

Tutorial 4

Working with Objects and Styles

Objectives

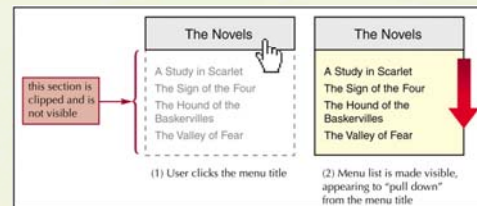
- Learn about objects and the document object model
- Reference documents objects by ID, name, and tag name
- Work with object collections
- Work with object properties and CSS styles
- Study the syntax of object methods
- Apply an event handler to an object

Objectives (cont'd)

- Work with mouse events
- Employ object detection to avoid programming errors
- Create an animation with timed functions
- Explore how to create sliding and tabbed menus
- Retrieve values from a style sheet
- Create custom objects

Introducing Pull-Down Menus

- In a **pull-down menu**, a menu title is always visible to the user, identifying the entries in the menu



Introducing Objects, Properties, Methods, and Events

- JavaScript
 - **Object-based** language
 - Based on manipulating objects through use of **properties, methods, and events**
 - Supports three types of objects:
 - **Built-in objects**
 - **Document objects**
 - **Customized objects**

Exploring the Document Object Model

- **Document object model (DOM)**
 - Organized structure of objects
 - Goal is to make every object related to the document or Web browser accessible to a scripting language

DOM	Descriptions
DOM Level 0 (Basic Model)	The basic DOM that supported how page and browser objects and allowed dynamic content only for form elements
DOM Level 0 + Images	The basic DOM with added support for image rollovers
Netscape 4 (DHTML)	The basic DOM with support for the Netscape 4 layer element and the ability to capture events within the browser
Internet Explorer 4	An expanded DOM allowing dynamic content for most page elements
Internet Explorer 5	The IE 4 DOM with additional refinements and enhancements
W3C DOM Level 1	The first DOM specification by the W3C, which supported all page and browser elements and handled all events occurring within the browser
W3C DOM Level 2	The second DOM specification, allowing for the capture of events, manipulation of CSS styles, working with element text, and document subsets
W3C DOM Level 3	The third DOM specification, providing a framework for working with document loading and saving, as well as working with DTDs and document validation

Exploring the Document Object Model

- Each document object model organizes objects into a hierarchy known as a **document tree**
- Tree structure becomes more elaborate as DOMs encompass more objects



New Perspectives on
JavaScript and AJAX, 2nd Edition

7

Referencing Objects

- Object name** identifies each object
- Use **dot syntax** (i.e., separate each level using a dot) to indicate location of an object within the hierarchy

Object Name	Description
window	The browser window
document	The Web document displayed in the window
document.body	The body of the Web document displayed in the browser window
event	Events or actions occurring within the browser window
history	The list of previously visited Web sites within the browser window
location	The URL of the document currently displayed in the browser window
navigator	The browser itself
screen	The screen displaying the document

New Perspectives on
JavaScript and AJAX, 2nd Edition

8

Referencing Objects

- Object collections**
 - Are arrays of more than one of the same type of object
 - Support the length property

Object Collection	Description
document.anchors	All anchors
document.applets	All applets
document.embeds	All embed elements
document.forms	All Web forms
document.forms.elements	All elements within a specific form
document.images	All inline images
document.links	All links
document.plugins	All plug-ins in the document
document.styleSheets	All style sheet elements
navigator.plugins	All plug-ins supported by the browser
navigator.mimeTypes	All mime-types supported by the browser
window.frames	All frames within the browser window

New Perspectives on
JavaScript and AJAX, 2nd Edition

9

Referencing Objects

To reference...	Use...
An object as part of the collection in a document	<code>collection[idref]</code> or <code>collection.idref</code>
A document object based on its ID	<code>document.getElementById(id)</code>
An array of elements based on the tag name	<code>object.getElementsByTagName(tag)</code>
An array of elements based on the value of the name attribute	<code>document.getElementsByName(name)</code>

