

Objects

Outline

- 12.1 Introduction**
- 12.2 Thinking About Objects**
- 12.3 Math Object**
- 12.4 String Object**
 - 12.4.1 Fundamentals of Characters and Strings**
 - 12.4.2 Methods of the String Object**
 - 12.4.3 Character-Processing Methods**
 - 12.4.4 Searching Methods**
 - 12.4.5 Splitting Strings and Obtaining Substrings**
 - 12.4.6 XHTML Markup Methods**
- 12.5 Date Object**
- 12.6 Boolean and Number Objects**
- 12.7 document Object**
- 12.8 window Object**
- 12.9 Using Cookies**
- 12.10 Final JavaScript Example**
- 12.11 Web Resources**

12.1 Introduction

- Use JavaScript to manipulate every element of XHTML document from a script
- Reference for several of JavaScript's built-in objects
- Demonstrates the capabilities

12.2 Thinking About Objects

- Objects
 - Attributes
 - Behaviors
 - Encapsulate data and methods
 - Property of information hiding
 - Details hidden within the objects themselves

12.3 Math Object

- Allow the programmer to perform many common mathematical calculations

12.3 Math Object

Method	Description	Example
<code>abs(x)</code>	absolute value of x	<code>abs(7.2)</code> is 7.2 <code>abs(0.0)</code> is 0.0 <code>abs(-5.6)</code> is 5.6
<code>ceil(x)</code>	rounds x to the smallest integer not less than x	<code>ceil(9.2)</code> is 10.0 <code>ceil(-9.8)</code> is -9.0
<code>cos(x)</code>	trigonometric cosine of x (x in radians)	<code>cos(0.0)</code> is 1.0
<code>exp(x)</code>	exponential method e^x	<code>exp(1.0)</code> is 2.71828 <code>exp(2.0)</code> is 7.38906
<code>floor(x)</code>	rounds x to the largest integer not greater than x	<code>floor(9.2)</code> is 9.0 <code>floor(-9.8)</code> is -10.0
<code>log(x)</code>	natural logarithm of x (base e)	<code>log(2.718282)</code> is 1.0 <code>log(7.389056)</code> is 2.0
<code>max(x, y)</code>	larger value of x and y	<code>max(2.3, 12.7)</code> is 12.7 <code>max(-2.3, -12.7)</code> is -2.3

Fig. 12.1 Math object methods.

12.3 Math Object

<code>min(x, y)</code>	smaller value of <code>x</code> and <code>y</code>	<code>min(2.3, 12.7)</code> is 2.3 <code>min(-2.3, -12.7)</code> is -12.7
<code>pow(x, y)</code>	<code>x</code> raised to power <code>y</code> (<code>xy</code>)	<code>pow(2.0, 7.0)</code> is 128.0 <code>pow(9.0, .5)</code> is 3.0
<code>round(x)</code>	rounds <code>x</code> to the closest integer	<code>round(9.75)</code> is 10 <code>round(9.25)</code> is 9
<code>sin(x)</code>	trigonometric sine of <code>x</code> (<code>x</code> in radians)	<code>sin(0.0)</code> is 0.0
<code>sqrt(x)</code>	square root of <code>x</code>	<code>sqrt(900.0)</code> is 30.0 <code>sqrt(9.0)</code> is 3.0
<code>tan(x)</code>	trigonometric tangent of <code>x</code> (<code>x</code> in radians)	<code>tan(0.0)</code> is 0.0
Fig. 12.1 Math object methods.		

12.3 Math Object

Constant	Description	Value
Math.E	Base of a natural logarithm (e).	Approximately 2.718.
Math.LN2	Natural logarithm of 2.	Approximately 0.693.
Math.LN10	Natural logarithm of 10.	Approximately 2.302.
Math.LOG2E	Base 2 logarithm of e .	Approximately 1.442.
Math.LOG10E	Base 10 logarithm of e .	Approximately 0.434.
Math.PI	π —the ratio of a circle's circumference to its diameter.	Approximately 3.141592653589793.
Math.SQRT1_2	Square root of 0.5.	Approximately 0.707.
Math.SQRT2	Square root of 2.0.	Approximately 1.414.

Fig. 12.2 Properties of the Math object.

12.4 String Object

- JavaScript's string and character-processing capabilities
- Appropriate for processing names, addresses, credit card information, etc.

12.4.1 Fundamentals of Characters and Strings

- Characters
 - Fundamental building blocks of JavaScript programs
- String
 - Series of characters treated as a single unit

12.4.2 Methods of the String Object

Method	Description
<code>charAt(index)</code>	Returns a string containing the character at the specified <i>index</i> . If there is no character at the <i>index</i> , <code>charAt</code> returns an empty string. The first character is located at <i>index</i> 0.
<code>charCodeAt(index)</code>	Returns the Unicode value of the character at the specified <i>index</i> . If there is no character at the <i>index</i> , <code>charCodeAt</code> returns NaN (Not a Number).
<code>concat(string)</code>	Concatenates its argument to the end of the string that invokes the method. The string invoking this method is not modified; instead a new String is returned. This method is the same as adding two strings with the string concatenation operator + (e.g., <code>s1.concat(s2)</code> is the same as <code>s1 + s2</code>).
<code>fromCharCode(value1, value2,)</code>	Converts a list of Unicode values into a string containing the corresponding characters.
<code>indexOf(substring, index)</code>	Searches for the first occurrence of <i>substring</i> starting from position <i>index</i> in the string that invokes the method. The method returns the starting index of <i>substring</i> in the source string or -1 if <i>substring</i> is not found. If the <i>index</i> argument is not provided, the method begins searching from index 0 in the source string.
<code>LastIndexOf(substring, index)</code>	Searches for the last occurrence of <i>substring</i> starting from position <i>index</i> and searching toward the beginning of the string that invokes the method. The method returns the starting index of <i>substring</i> in the source string or -1 if <i>substring</i> is not found. If the <i>index</i> argument is not provided, the method begins searching from the end of the source string.

Fig. 12.3 String object methods.

12.4.2 Methods of the String Object

<code>slice(<i>start</i>, <i>end</i>)</code>	Returns a string containing the portion of the string from index <i>start</i> through index <i>end</i> . If the <i>end</i> index is not specified, the method returns a string from the <i>start</i> index to the end of the source string. A negative <i>end</i> index specifies an offset from the end of the string starting from a position one past the end of the last character (so <code>-1</code> indicates the last character position in the string).
<code>split(<i>string</i>)</code>	Splits the source string into an array of strings (tokens) where its <i>string</i> argument specifies the delimiter (i.e., the characters that indicate the end of each token in the source string).
<code>substr(<i>start</i>, <i>length</i>)</code>	Returns a string containing <i>length</i> characters starting from index <i>start</i> in the source string. If <i>length</i> is not specified, a string containing characters from <i>start</i> to the end of the source string is returned.
<code>substring(<i>start</i>, <i>end</i>)</code>	Returns a string containing the characters from index <i>start</i> up to but not including index <i>end</i> in the source string.
<code>toLowerCase()</code>	Returns a string in which all uppercase letters are converted to lowercase letters. Non-letter characters are not changed.
<code>toUpperCase()</code>	Returns a string in which all lowercase letters are converted to uppercase letters. Non-letter characters are not changed.
<code>toString()</code>	Returns the same string as the source string.
<code>valueOf()</code>	Returns the same string as the source string.

