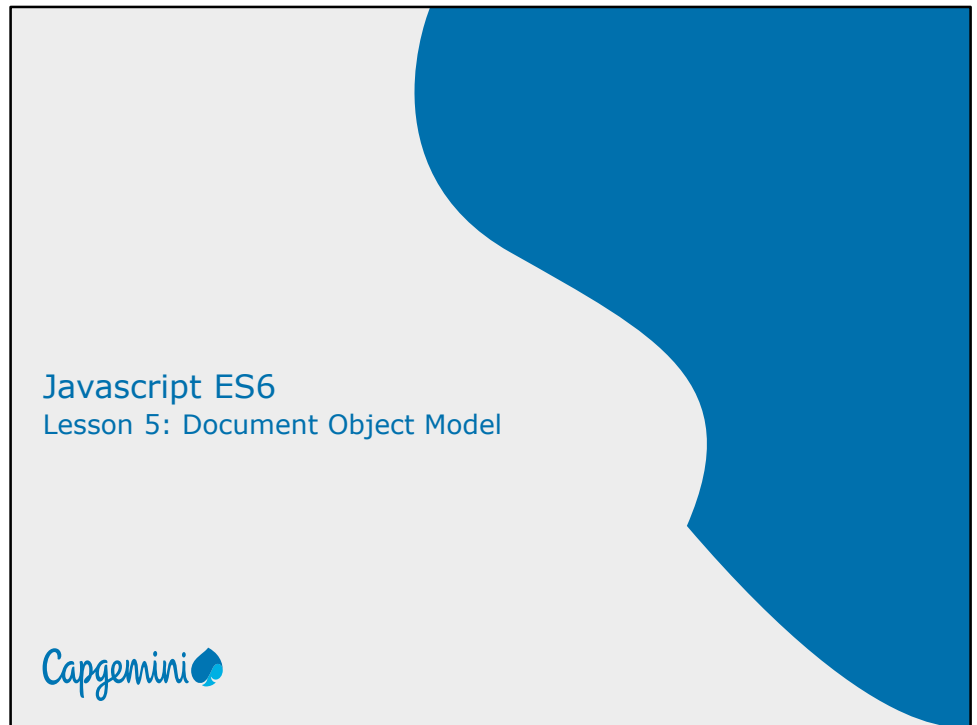


**Instructor Notes:**



**Instructor Notes:**

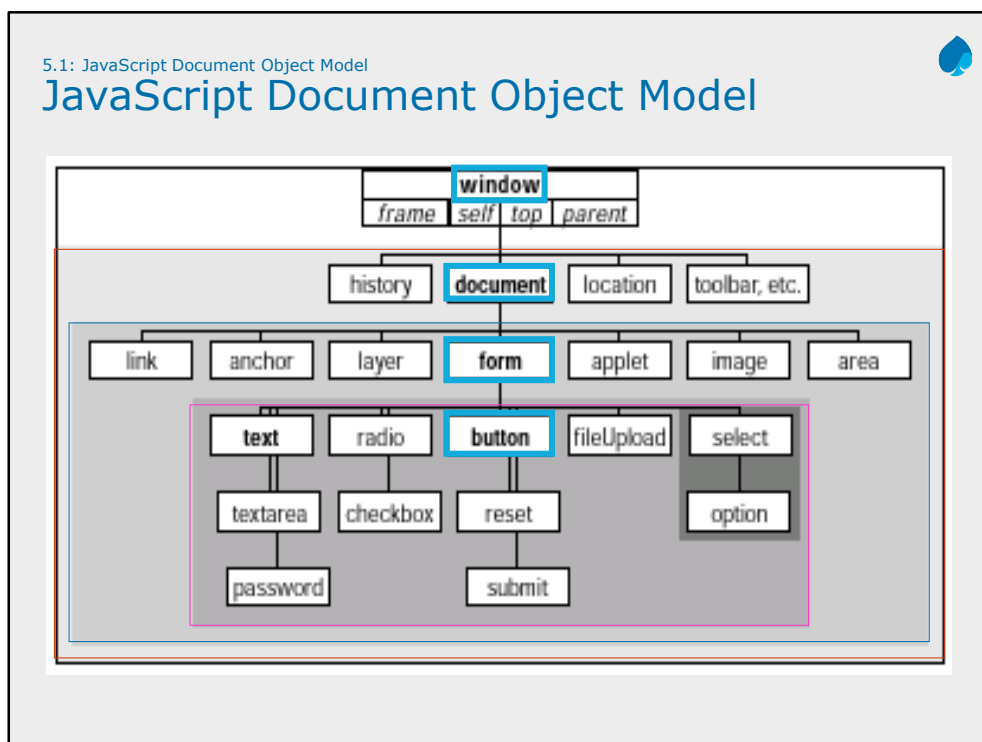
### Lesson Objectives

After completing this module you will be able to:

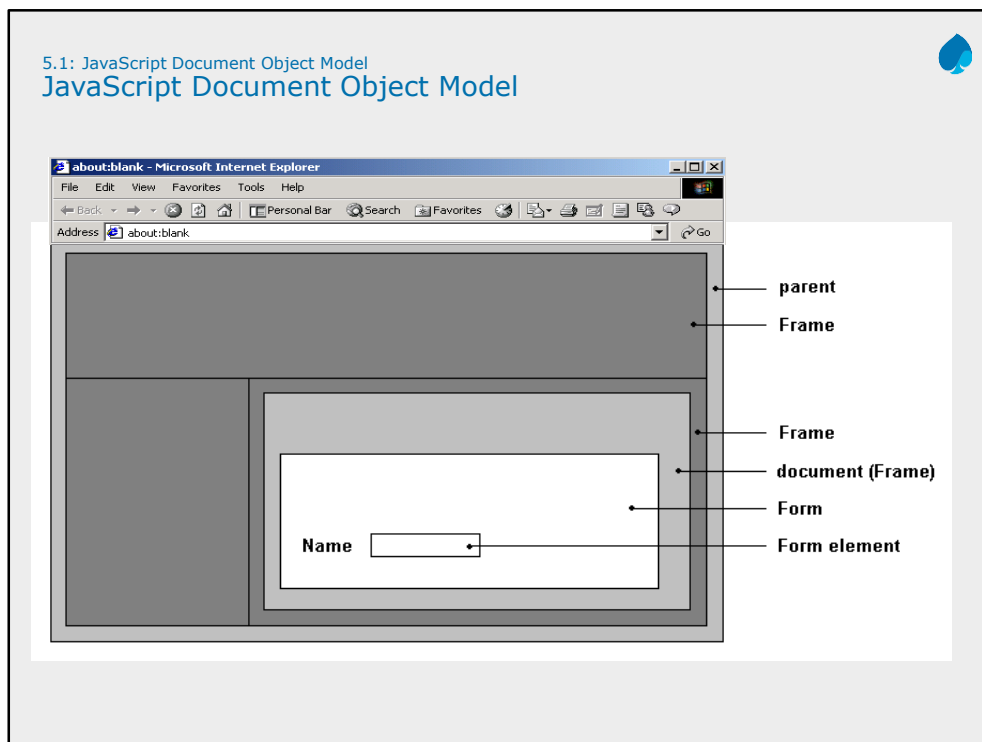
- Understand the JavaScript Object Model
- Understand the *Window* object



## Instructor Notes:



The figure shows the complete JavaScript document object hierarchy as implemented in Netscape Navigator 4. Notice that the window object is the topmost object in the entire scheme. Everything you script in JavaScript is in the browser's window, be it the window itself or a form element. Of all the objects shown in the figure, you are likely to work most with the ones that appear in **boldface**. Objects whose names appear in *italics* are synonyms for the window object, and are used only in some circumstances. Pay attention to the shading of the concentric rectangles. Every object in the same shaded area is at the same level relative to the window object. When a link from an object extends to the next darker shaded rectangle, that object contains all the objects in darker areas. There exists at most one of these links between levels. A window object contains a document object; a document object contains a form object; a form object contains many different kinds of form elements. Study this figure to establish a mental model for the scriptable elements of a Web page. After you script these objects a few times, the object hierarchy will become second nature to you — even if you do not remember every detail ( property, method, and event handler) of every object. At least you know where to look for information.

**Instructor Notes:****Creating JavaScript Objects**

Most of the objects that a browser creates for you are established when an HTML document loads into the browser. The same kind of HTML code you used to create links, anchors, and input elements tell a JavaScript-enhanced browser to create those objects in memory. The objects are there whether or not your scripts call them into action.

The only visible differences to the HTML code for defining those objects are one or more optional attributes specifically dedicated to JavaScript. By and large, these attributes specify the event you want the user interface element to react to and what JavaScript should do when the user takes that action. If you rely on the document's HTML code to perform the object generation, you spend more time figuring out how to do things with those objects or have them do things for you. Bear in mind that objects are created in their load order, which is why you should put most, if not all, deferred function definitions in the document's Head. If you create a multi-frame environment, a script in one frame cannot communicate with another frame's objects until both frames load.