New Perspectives on
JavaScript and AJAX, 2nd Edition

10

Working with Object Properties

- Most objects are associated with one or more object properties
`object.property`
- To set the value of an object property:
`object.property = expression`
- To apply a CSS style to a document object:
`object.style.attribute = expression`

New Perspectives on
JavaScript and AJAX, 2nd Edition

11

Working with Object Properties

- Conventions for object properties that mirror HTML attributes:
 - Begin with lowercase
 - Use camel case for multiple words
 - Are prefaced with text string `html` if a reserved JavaScript name/keyword

New Perspectives on
JavaScript and AJAX, 2nd Edition

12

Exploring Object Methods

- To apply a method to an object:
`object.method(parameters)`

Expression	Action
<code>location.reload()</code>	Reload the current page in the browser
<code>document.form[0].reset()</code>	Reset the first form in the Web page
<code>document.form[0].submit()</code>	Submit the first form in the Web page
<code>document.write("Sherlock Holmes Novels")</code>	Write "Sherlock Holmes Novels" to the Web page
<code>history.back()</code>	Go back to the previous page in the browser's history list
<code>thisDay.getFullYear()</code>	Return the four-digit year value from the <code>thisDay</code> date object
<code>Math.rand()</code>	Return a random value using the <code>Math</code> object
<code>navigator.javaEnabled()</code>	Return a Boolean value indicating whether Java is enabled in the browser
<code>window.close()</code>	Close the browser window
<code>window.print()</code>	Print the contents of the browser window
<code>window.scrollTo(x, y)</code>	Scroll the browser window to the (x, y) coordinate

New Perspectives on
JavaScript and AJAX, 2nd Edition

13

Working with Event Handlers

- All objects can be affected by events initiated by the user or browser

```
window.onload = init; // runs the init() function after the page is loaded into the browser window

function init() {
    var menus = new Array();
    var allItems = document.getElementsByTagName("*");
    for (var i = 0; i < allItems.length; i++) {
        if (allItems[i].className == "menu") menus.push(allItems[i]);
    }
}
```

New Perspectives on
JavaScript and AJAX, 2nd Edition

14

Treating an Event Handler as an Object Property

- Advantages:
 - Provides greater flexibility in designing scripts
 - Removes scripting from the HTML code, placing it within the external script file
- Disadvantages:
 - Difficulty passing parameter values to the function assigned to the event
 - Can assign only one function at a time to an object and event

New Perspectives on
JavaScript and AJAX, 2nd Edition

15

Programming a Pull-Down Menu

- `ChangeMenu` function changes the pull-down menu from one menu list to another

```
function init() {
    var menus = new Array();
    var allItems = document.getElementsByTagName("*");
    for (var i = 0; i < allItems.length; i++) {
        if (allItems[i].className == "menu") menus.push(allItems[i]);
    }
    for (var i = 0; i < menus.length; i++) {
        menus[i].onclick = changeMenu;
    }
}

// this function changes the pull-down menu displayed in the document
function changeMenu() {
    // initial code for the changeMenu() function
}
```

New Perspectives on
JavaScript and AJAX, 2nd Edition

16

Programming a Pull-Down Menu

- `ActiveMenu` variable contains object reference of the current pull-down menu

```
var activeMenu = null; // the activeMenu variable stores the object reference of the pull-down menu

function init() {
    var menus = new Array();
    var allItems = document.getElementsByTagName("*");
    for (var i = 0; i < allItems.length; i++) {
        if (allItems[i].className == "menu") menus.push(allItems[i]);
    }
    for (var i = 0; i < menus.length; i++) {
        menus[i].onclick = changeMenu;
    }
}
```

New Perspectives on
JavaScript and AJAX, 2nd Edition

17

Programming a Pull-Down Menu

- The `this` keyword references the currently active object in the Web browser

```
function changeMenu() {
    // this function changes the pull-down menu displayed in the document
    closeMenu();
    menuID = this.id + 1;
    activeMenu = document.getElementById(menuID);
    activeMenu.style.display = "block";
}

// this keyword references the object that called the changeMenu() function
```