Fig. 12.3 String object methods.

12.4.2 Methods of the String Object

<i>Methods that generate XHTML tags</i>	
<code>anchor(name)</code>	Wraps the source string in an anchor element (<code><a></code>) with <i>name</i> as the anchor name.
<code>blink()</code>	Wraps the source string in a <code><blink></blink></code> element.
<code>fixed()</code>	Wraps the source string in a <code><tt></tt></code> element.
<code>link(url)</code>	Wraps the source string in an anchor element (<code><a></code>) with <i>url</i> as the hyperlink location.
<code>strike()</code>	Wraps the source string in a <code><strike></strike></code> element.
<code>sub()</code>	Wraps the source string in a <code><sub></sub></code> element.
<code>sup()</code>	Wraps the source string in a <code><sup></sup></code> element.

Fig. 12.3 String object methods.

12.4.3 Character Processing Methods

- `charAt`
 - Returns the character at specific position
- `charCodeAt`
 - Returns Unicode value of the character at specific position
- `fromCharCode`
 - Returns string created from series of Unicode values
- `toLowerCase`
 - Returns lowercase version of string
- `toUpperCase`
 - Returns uppercase version of string

```

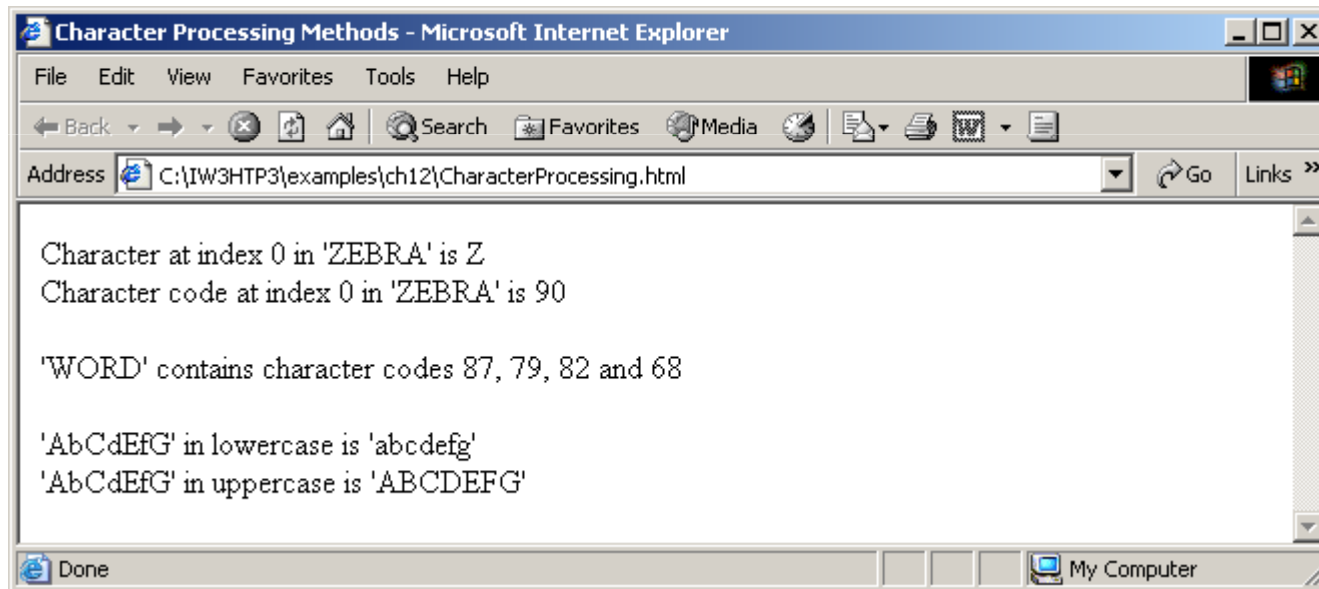
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 12.4: CharacterProcessing.html -->
6 <!-- Character Processing Methods      -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head>
10     <title>Character Processing Methods</title>
11
12     <script type = "text/javascript">
13       <!--
14       var s = "ZEBRA";
15       var s2 = "AbCdEfg";
16
17       document.writeln( "<p>Character at index 0 in '" +
18         s + "' is " + s.charAt( 0 ) );
19       document.writeln( "<br />Character code at index 0 in '"
20         + s + "' is " + s.charCodeAt( 0 ) + "</p>" );
21
22       document.writeln( "<p>'" +
23         String.fromCharCode( 87, 79, 82, 68 ) +
24         "' contains character codes 87, 79, 82 and 68</p>" )
25

```

CharacterProcessing.
html
1 of 2

```
26     document.writeln( "<p>" + s2 + "' in lowercase is '" +
27         s2.toLowerCase() + "'" );
28     document.writeln( "<br />" + s2 + "' in uppercase is '"
29         + s2.toUpperCase() + "'</p>" );
30     // -->
31 </script>
32
33 </head><body></body>
34 </html>
```

CharacterProcessing.
html
2 of 2



12.4.4 Searching Methods

- `indexOf` and `lastIndexOf`
 - Search for a specified substring in a string


```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 12.5: SearchingStrings.html -->
6 <!-- Searching Strings           -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head>
10     <title>
11       Searching Strings with indexOf and lastIndexOf
12     </title>
13
14     <script type = "text/javascript">
15       <!--
16       var letters = "abcdefghijklmnopqrstuvwxyzabcdefghijklmnop";
17
18       function buttonPressed()
19       {
20         searchForm.first.value =
21           letters.indexOf( searchForm.inputVal.value );
22         searchForm.last.value =
23           letters.lastIndexOf( searchForm.inputVal.value );
24         searchForm.first12.value =
25           letters.indexOf( searchForm.inputVal.value, 12 );