**Instructor Notes:**

## 5.1: JavaScript Document Object Model

**Object Properties**

Define a particular, current setting of an object

Property names are case-sensitive

Each property determines its own read-write status

Any property you set survives as long as the document remains loaded in the window

For example:

```
document.forms[0].phone.value = "555-1212"  
  
document.forms[0].phone.delimiter = "-"
```

Object Properties

A property generally defines a particular, current setting of an object. The setting may reflect a visible attribute, such as a document's background color. It may also contain information that is not so obvious, such as the form *action* and *method* when it is submitted.

Document objects have most of their properties assigned by attribute settings of HTML tags that generate the objects. Thus, a property may be a string (for example, a name) or a number (for example, a size). A property can also be an array, such as an array of images contained by a document. If the HTML does not include all attributes, the browser usually provides default value for both attributes and corresponding JavaScript properties.

When used in script statements, property names are case-sensitive. Therefore, if you see a property name listed as *bgColor*, you must use it in a script statement with that exact case usage. But when you set an initial value of a property by way of an HTML attribute, the attribute name (like all of HTML) is not case-sensitive. Thus, **<BODY BGCOLOR="white">** and **<body bgcolor="white">** both set the same property value.

Object Methods

An object's method is a command that a script can give to that object. Some methods return values, but that is not a prerequisite for a method. Also, not every object has methods defined for it. In a majority of cases, invoking a method from a script causes some action to take place. It may be an obvious action, such as resizing a window, or something more subtle, such as processing a mouse click.

**Instructor Notes:**

## 5.1: JavaScript Document Object Model

**Event Handlers**

Specify how an object reacts to an event

- Event can be triggered by a user action or a browser action.

There are two ways to map functions to events

- Event handlers as methods:

```
document.formName.button1.onclick=f1()
```

- Event handlers as properties:

```
<INPUT TYPE="button" NAME="button1" onClick="f1()">
```

Object Event Handlers

Event handlers specify how an object reacts to an event, whether the event is triggered by a user action (for example, a button click) or a browser action (for example, the completion of a document load). Event Handlers can be specified as methods or they can be specified using attributes in tags.

**Instructor Notes:**

## 5.2: Window Object

**Working with Window Object****Window object:**

- Unique position at the top of the JavaScript object hierarchy
- Can be omitted from object references since everything takes place in a window

The following two statements are the same

- `window.alert("Welcome to Javascript ")`
- `alert("Welcome to Javascript ")`

**Properties**

- `defaultStatus`
- `status`
- `closed`

**About this Object**

The *window* object has the unique position of being at the top of the JavaScript object hierarchy. This exalted location gives it a number of properties and behaviors unlike any other object. Among the list of properties for the window object is one called *self*. This property is synonymous to the window object itself. When you start your browser, it usually opens a window. That window is a valid window object, even if it is blank. This object is also the level at which a script asks the browser to display any of the three styles of the dialog boxes (a plain alert dialog box, an OK-Cancel confirmation dialog box, or a prompt for user text entry).


Property	Description
<code>defaultStatus</code>	<code>window.defaultStatus</code> property is normally an empty string, it sets or returns the default text which is in the statusbar of the window
<code>Status</code>	This property sets a text value to be displayed in the status bar
<code>closed</code>	Returns a boolean value which indicated if the window has been closed or no

## Instructor Notes:

5.2: Window Object  
Window Object Methods


**alert(message)**

```
window.alert("Display Message")
```



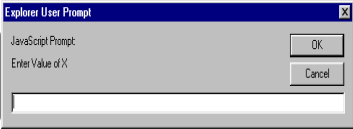
**confirm(message)**

```
window.confirm("Exit Application ?")
```



**prompt(message,[defaultReply])**

```
var input= window.prompt("Enter value of X")
```



Method	Description
alert(message)	An alert dialog box is a modal window that presents a message to the user with a single OK button to dismiss the dialog box.
confirm(message)	A confirm dialog box presents a message in a modal dialog box along with OK and Cancel buttons. Such a dialog box can be used to ask a question of the user, usually prior to a script performing actions that will not be undoable.
prompt(message, defaultReply)	The third kind of dialog box that JavaScript can display includes a message from the script author, a field for user entry, and two buttons (OK and Cancel).