New Perspectives on
JavaScript and AJAX, 2nd Edition

18

Adding Handlers for Mouse Events

- Enables the displayed pull-down menu to change as users move the pointer from one menu title to another

```
function init() {
    var menus = new Array();
    var allItems = document.getElementsByTagName("*");
    for (var i = 0; i < allItems.length; i++) {
        if (allItems[i].className == "menu") menus.push(allItems[i]);
    }
    for (var i = 0; i < menus.length; i++) {
        menus[i].onclick = changeMenu;
        menus[i].onmouseover = moveMenu;
    }
    document.getElementById("logo").onclick = closeAllMenus;
    document.getElementById("title1st").onclick = closeAllMenus;
    document.getElementById("main").onclick = closeAllMenus;
}

function moveMenu() {
    // this function moves the pull-down menu from one title to another
}
```

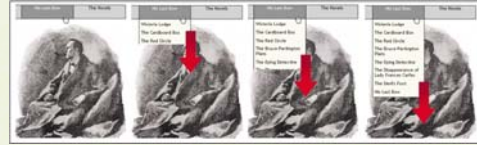
runs the moveMenu() when the pointer initially moves over a menu title

19

New Perspectives on JavaScript and AJAX, 2nd Edition

Animating a Pull-Down Menu

- Create the illusion of a menu being pulled down using the setInterval() method and the CSS clip style



20

New Perspectives on JavaScript and AJAX, 2nd Edition

Animating a Pull-Down Menu

- Adding the rollDown() function

```
function closeAllMenus() {
    if (activeMenu) {
        activeMenu.style.display = "none";
        activeMenu = null;
    }
}

function rollDown() {
    clipHeight = clipHeight + 10;
    if (clipHeight < 400) {
        activeMenu.style.clip = "rect(0px, 350px, " + clipHeight + "px, 0px)";
        clearInterval(timerID);
        clipHeight = 0;
    }
    // otherwise stops running the rollDown() function and resets the clipHeight value to 0
}
```

increases the clipping height by 10 pixels

if the height is less than 400 pixels, clips the menu

21

New Perspectives on JavaScript and AJAX, 2nd Edition

Animating a Pull-Down Menu

- Calling the rollDown() function

```
function moveMenu() {
    // this function moves the pull-down menu from one title to another
    if (activeMenu) {
        closeAllMenus();
    }
    menuID = this.id + "list";
    activeMenu = document.getElementById(menuID);
    activeMenu.style.clip = "rect(0px, 350px, 0px, 0px)";
    activeMenu.style.display = "block";
    timerID = setInterval("rollDown()", 1);
}

function changeMenu() {
    // this function changes the pull-down menu displayed in the document
    closeAllMenus();
    menuID = this.id + "list";
    activeMenu = document.getElementById(menuID);
    activeMenu.style.clip = "rect(0px, 350px, 0px, 0px)";
    activeMenu.style.display = "block";
    timerID = setInterval("rollDown()", 1);
}
```

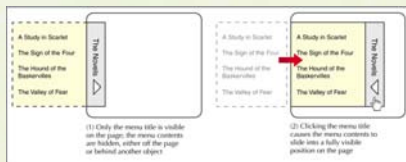
runs the rollDown() function once every millisecond until it is cleared

22

New Perspectives on JavaScript and AJAX, 2nd Edition

Creating Other Types of Menus

- Pop-up menu:** User clicks an object on the page and the menu appears, sometimes elsewhere on the page
- Sliding menu:** A menu is partially hidden either off the Web page or behind another object

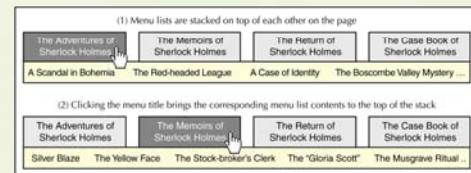


23

New Perspectives on JavaScript and AJAX, 2nd Edition

Creating Other Types of Menus

- Tabbed menu:** Several menus are stacked on the page with one part of each menu visible to the user



24

New Perspectives on JavaScript and AJAX, 2nd Edition