```

SearchingStrings
.html
1 of 3

```

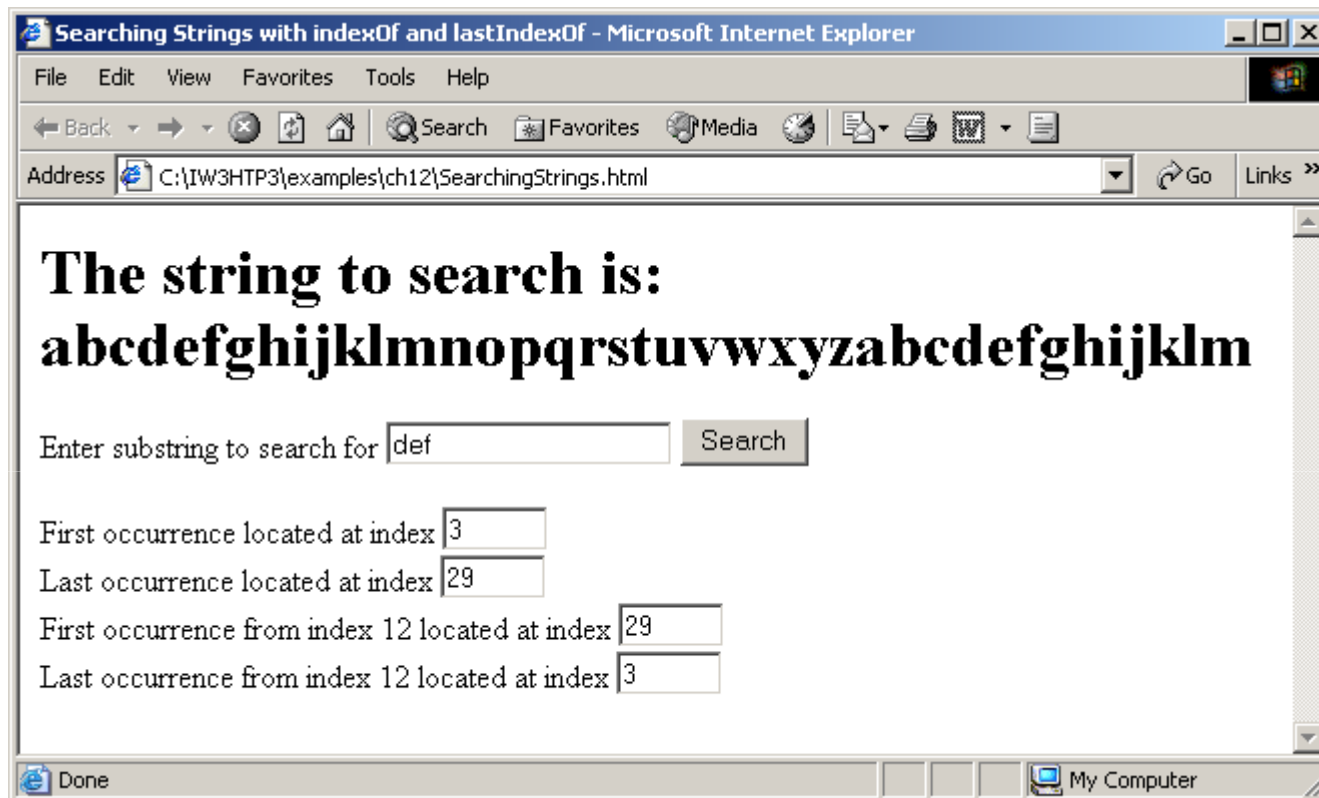
26         searchForm.last12.value =
27             letters.lastIndexOf(
28                 searchForm.inputVal.value, 12 );
29     }
30     // -->
31 </script>
32
33 </head>
34 <body>
35     <form name = "searchForm" action = "">
36         <h1>The string to search is:<br />
37             abcdefghijklmnopqrstuvwxyzabcdefghijklmnop</h1>
38         <p>Enter substring to search for
39         <input name = "inputVal" type = "text" />
40         <input name = "search" type = "button" value = "Search"
41             onclick = "buttonPressed()" /><br /></p>
42
43         <p>First occurrence located at index
44         <input name = "first" type = "text" size = "5" />
45         <br />Last occurrence located at index
46         <input name = "last" type = "text" size = "5" />
47         <br />First occurrence from index 12 located at index
48         <input name = "first12" type = "text" size = "5" />
49         <br />Last occurrence from index 12 located at index
50         <input name = "last12" type = "text" size = "5" /></p>

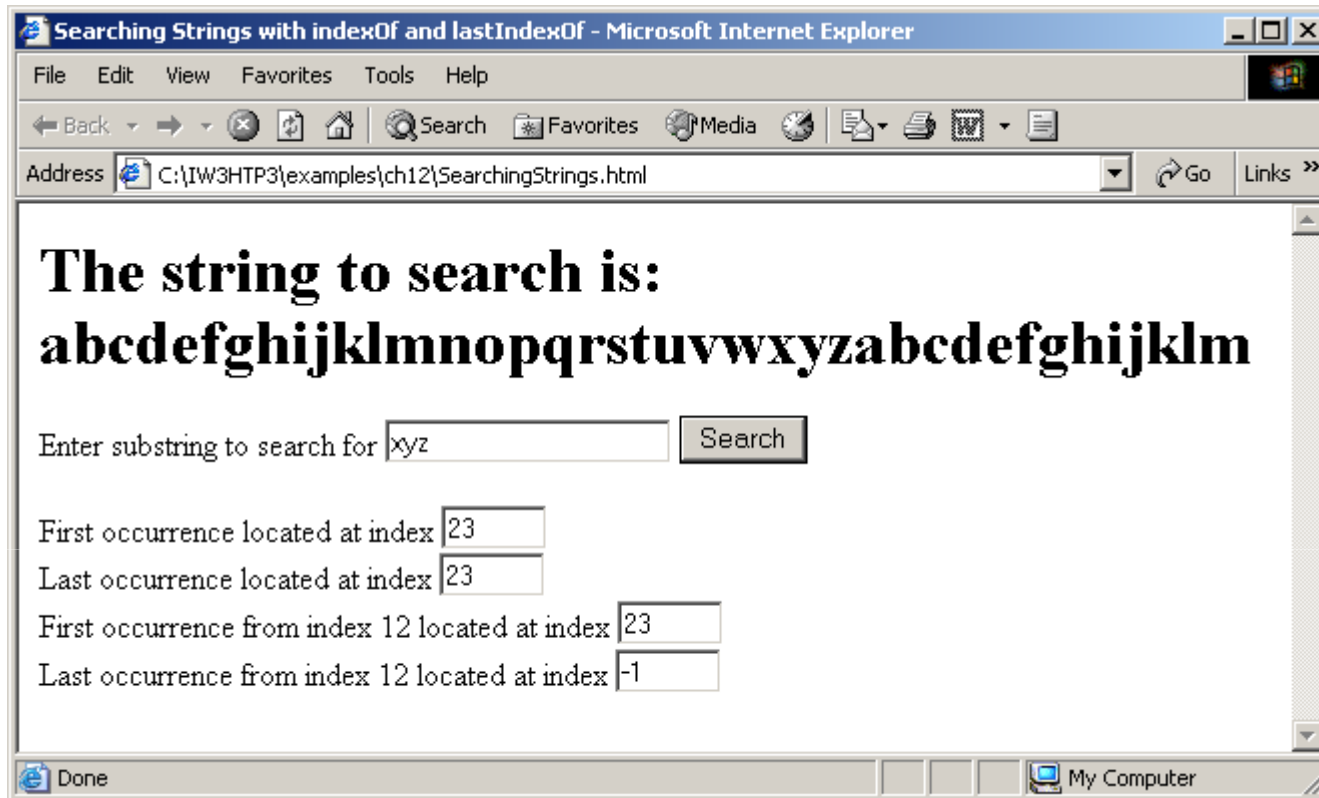
```

