In this module you will be introduced to the concepts of Objects, the DOM, and events and how you can use them.

4 Objectives

1. Objects
2. The "new" operator
3. The Document Object Model
4. Arrays
5. Events

    i. onClick
    ii. onFocus
    iii. onChange
    iv. onBlur
    v. onLoad
    vi. onUnload

Objects

An object is a collection of variables (parameters) and functions (methods). The syntax for using an object is: object.parameter, or object.method(). A string is an object in JavaScript and has several methods and parameters.

```
var StringVar = "This is a string of characters.";

var x = StringVar.length;  //the same as LEN in true basic; len in Python
StringVar = StringVar.toUpperCase();  // change to lower case is StringVar.toLowerCase();
```

If this code is executed StringVar is a new variable with the value of "This is a string of characters.". The x variable is set to the length of the StringVar, in this case 31. Length is a property of the string object. If you count the characters (letters, spaces, and punctuation) between the quotes you will see that it adds up to 31. The toUpperCase() method converts all of the alpha characters in the string to upper case i.e. "THIS IS A STRING OF CHARACTERS.".

JavaScript has the following objects built into the language: String, Math, Date and Array. The string object has a number of methods for manipulating strings as demonstrated above with the toUpperCase() method. The math object is a collection of methods and properties for performing mathematical operations like: min(), max(), sin(), cos(), etc.. The date object is a collection of methods for working with dates and time. The array object allows programmers to create collections of data.

The new operator

Objects and arrays **cannot** simply be typed into your JavaScript programs! They must be created. We use the "new" operator to create a new instance of an object or an array. To put that another way the new operator creates a copy of an existing object or an array structure and assigns the name you want to it.

The generic syntax is:

var o = new *Object*(); // the Object is ***D***ate, ***S***tring, ***M***ath, or ***A***rray

Since objects are made up of methods (functions) and parameters (variables), the newly created object "o" in this case has all of the same methods and parameters of the original Object. The parameters will all be set to their default value.

Arrays

JavaScript, similar to other programming languages, has the capabilities to use a data structure called an array. An array is a collection of data where each piece of data is held in a numbered position within the array. A one-dimensional array would be similar to a column in a spreadsheet.

Each position in the array is assigned an index number beginning with 0 for the first position, 1 for the second position, and so on. This allows any position within the array, or "element" of the array, to be referenced **by its index number**.

The index of the array is indicated by the number contained within the **square brackets**. An array is implemented as an object in JavaScript and can be created using the "**new**" statement.

```
var a = new Array(); // creates an array called "a"
a[0] = 1.2; // sets the first element
a[1] = "JavaScript"; // sets the second element

var a = new Array(1.2, "JavaScript");
```

Arrays are important to understand because a number of components of the Document Object Model (DOM) are implemented as arrays, like forms, images, and elements.

The Document Object Model (DOM)

The browser provides us with a series of objects. The browser window that the page is displayed in is known as the window object. The HTML page displayed by your browser is known as the document object. The document object is probably the most commonly used object in client-side JavaScript.

The HTML elements that you add to a page also extend the object hierarchy. An example is the FORM element and the elements that reside inside the form. This means that you can reference these objects, as illustrated in the HTML page below:

window.document.forms[0]

Refers to the first form in the document. Forms are implemented as arrays in the DOM. If there is more than one form on page the numbers will start at zero and go up.

window.document.Form1

Refers to the form by name Form1

if (window.document.Form1.FirstName.value == "David") {….}else {alert('invalid user')}

Refers to the value typed into the textbox named FirstName by the client, in the form named Form1

```
<HTML>
<HEAD>
```

```
        <TITLE>Simple Form</TITLE>
</HEAD>
<BODY>

<FORM NAME="Form1">
Name: <INPUT TYPE="TEXT" NAME="FirstName"><BR>
<INPUT TYPE="Button" VALUE="Submit Info" >
</FORM>

<FORM NAME="Form2">
Name: <INPUT TYPE="TEXT" NAME="LastName"><BR>
<INPUT TYPE="Button" VALUE="Submit Info" >
</FORM>

</BODY>
</HTML>
```