## Instructor Notes:

 5.2: Window Object  
 Window Object Methods


```
open("URL", "windowName" [, "windowFeatures"])
```

```
newwin=window.open("new/URL","NewWindow",  
"toolbar,status,resizable")
```

```
close()
```

```
moveBy(deltaX,deltaY), moveTo(x,y)
```

```
scrollBy(deltaX,deltaY), scrollTo(x,y)
```

<pre>open("URL", "windowName" [, "windowFeatures"])</pre>	<p>The <code>window.open()</code> method, provides a Web site designer with options for the way a new browser window should look on the user's computerscreen. The optional <i>windowFeatures</i> parameter is <i>one string</i>, that comprises a comma-separated list of assignment expressions. Boolean values for true can be either yes, 1, or just the feature name by itself; for false, use a value of no or 0. If you omit any Boolean attributes, they are rendered as false. Therefore, if you want to create a new window that shows only the toolbar and statusbar and is resizable, the method looks like this:  <code>window.open("newURL","NewWindow", "toolbar,status,resizable")</code>.</p>
<pre>close()</pre>	<p>The <code>window.close()</code> method closes the browser window referenced by the window object.</p>
<pre>scrollBy(deltaX,deltaY) scrollTo(x,y)</pre>	<p><code>scrollBy(..)</code> method scrolls the content by the specified number of pixels which is relative scroll. <code>scrollTo(..)</code> is an absolute scroll to the specified coordinates</p>
<pre>moveBy(deltaX,delta Y) moveTo(x,y)</pre>	<p><code>moveBy(..)</code> moves the window relative to the current position. <code>moveT(..)</code> moves it to the specified coordinates</p>

## Instructor Notes:

 5.2: Window Object  
 Window Object Methods


setTimeout, clearTimeout

```
y=setTimeout('scroll()', '100')
```

```
clearTimeout(y)
```

setInterval, clearInterval

```
y=setInterval('scroll()', '100')
```

```
clearInterval(y)
```

setTimeout("functionOrExpr", msecDelay [, funcarg1, ..., funcargn])	Javascript holds a statement or function from executing for the desired amount of time. The timeout value is in milliseconds
setInterval("functionOrExpr", msecDelay, language)	Use this method when your script needs to call a function or execute some expression repeatedly with a fixed time delay between calls to that function or expression. The timeinterval is in milliseconds. Optional Language i.e Javascript, vbscript
clearInterval (intervalIDnumber)	Use this method to turn off an interval loop action started with the <i>window.setInterval()</i> method. The parameter is the ID number returned by the <i>setInterval()</i> method.
clearTimeout (timeoutIDnumber)	Use the <i>clearTimeout()</i> method in concert with the <i>window.setTimeout()</i> method when you want your script to cancel a timer that is waiting to run its expression. The parameter for this method is the ID number that the <i>setTimeout()</i> method returns when the timer starts ticking.

**Instructor Notes:**

5.2: Window Object

**Window Object Event Handlers****Event Handlers for the Window Object**

- onBlur
- onFocus
- onLoad

Event Handlers

Table 6.3 Window Object Event Handlers

Event Handler	Description
onBlur onFocus	Fired when window or frame has been activated and deactivated respectively.
onLoad	The load event is sent to the current window at the end of the document loading process.

**Instructor Notes:****Demo**

Window\_object.html  
setTimeout\_method.html  
Window\_ex.html  
setInterval\_method.html



**Instructor Notes:****Lab****Lab 5 :**

- Working with Document Object Model (DOM)



**Instructor Notes:**

### Summary



Document Object Model is a interface that allows programs and scripts to dynamically access and update content, structure and style of documents

Window object is the topmost object in the entire scheme. It has properties, methods and event handlers

The history object has an array of history items having details of the URL's visited from within that window

The Location object contains information about the current URL



Summary

**Instructor Notes:****Review Questions**

Question 1: Closed property returns \_\_\_\_\_ if the window object is closed either by a script or by the user.

- Option 1: 1
- Option 2: True
- Option 3: 0

Question 2: An alert dialog box is a modal window that presents a message for users with a single OK button to dismiss it.

- True / False



**Instructor Notes:****Review Questions (Contd..)**

Question 4: The \_\_\_\_\_ and appName properties are simply the official name and the internal code name for the browser application.

- Option 1: Appname
- Option 2: appName
- Option 3: applName



Question 5: The \_\_\_\_\_ property supplies a string of the entire URL of the specified window object.

- Option 1: location.href
- Option 2: hostname
- Option 3: hash

Question 6: The \_\_\_\_\_ property describes both the hostname and port of a URL.