SearchingStrings
.html
2 of 3

```
51     </form>
52 </body>
53 </html>
```

SearchingStrings
.html
3 of 3





12.4.5 Splitting Strings and Obtaining Substrings

- Tokenization
 - The process of breaking a string into tokens
- Tokens
 - Individual words
 - Separated by delimiters

```

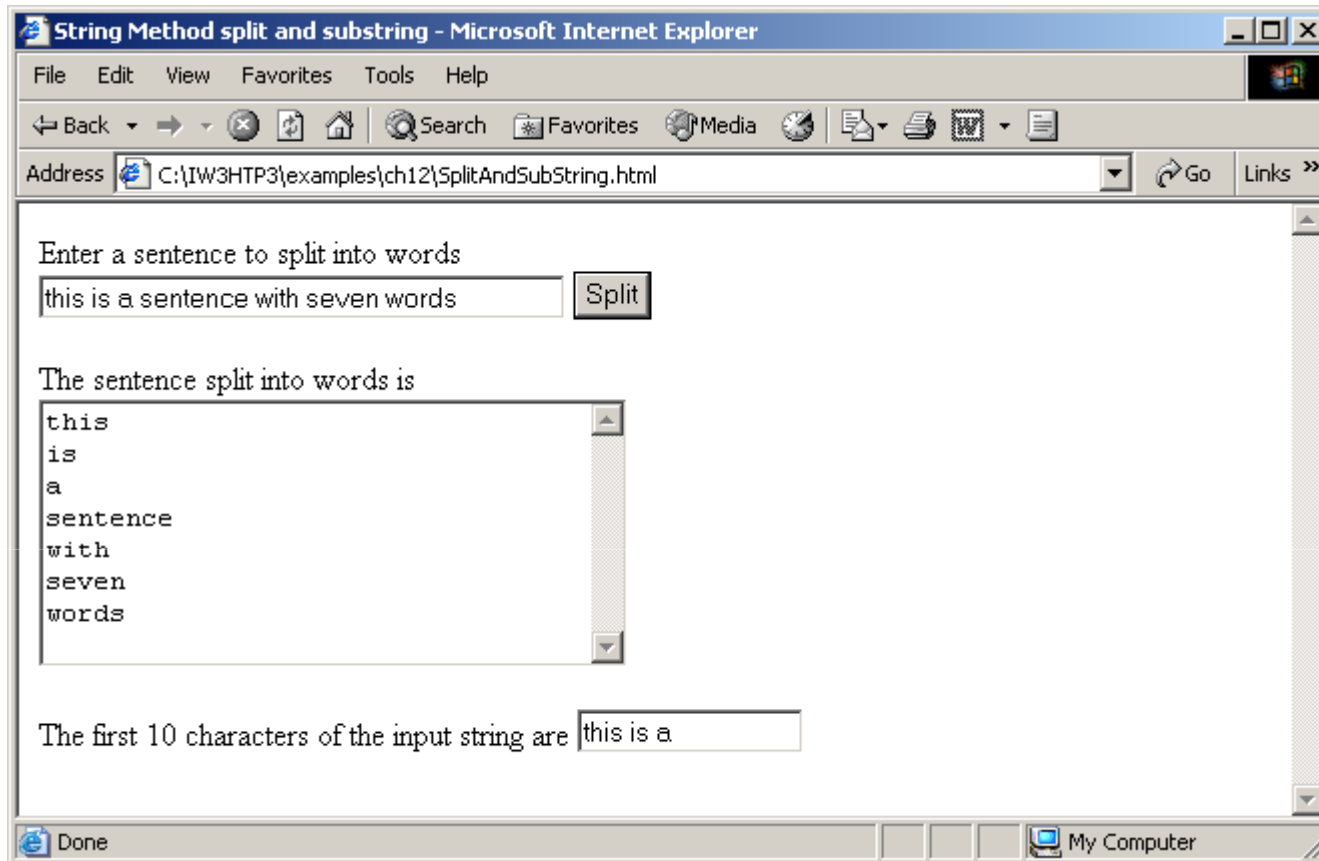
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 12.6: SplitAndSubString.html -->
6 <!-- String Method split and substring -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head>
10     <title>String Method split and substring</title>
11
12     <script type = "text/javascript">
13       <!--
14       function splitButtonPressed()
15       {
16         var strings = myForm.inputVal.value.split( " " );
17         myForm.output.value = strings.join( "\n" );
18
19         myForm.outputSubstring.value =
20         myForm.inputVal.value.substring( 0, 10 );
21       }
22       // -->
23     </script>
24   </head>
25

```

SplitAndSubString
.html
1 of 2

```
26 <body>
27   <form name = "myForm" action = "">
28     <p>Enter a sentence to split into words<br />
29     <input name = "inputVal" type = "text" size = "40" />
30     <input name = "splitButton" type = "button" value =
31       "Split" onclick = "splitButtonPressed()" /></p>
32
33     <p>The sentence split into words is<br />
34     <textarea name = "output" rows = "8" cols = "34">
35     </textarea></p>
36
37     <p>The first 10 characters of the input string are
38     <input name = "outputSubstring" type = "text"
39       size = "15" /></p>
40   </form>
41 </body>
42 </html>
```

SplitAndSubString
.html
2 of 2



12.4.6 XHTML Markup Methods

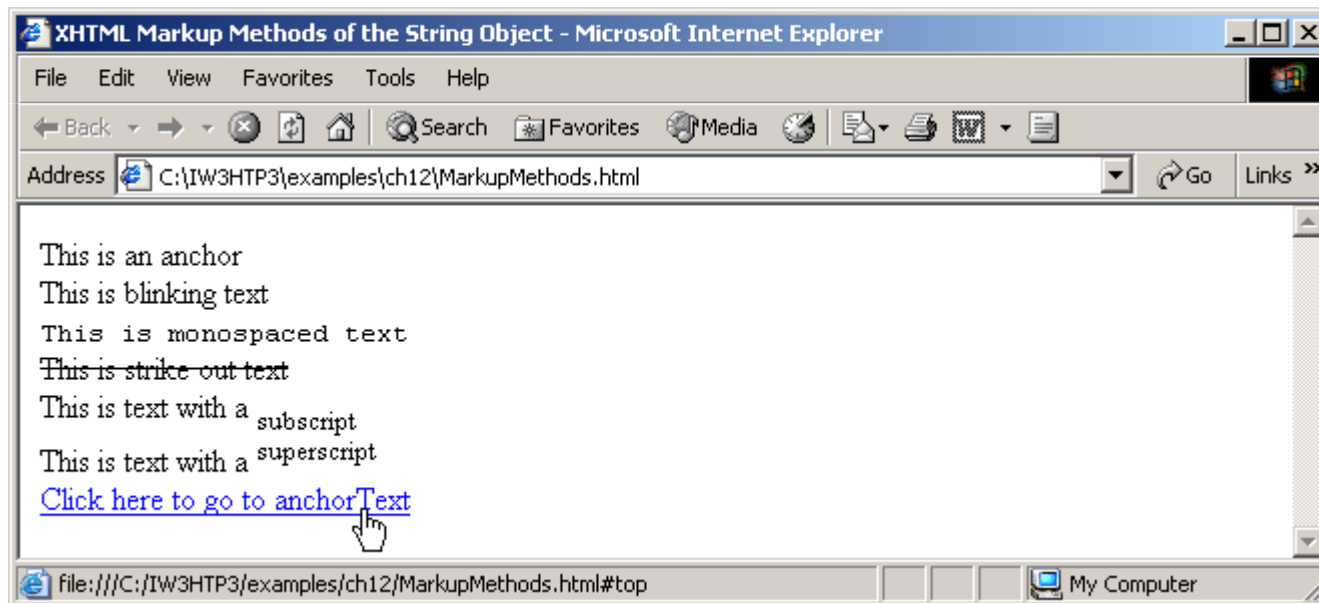
- Anchor
 - ` Anchor `
- Blink
 - `<blink> blinking text </blink>`
- Fixed
 - `<tt> monospaced text </tt>`
- Strike
 - `<strike> strike out text </strike>`
- Subscript
 - `_{subscript}`
- Superscript
 - `^{superscript}`

```
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 12.7: MarkupMethods.html -->
6 <!-- XHTML markup methods of the String object -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head>
10     <title>XHTML Markup Methods of the String Object</title>
11
12     <script type = "text/javascript">
13       <!--
14       var anchorText = "This is an anchor",
15           blinkText = "This is blinking text",
16           fixedText = "This is monospaced text",
17           linkText = "Click here to go to anchorText",
18           strikeText = "This is strike out text",
19           subText = "subscript",
20           supText = "superscript";
21
22       document.writeln( anchorText.anchor( "top" ) );
23       document.writeln( "<br />" + blinkText.blink() );
24       document.writeln( "<br />" + fixedText.fixed() );
25       document.writeln( "<br />" + strikeText.strike() );
```