Objects located in the current document, in the current window can drop the reference to those two objects. For example:

forms[0]

refers to the first form within this document

Form1 // we prefer to have a variable name for each form

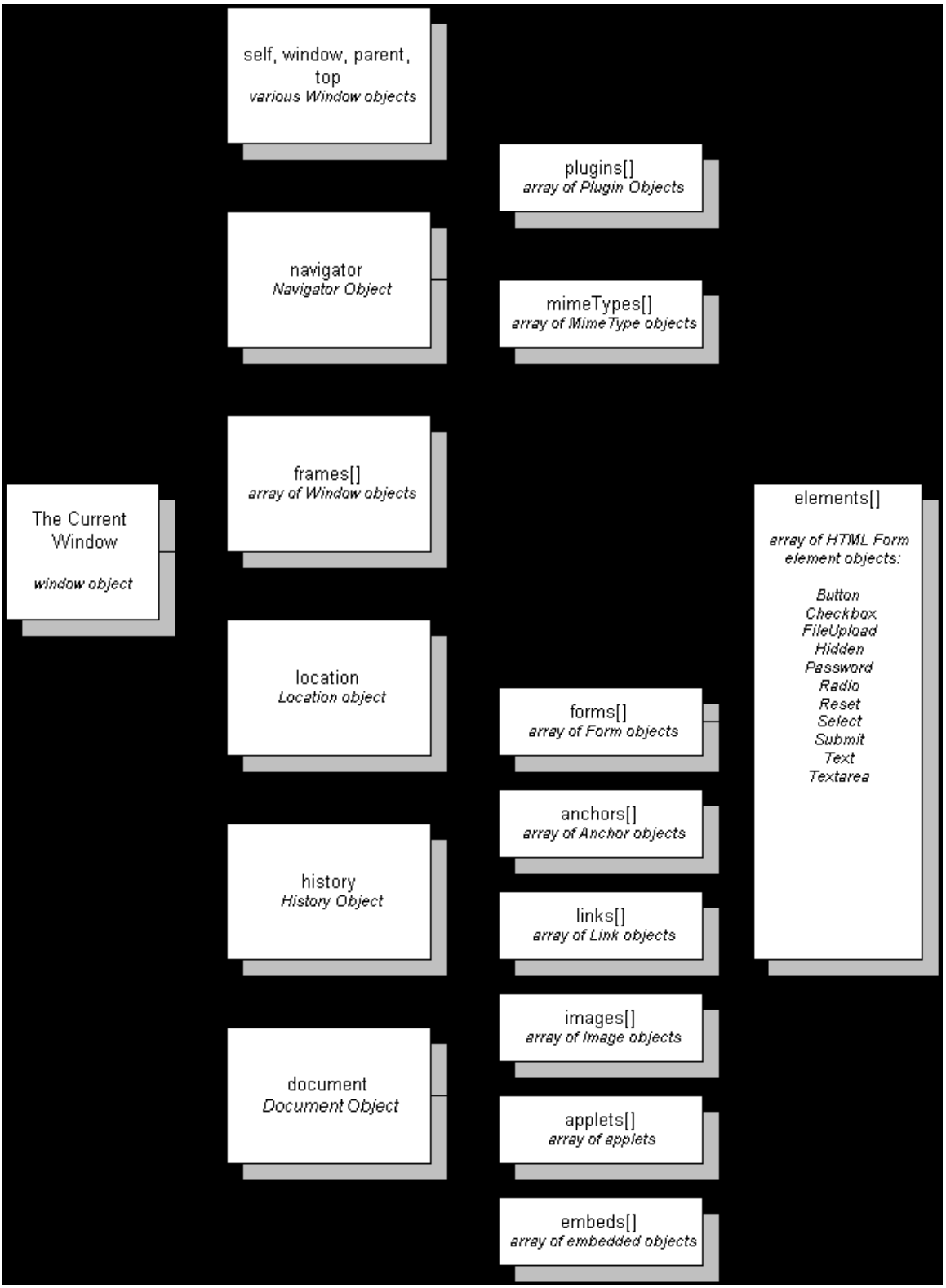
refers to the form named Form1 in this document

Form1.FullName.value

refers to the value typed, in the browser by the client, into the textbox named FullName, in the form named Form1, in this document

We recommend that you use the NAME attribute of any HTML tag that you are going to script, as in the above example the form is named Form1. This practice will simplify object naming for you.

