
</head>

<body onload="message()">
</body>
</html>
Where to put your scripts? <body>.

<html>
<body>
<script type="text/javascript">
document.write("JavaScript is not Java");
</script>
</body>
</html>
Where to put your scripts? External javascript

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

JavaScript external file with a .js file extension.
Where to put your scripts? <head> and <body>

- <html>
  <head>
  <script type="text/javascript">
  function message() {
  alert("This alert box was called with the onload event");
  }
  </script>
  </head>

  <body onload="message()">
  <script type="text/javascript">
  document.write("This message is written by JavaScript");
  </script>
  </body>

  </html>
Writing Output to the Web Page

- To write text to a Web page, use the following JavaScript commands:

  ```javascript
  document.write("text");
  
or
  document.writeln("text")
  ```

Where `text` is the content to be written to the page. The `document.write()` and `document.writeln()` methods are identical, except that the `document.writeln()` method preserves any line breaks in the text string.
Rules for variable names:

- Variable names are case sensitive
- They must begin with a letter or the underscore character
- You can create a variable with the `var` statement:
  ```javascript
  var strname = some value
  ```
- You can also create a variable without the `var` statement:
  ```javascript
  strname = some value
  ```
- Assign a Value to a Variable
  ```javascript
  var strname = "Hege"
  Or like this:
  strname = "Hege"
  ```
The script above declares a variable, assigns a value to it, displays the value, changes the value, and displays the value again.
Working with Variables and Data

• JavaScript variable types:
  – Numeric variables- 1,2,3,
  – String variables- ”hello”
  – Boolean variables - true and false
  – Null variables- no value at all

• You must **declare** a variable before using it
Working with Expressions and Operators

• **Expressions** are JavaScript commands that assign values and variables

• **Operators** are elements that perform actions within expressions

  1. **Arithmetic Operators** (+, -, *)
  2. **Assignment Operators** (=, +=)
  3. **Comparison Operators** (==, >, <)
  4. **Logical/boolean Operators** (&&, !)
  5. **String Operators**
     
     ```javascript
     txt1="What a very"
     txt2="nice day!"
     txt3=txt1+txt2
     ```
Creating JavaScript Functions

• A **function** is a series of commands that perform an action or calculates a value
• A **function name** identifies a function
• **Parameters** are values used by the function

**Syntax**

```javascript
function functionname(var1,var2,...,varX) {
    some code
}
```
Example javascript function

```html
<html>
<head>
<script type="text/javascript">
<!--
function displayMessage(firstName) {
    alert("Hello " + firstName + ", miss me!, call me")
}
//-->
</script>
</head>

<body>
<form>
First name:
<input type="input" name="yourName" />
<input type="button"
    onclick="displayMessage(form.yourName.value)"
    value="Display Message" />
</form>

</body>
</html>
```
Working with Conditional Statements

• **Conditional statements** are commands that run only when specific conditions are met

• Conditional statements require a **Boolean expression**
  – you need one of the following operators to create a Boolean expression:
    • Comparison operator
    • Logical operator
    • Conditional operator
CONDITIONAL STATEMENTS

• Conditional statements in JavaScript are used to perform different actions based on different conditions.

• In JavaScript we have three conditional statements:
  – *if statement* - use this statement if you want to execute a set of code when a condition is true
  – *if...else statement* - use this statement if you want to select one of two sets of lines to execute
  – *switch statement* - use this statement if you want to select one of many sets of lines to execute

• If and If...else Statement
  – You should use the if statement if you want to execute some code if a condition is true.
JAVASCRIPTS LOOPING

• In JavaScript we have the following looping statements:
  – `while` loops through a block of code while a condition is true
  – `do...while` loops through a block of code once, and then repeats the loop while a condition is true
  – `for` run statements a specified number of times
JAVASCRIPTS ARRAY

An **array** is an ordered collection of values referenced by a single variable name

```javascript
var variable = new Array (size);
```

Where **variable** is the name of the array variable and **size** is the number of elements in the array

```javascript
var faq = new Array(3)
faq[0] = "What are JavaScript arrays"
faq[1] = "How to create arrays in JavaScript?"
faq[2] = "What are two dimensional arrays?"
```
JavaScript Popup Boxes

Alert

```javascript
alert("Hey, remember to tell your friends about Quackit.com!");
```

Confirm

```javascript
confirm("Are you sure you want to delete the Internet?");
```

Prompt

```javascript
prompt("Message","Default response");prompt('Please enter your favorite website', 'Quackit.com');
```
event handler

• JavaScript are often event-driven. That is, a piece of codes (called event handler) fires in response to a certain user's or browser's action, which generates an event.

• The commonly-used events are:
  1. **click**: generated when the user clicks on an HTML element.
  2. **mouseover, mouseout**: generated when the user positions the mouse pointer inside/away from the HTML element.
  3. **load, unload**: generated after the browser loaded a document, and before the next document is loaded, respectively.
Objects

• JavaScript is object-oriented, although it does not support all the OO features, so as to keep the language simple.
• JavaScript's OO is prototype-based, instead of class-based.

• A **class-based** OO language (such as Java/C++/C#) is founded on concepts of *class* and *instance*. A class is a blue-print or prototype of things of the same kind. An instance is a particular realization (instantiation, member) of a class. For example, "Student" is a class; and "Tan Ah Teck" is an instance of the "Student" class. In a class-based OO language, you must first write a class definition, before you can create instances based on the class definition.

• A **prototype-based** OO language (such as JavaScript) simply has objects (or instances). It uses a *prototype* as a template to get the initial properties of a new object. In JavaScript, you do not have to define a class explicitly. Instead, you define a constructor method to create objects (or instances) with a set of initial properties. Furthermore, you can add or remove properties of an object at runtime, which is not permitted in class-based OO languages.
END OF PART 1