MarkupMethods.html
1 of 2

```
26     document.writeln(  
27         "<br />This is text with a " + subText.sub() );  
28     document.writeln(  
29         "<br />This is text with a " + supText.sup() );  
30     document.writeln(  
31         "<br />" + linkText.link( "#top" ) );  
32     // -->  
33 </script>  
34  
35 </head><body></body>  
36 </html>
```

MarkupMethods.html
2 of 2



12.5 Date Object

- Provides methods for date and time manipulations

12.5 Date Object

Method	Description
<code>getDate()</code> <code>getUTCDate()</code>	Returns a number from 1 to 31 representing the day of the month in local time or UTC, respectively.
<code>getDay()</code> <code>getUTCDay()</code>	Returns a number from 0 (Sunday) to 6 (Saturday) representing the day of the week in local time or UTC, respectively.
<code>getFullYear()</code> <code>getUTCFullYear()</code>	Returns the year as a four-digit number in local time or UTC, respectively.
<code>getHours()</code> <code>getUTCHours()</code>	Returns a number from 0 to 23 representing hours since midnight in local time or UTC, respectively.
<code>getMilliseconds()</code> <code>getUTCMilliseconds()</code>	Returns a number from 0 to 999 representing the number of milliseconds in local time or UTC, respectively. The time is stored in hours, minutes, seconds and milliseconds.
<code>getMinutes()</code> <code>getUTCMinutes()</code>	Returns a number from 0 to 59 representing the minutes for the time in local time or UTC, respectively.
<code>getMonth()</code> <code>getUTCMonth()</code>	Returns a number from 0 (January) to 11 (December) representing the month in local time or UTC, respectively.
<code>getSeconds()</code> <code>getUTCSeconds()</code>	Returns a number from 0 to 59 representing the seconds for the time in local time or UTC, respectively.
<code>getTime()</code>	Returns the number of milliseconds between January 1, 1970 and the time in the <code>Date</code> object.
<code>getTimezoneOffset()</code>	Returns the difference in minutes between the current time on the local computer and UTC—previously known as Greenwich Mean Time (GMT).
<code>setDate(val)</code> <code>setUTCDate(val)</code>	Sets the day of the month (1 to 31) in local time or UTC, respectively.

Fig. 12.8 Methods of the Date object.

12.5 Date Object

Method	Description
<code>setFullYear(y, m, d)</code> <code>setUTCFullYear(y, m, d)</code>	Sets the year in local time or UTC, respectively. The second and third arguments representing the month and the date are optional. If an optional argument is not specified, the current value in the <code>Date</code> object is used.
<code>setHours(h, m, s, ms)</code> <code>setUTCHours(h, m, s, ms)</code>	Sets the hour in local time or UTC, respectively. The second, third and fourth arguments representing the minutes, seconds and milliseconds are optional. If an optional argument is not specified, the current value in the <code>Date</code> object is used.
<code>setMilliseconds(ms)</code> <code>setUTCMilliseconds(ms)</code>	Sets the number of milliseconds in local time or UTC, respectively.
<code>setMinutes(m, s, ms)</code> <code>setUTCMinutes(m, s, ms)</code>	Sets the minute in local time or UTC, respectively. The second and third arguments representing the seconds and milliseconds are optional. If an optional argument is not specified, the current value in the <code>Date</code> object is used.
<code>setMonth(m, d)</code> <code>setUTCMonth(m, d)</code>	Sets the month in local time or UTC, respectively. The second argument representing the date is optional. If the optional argument is not specified, the current date value in the <code>Date</code> object is used.
<code>setSeconds(s, ms)</code> <code>setUTCSeconds(s, ms)</code>	Sets the second in local time or UTC, respectively. The second argument representing the milliseconds is optional. If this argument is not specified, the current millisecond value in the <code>Date</code> object is used.

Fig. 12.8 Methods of the Date object.

12.5 Date Object

Method	Description
<code>setTime(ms)</code>	Sets the time based on its argument—the number of elapsed milliseconds since January 1, 1970.
<code>toLocaleString()</code>	Returns a string representation of the date and time in a form specific to the computer's locale. For example, September 13, 2001 at 3:42:22 PM is represented as <i>09/13/01 15:47:22</i> in the United States and <i>13/09/01 15:47:22</i> in Europe.
<code>toUTCString()</code>	Returns a string representation of the date and time in the form: <i>19 Sep 2001 15:47:22 UTC</i>
<code>toString()</code>	Returns a string representation of the date and time in a form specific to the locale of the computer (<i>Mon Sep 19 15:47:22 EDT 2001</i> in the United States).
<code>valueOf()</code>	The time in number of milliseconds since midnight, January 1, 1970.