The diagram below illustrates the Document Object Model (DOM).



Events

Events are the triggers that call (start) one of your functions. Your client-side JavaScript programs will not execute (run / be interpreted) unless started by an event. We will use the onClick event for starting our form validation scripts.

The available event handlers in JavaScript are: (remember JavaScript is case-sensitive)

onClick( )

A click event occurs when a button, checkbox, radio button, reset button, or submit button is clicked. This event is regularly used with a button to start script execution.

<INPUT TYPE="Button" VALUE="Click Me"

onClick="window.alert('You Clicked me');">

In the above example when you click on the button "Click Me" it will execute the JavaScript statement "window.alert('You Clicked me');". You can call any function or method this way.

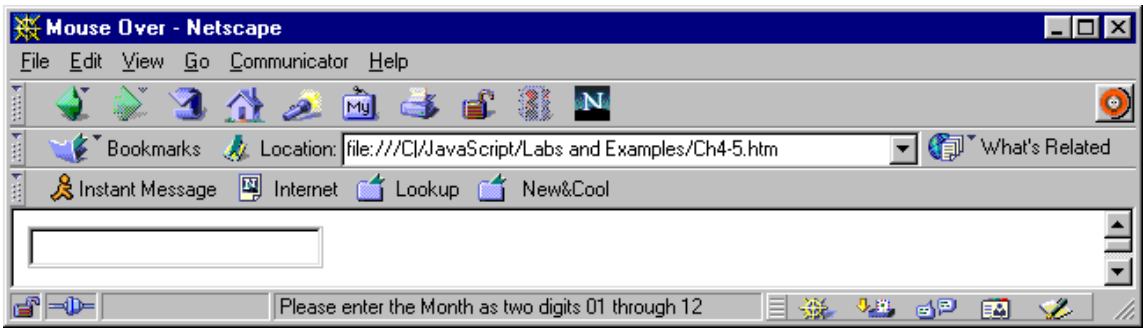
onFocus( )

This event occurs when a user tabs into or clicks on a password field, a text field, a textarea, or a FileUpload field in an HTML form. If a user clicks on one of these elements in a form, it is receiving the user's focus.

<INPUT TYPE="TEXT" NAME="Month"

onFocus="window.status=('Please enter the Month as two digits 01 through 12'); return true;">

In this example when the user clicks on the month box or tabs into it a message is displayed in the status line of the browser that indicates what the user should type in.



onChange( )

The change event happens when the user leaves a password, text, textarea or FileUpload field in an HTML form, and its value has changed.

<INPUT TYPE="TEXT" NAME="Month"

onChange="window.status=('The value of the Month Changed!!!!'); return true;" >

<INPUT TYPE="TEXT" NAME="Year">

onBlur( )

The blur event triggers when the user leaves a password, text, textarea or FileUpload field in an HTML form.

<INPUT TYPE="TEXT" NAME="Month"

onBlur="window.status=('Do you not care about the value of the Month!'); return true;">

<INPUT TYPE="TEXT" NAME="Year">

Note: It is usually a good idea to use either onChange or onBlur, and not both at the same time as they will appear to conflict with one another.

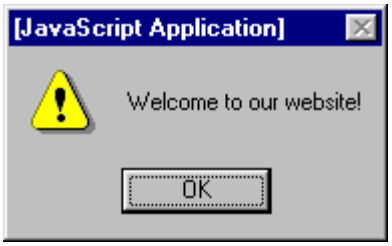
onLoad( )

The load event triggers when the browser finishes loading a document or all the frame pages within a <FRAMESET>.

<BODY onLoad="alert('Welcome to our website!');">

// most of time, we use onLoad() to initialize variable values

In this example after the page has completed loading into the browser the alert message below is popped up.



onUnload( )

The unload event occurs when you move to a new document. For example if you use the back button, or click on a link to another page the unload event will occur.

<BODY onUnload="alert('Thanks for checking out our site!');">

This example will popup the following message when someone moves off of the current page.