Javascript and HTML Forms

Scripting languages such as Javascript and VBScript are used to add functionality to web pages. They can be either included in the web page itself or stored in a separate file. In the latter case, the web page must include a link to the file.

Scripts are downloaded with the web page, unlike Java applets and servlets. Applets are not downloaded until the browser reaches the <applet> tag. Servlets always execute on the server, not the user's computer. But Javascript is downloaded immediately and is executed on the user's computer, thus saving time that would otherwise be used for repeated connections to the server.

Javascript is not Java, but it shares some syntax with Java and C. It is not strongly typed, so variables do not have to be declared, although they may be. Also statements do not require a terminating semi-colon, but again these may be included. If two statements are on the same line, they must be separated by a semi-colon. Also like Java, Javascript is case sensitive.

A Simple Example

The following example simply writes the word "Hello" into the box when the user clicks the button. It does not send anything to the server, so the form includes neither an action nor a method attribute.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
    <title>Javascript Hello Example</title>
    <script language="Javascript">
    <!-- sayHello writes the word "Hello" into the box when the button is clicked.
    function sayHello ()
    {
        HelloForm.Hello.value = "Hello";
    }
    //--></script>
</head>
<body>
    <form name = "HelloForm">
        <input name="Hello" type="text" value="" size = 10>
        <input type="button" value="Click" onClick = "sayHello ()">
    </form>
</body>
</html>
```

Scripts may be placed either in the head or body of the page. It is somewhat more common to put them into the head, since it is downloaded first. Also this separates the script from the rest of the page. If the script is external, the head would be as follows:

```
<head>
    <script language = "Javascript" src = "helloScript.js"> </script>
    <title>Javascript Hello Example</title>
</head>
```

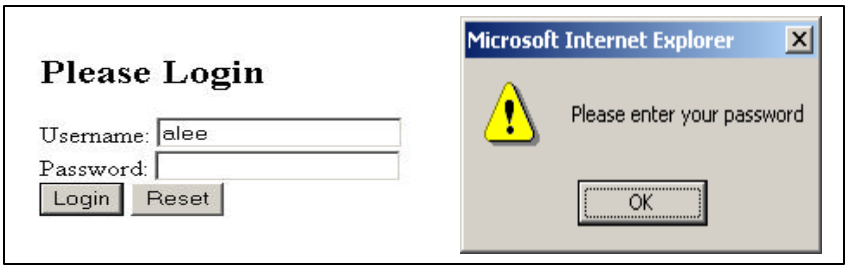
The script file, *helloScript.js*, would contain only the following code:

```
<!-- sayHello writes the word "Hello" into the box when the button is clicked.
function sayHello ()
{
    HelloForm.Hello.value = "Hello";
}
//-->
```

Functions

Functions in Javascript are very simple. They may either perform an action, such as the one above, or return a value. A useful example of the latter is found in the next example. It checks a login form to see that all the boxes have been filled in. This check is performed by the user's browser rather than on the server. This saves a significant amount of time when the form contains a number of boxes. This example shows an *alert* message and returns false, if the box is empty. The false value prevents the action from taking place. However, if both boxes contain text, the value true is returned and the request is sent to the server.

```
<html>
<head>
    <script language = "Javascript">
    <!-- CheckForm displays an alert if a box has not been filled in.
        function CheckForm ()
        {
            if (LogonForm.username.value == "")
            {
                alert ("Please enter your username");
                return false;
            }
            if (LogonForm.password.value == "")
            {
                alert ("Please enter your password");
                return false;
            }
            return true;
        }
    //-->
    </script>
    <title>Login Form</title>
</head>
<body>
    <h2>Please Login</h2>
    <form name = "LogonForm" method = "post" action = "http://localhost/servlet/user.Logon">
        Username: <input type = "text" name = "username" /> <br />
        Password: <input type = "password" name = "password" /> <br />
        <input type = "submit" value = "Login" onClick = "return CheckForm ()"/>
        <input type = "reset" value = "Reset" />
    </form>
</body>
</html>
```



Functions can also have parameters. These are not typed, but otherwise they are similar to those in Java methods. An example would be if there were two forms on the same page and the function had to distinguish between them. The function heading might look like

```
function CheckForm (formName)
```

and the submit tag

```
<input type = "submit"  value = "LogonForm" onClick = "return CheckForm(LogonForm)" />.
```

Objects

Javascript also has objects. The one in the next example is a Date object. Like in Java, it must be instantiated before it can be used. There are a number of accessor methods included with this object that get parts of the date, such as the year, month, day, hours, minutes and seconds. The values are those that are stored in the user's computer. If the computer time is accurate, the results of the function will be also.

```
<script language="Javascript">
<!--Example that writes the current date on the page>
      date = new Date ()
      month = date.getMonth ()
      month = month + 1                  // Months in Javascript begin with January as 0.
      day = date.getDate ()
      year = date.getYear ()
      document.write ("Month ", month)
      document.write ("<br />Date ", day)
      document.write ("<br />Year ", year)
      hour = date.getHours ()
      if (hour > 12) hour = hour – 12     // Afternoon hours are returned between 13 and 24.
      minutes = date.getMinutes ()
      seconds = date.getSeconds ()
      document.write ("<br>Time ", hour, ":", minutes, ":", seconds)
//-->
</script>
```

The *document* mentioned in the script is the web page. This will write the date and time on the web page. As with any html page you have to use tags such as
 or <p> to indicate a line or paragraph break.

Numerical Data

Text boxes contain text, i.e. strings, and not numbers. If the contents of a box are to represent numbers, they must first be *parsed* before they can be used in a calculation. As in Java, the method to use is

parseInt. If the text box, weightBox, contains an integer, the following can be used to change the contents from a string to a number:

```
weight = parseInt (weightBox.value);
```

After a calculation, the answer may turn out to be a double with a number of decimal places. The following will round these to two decimal places:

```
mean = Math.round ((sum / count)* 100) / 100;
```

Occasionally a box may be empty or an answer may not exist. Then Javascript displays *NAN*. This stands for *Not a Number*. Checking a box for empty before using it is prudent.

Arrays

Javascript has arrays that like those in Java begin with index 0. They are instantiated with *new*.

```
var arrayName = new Array ();
```

They do not have a fixed length, so the number of objects in the array can be found using

```
arrayName.length;
```

Arrays can also be filled initially using parentheses:

```
var prices = new Array (2.89, 1.50, 1.00, 4.95, 3.50);
```

The contents of an array are accessed the same as in Java using square brackets.

```
prices [0] = 3.75;
```

Creating New Windows

Javascript can also create a new window with a specified html document in it. This document can either be on your local disk or on the Internet. The following example creates a new window for a date-time page. The code is straightforward. You have to tell the browser where to find the html page and how large the window should be.

The following shows a sample script that opens a new window. The first attribute tells where to find the page to open. The second says that it should be opened in a window. The last two give the width and height of the new window (in pixels). Notice that the width and height are enclosed by a single set of double quotation marks since they combine to make up a single attribute.

```
<script language="Javascript"><!--
    timeWindow = window.open ("date-time.html", "Window", "width=200, height=100")
//--></script>
```

The new window so created appears below. It is a functioning window and can be used like any other html page.