Fig. 12.8 Methods of the Date object.

```
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5 <!-- Fig. 12.9: DateTime.html -->
6 <!-- Date and Time Methods -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head>
10     <title>Date and Time Methods</title>
11
12     <script type = "text/javascript">
13       <!--
14       var current = new Date();
15
16       document.writeln(
17         "<h1>String representations and valueOf</h1>" );
18       document.writeln( "toString: " + current.toString() +
19         "<br />toLocaleString: " + current.toLocaleString() +
20         "<br />toISOString: " + current.toISOString() +
21         "<br />valueOf: " + current.valueOf() );
22
23       document.writeln(
24         "<h1>Get methods for local time zone</h1>" );
```

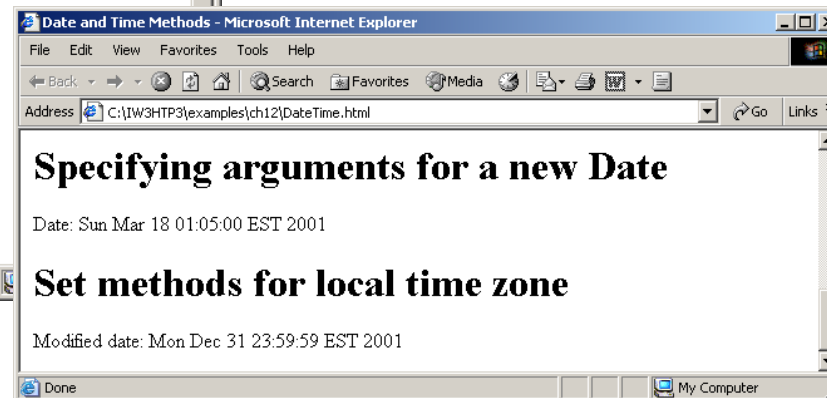
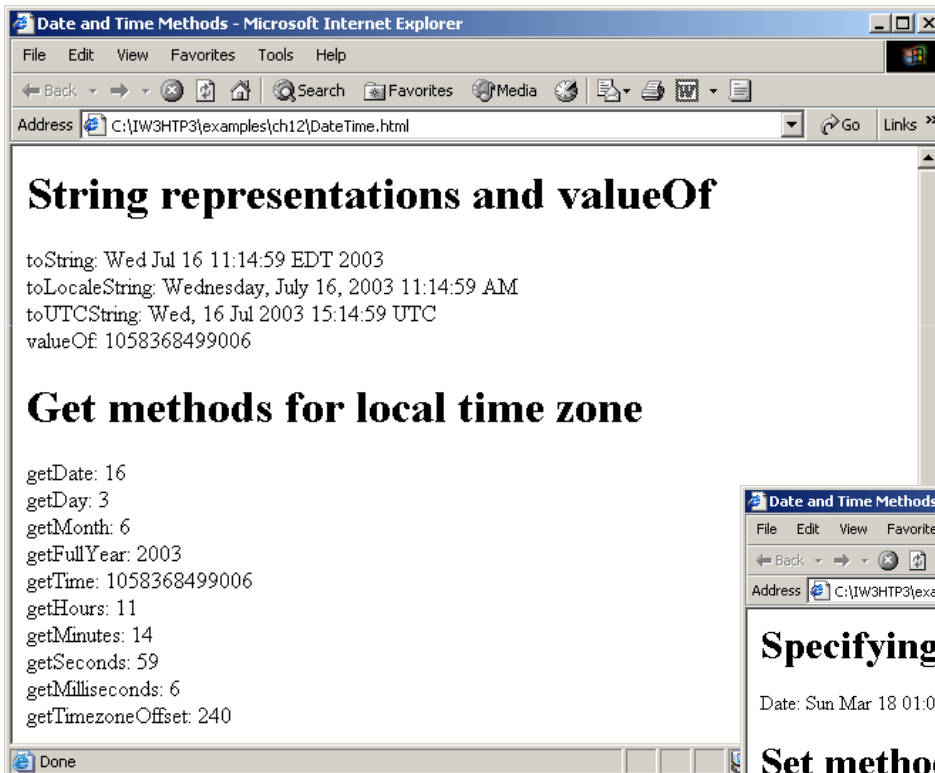
DateTime.html
1 of 3


```
25     document.writeln( "getDate: " + current.getDate() +
26         "<br />getDay: " + current.getDay() +
27         "<br />getMonth: " + current.getMonth() +
28         "<br />getFullYear: " + current.getFullYear() +
29         "<br />getTime: " + current.getTime() +
30         "<br />getHours: " + current.getHours() +
31         "<br />getMinutes: " + current.getMinutes() +
32         "<br />getSeconds: " + current.getSeconds() +
33         "<br />getMilliseconds: " +
34         current.getMilliseconds() +
35         "<br />getTimezoneOffset: " +
36         current.getTimezoneOffset() );
37
38     document.writeln(
39         "<h1>Specifying arguments for a new Date</h1>" );
40     var anotherDate = new Date( 2001, 2, 18, 1, 5, 0, 0 );
41     document.writeln( "Date: " + anotherDate );
42
43     document.writeln(
44         "<h1>Set methods for local time zone</h1>" );
45     anotherDate.setDate( 31 );
46     anotherDate.setMonth( 11 );
47     anotherDate.setFullYear( 2001 );
48     anotherDate.setHours( 23 );
49     anotherDate.setMinutes( 59 );
```

DateTime.html
2 of 3

```
50     anotherDate.setSeconds( 59 );
51     document.writeln( "Modified date: " + anotherDate );
52     // -->
53 </script>
54
55 </head><body></body>
56 </html>
```

DateTime.html
3 of 3



12.6 Boolean and Number Objects

- Object wrappers for boolean `true/false` values and numbers

12.6 Boolean and Number Objects

Method	Description
<code>toString()</code>	Returns the string “true” if the value of the <code>Boolean</code> object is true; otherwise, returns the string “false.”
<code>valueOf()</code>	Returns the value <code>true</code> if the <code>Boolean</code> object is <code>true</code> ; otherwise, returns <code>false</code> .

Fig. 12.10 `Boolean` object methods.

12.6 Boolean and Number Objects

Method or Property	Description
<code>toString(radix)</code>	Returns the string representation of the number. The optional <i>radix</i> argument (a number from 2 to 36) specifies the number's base. For example, radix 2 results in the binary representation of the number, 8 results in the octal representation, 10 results in the decimal representation and 16 results in the hexadecimal representation. See Appendix E, Number Systems for a review of the binary, octal, decimal and hexadecimal number systems.
<code>valueOf()</code>	Returns the numeric value.
<code>Number.MAX_VALUE</code>	This property represents the largest value that can be stored in a JavaScript program—approximately 1.79E+308
<code>Number.MIN_VALUE</code>	This property represents the smallest value that can be stored in a JavaScript program—approximately 2.22E-308
<code>Number.NaN</code>	This property represents <i>not a number</i> —a value returned from an arithmetic expression that does not result in a number (e.g., the expression <code>parseInt("hello")</code> cannot convert the string "hello" into a number, so <code>parseInt</code> would return <code>Number.NaN</code> . To determine whether a value is NaN, test the result with function <code>isNaN</code> , which returns <code>true</code> if the value is NaN; otherwise, it returns <code>false</code> .
<code>Number.NEGATIVE_INFINITY</code>	This property represents a value less than <code>-Number.MAX_VALUE</code> .
<code>Number.POSITIVE_INFINITY</code>	This property represents a value greater than <code>Number.MAX_VALUE</code> .

Fig. 12.11 `Number` object methods and properties.

12.7 document Object

- Manipulate document that is currently visible in the browser window

12.7 document Object

Method or Property	Description
<code>write(string)</code>	Writes the string to the XHTML document as XHTML code.
<code>writeln(string)</code>	Writes the string to the XHTML document as XHTML code and adds a newline character at the end.
<code>document.cookie</code>	This property is a string containing the values of all the cookies stored on the user's computer for the current document. See Section 12.9, Using Cookies.
<code>document.lastModified</code>	This property is the date and time that this document was last modified.
Fig. 12.12	Important document object methods and properties.

12.8 window Object

- Provides methods for manipulating browser window


```
1 <?xml version = "1.0" encoding = "utf-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3   "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
4
5 <!-- Fig. 12.13: window.html -->
6 <!-- Using the Window Object -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Using the window Object</title>
11
12 <script type = "text/javascript">
13   <!--
14   var childwindow; // variable to control the child window
15
16   function createChildwindow()
17   {
18     // these variables all contain either "yes" or "no"
19     // to enable or disable a feature in the child window
20     var toolBar // specify if toolbar will appear in child window
21     var menuBar; // specify if menubar will appear in child window
22     var location; // specify if address bar will appear in child window
23     var scrollBars; // specify if scrollbars will appear in child window
24     var status; // specify if status bar will appear in child window
25     var resizable; // specify if the child window will be resizable
```

window.html
1 of 7

```
26
27 // determine whether the Tool Bar checkbox is checked
28 if ( toolBarCheckBox.checked )
29     toolBar = "yes";
30 else
31     toolBar = "no";
32
33 // determine whether the Menu Bar checkbox is checked
34 if ( menuBarCheckBox.checked )
35     menuBar = "yes";
36 else
37     menuBar = "no";
38
39 // determine whether the Address Bar checkbox is checked
40 if ( locationCheckBox.checked )
41     location = "yes";
42 else
43     location = "no";
44
45 // determine whether the Scroll Bar checkbox is checked
46 if ( scrollBarsCheckBox.checked )
47     scrollBars = "yes";
48 else
49     scrollBars = "no";
50
```

window.html
2 of 7

```
51 // determine whether the Status Bar checkbox is checked
52 if ( statusCheckBox.checked )
53     status = "yes";
54 else
55     status = "no";
56
57 // determine whether the Resizable checkbox is checked
58 if ( resizableCheckBox.checked )
59     resizable = "yes";
60 else
61     resizable = "no";
62
63 // display window with selected features
64 childwindow = window.open( "", "", "resizable = " + resizable +
65     ", toolbar = " + toolBar + ", menubar = " + menuBar +
66     ", status = " + status + ", location = " + location +
67     ", scrollbars = " + scrollBars );
68
69 // disable buttons
70 closeButton.disabled = false;
71 modifyButton.disabled = false;
72 getURLButton.disabled = false;
73 setURLButton.disabled = false;
74 } // end function createChildwindow
75
```

window.html
3 of 7

```
76 // insert text from the textbox into the child window
77 function modifyChildwindow()
78 {
79     if ( childwindow.closed )
80         alert( "You attempted to interact with a closed window" );
81     else
82         childwindow.document.write( textForChild.value );
83 } // end function modifyChildwindow
84
85 // close the child window
86 function closeChildwindow()
87 {
88     if ( childwindow.closed )
89         alert( "You attempted to interact with a closed window" );
90     else
91         childwindow.close();
92
93     closeButton.disabled = true;
94     modifyButton.disabled = true;
95     getUrlButton.disabled = true;
96     setURLButton.disabled = true;
97 } // end function closeChildwindow
98
```

window.html
4 of 7

```
99 // copy the URL of the child window into the parent window's myChildURL
100 function getChildwindowURL()
101 {
102     if ( childwindow.closed )
103         alert( "You attempted to interact with a closed window" );
104     else
105         myChildURL.value = childwindow.location;
106 } // end function getChildwindowURL
107
108 // set the URL of the child window to the URL
109 // in the parent window's myChildURL
110 function setChildwindowURL()
111 {
112     if ( childwindow.closed )
113         alert( "You attempted to interact with a closed window" );
114     else
115         childwindow.location = myChildURL.value;
116 } // end function setChildwindowURL
117 //-->
118 </script>
119
120 </head>
121
122 <body>
123 <h1>Hello, This is the main window</h1>
```

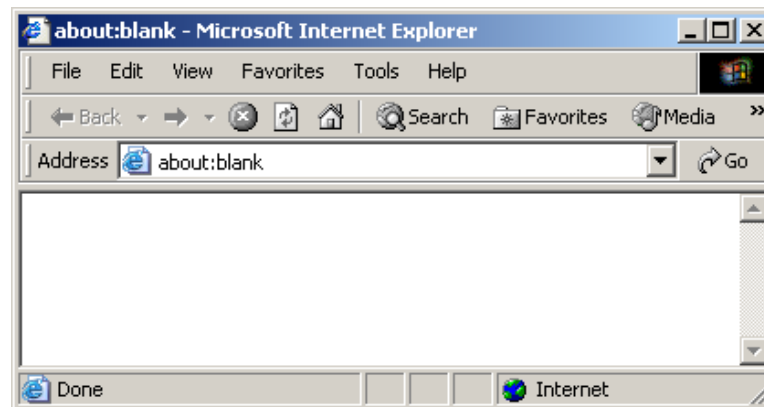
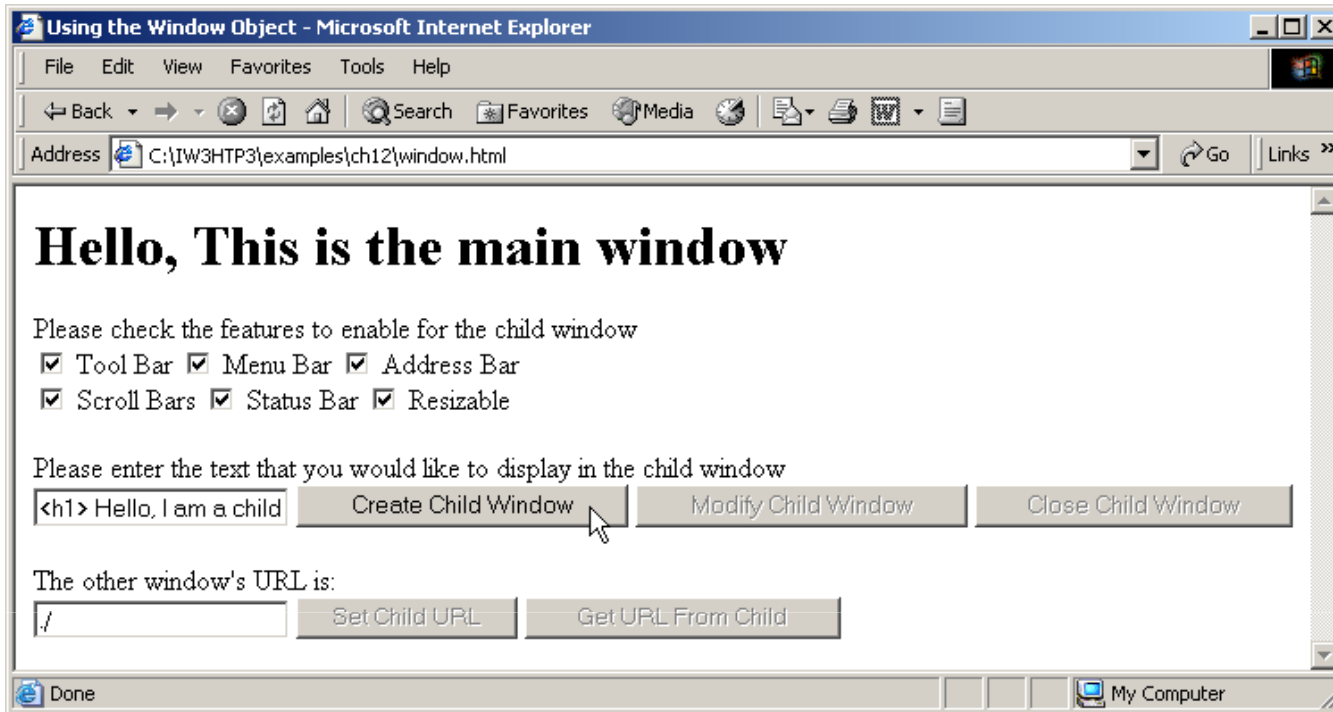
window.html
5 of 7

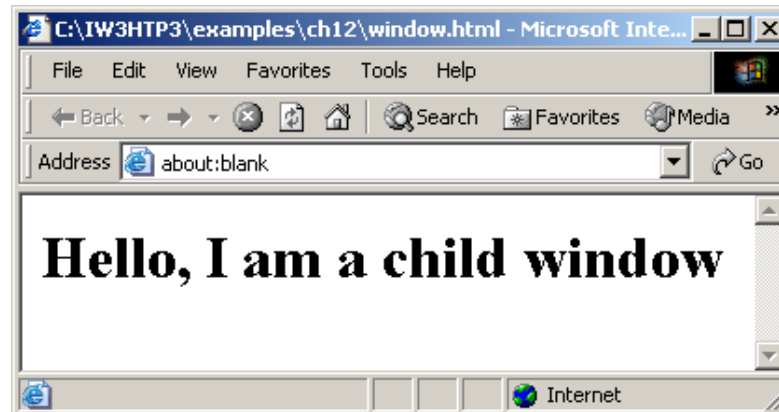
```
124 <p>Please check the features to enable for the child window<br/>
125   <input id = "toolBarCheckBox" type = "checkbox" value = ""
126     checked = "checked" />
127     <label>Tool Bar</label>
128   <input id = "menuBarCheckBox" type = "checkbox" value = ""
129     checked = "checked" />
130     <label>Menu Bar</label>
131   <input id = "locationCheckBox" type = "checkbox" value = ""
132     checked = "checked" />
133     <label>Address Bar</label><br/>
134   <input id = "scrollBarsCheckBox" type = "checkbox" value = ""
135     checked = "checked" />
136     <label>Scroll Bars</label>
137   <input id = "statusCheckBox" type = "checkbox" value = ""
138     checked = "checked" />
139     <label>Status Bar</label>
140   <input id = "resizableCheckBox" type = "checkbox" value = ""
141     checked = "checked" />
142     <label>Resizable</label><br/></p>
143
144 <p>Please enter the text that you would like to display
145   in the child window<br/>
146   <input id = "textForChild" type = "text"
147     value = "<h1> Hello, I am a child window</h1> <br\>" />
```

window.html
6 of 7

```
148 <input id = "createButton" type = "button"
149     value = "Create Child window" onclick = "createChildwindow()" />
150 <input id= "modifyButton" type = "button" value = "Modify Child window"
151     onclick = "modifyChildwindow()" disabled = "disabled"/>
152 <input id = "closeButton" type = "button" value = "Close Child window"
153     onclick = "closeChildwindow()" disabled = "disabled"/></p>
154
155 <p>The other window's URL is: <br/>
156 <input id = "myChildURL" type = "text" value = "./"/>
157 <input id = "setURLButton" type = "button" value = "Set Child URL"
158     onclick = "setChildwindowURL()" disabled = "disabled"/>
159 <input id = "getURLButton" type = "button" value = "Get URL From Child"
160     onclick = "getChildwindowURL()" disabled = "disabled"/></p>
161
162 </body>
163 </html>
```

window.html
7 of 7





12.8 window Object

Method or Property	Description
<code>open(url, name, options)</code>	Creates a new window with the URL of the window set to <i>url</i> , the name set to <i>name</i> , and the visible features set by the string passed in as <i>option</i> .
<code>prompt(prompt, default)</code>	Displays a dialog box asking the user for input. The text of the dialog is <i>prompt</i> , and the default value is set to <i>default</i> .
<code>close()</code>	Closes the current window and deletes its object from memory.
<code>window.focus()</code>	This method gives focus to the window (i.e., puts the window in the foreground, on top of any other open browser windows).
<code>window.document</code>	This property contains the document object representing the document currently inside the window.
<code>window.closed</code>	This property contains a boolean value that is set to true if the window is closed, and false if it is not.
<code>window.opener</code>	This property contains the window object of the window that opened the current window, if such a window exists.

Fig. 12.14 Important `window` object methods and properties.

12.9 Using Cookies

- Cookie
 - Data stored on user's computer to maintain information about client during and between browser sessions
 - Can be accessed through `cookie` property
 - Set expiration date through `expires` property
 - Use `escape` function to convert non-alphanumeric characters to hexadecimal escape sequences
 - `unescape` function converts hexadecimal escape sequences back to English characters

```
1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3   "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
4
5 <!-- Fig. 12.15: cookie.html -->
6 <!-- Using Cookies      -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head>
10     <title>Using Cookies</title>
11
12     <script type = "text/javascript">
13       <!--
14       var now = new Date(); // current date and time
15       var hour = now.getHours(); // current hour (0-23)
16       var name;
17
18       if ( hour < 12 ) // determine whether it is morning
19         document.write( "<h1>Good Morning, " );
20       else
21       {
22         hour = hour - 12; // convert from 24 hour clock to PM time
23
```

cookie.html
1 of 4

```
24     // determine whether it is afternoon or evening
25     if ( hour < 6 )
26         document.write( "<h1>Good Afternoon, " );
27     else
28         document.write( "<h1>Good Evening, " );
29 }
30
31 // determine whether there is a cookie
32 if ( document.cookie )
33 {
34     // convert escape characters in the cookie string to their
35     // english notation
36     var myCookie = unescape( document.cookie );
37
38     // split the cookie into tokens using = as delimiter
39     var cookieTokens = myCookie.split( "=" );
40
41     // set name to the part of the cookie that follows the = sign
42     name = cookieTokens[ 1 ];
43 }
44 else
45 {
46     // if there was no cookie then ask the user to input a name
47     name = window.prompt( "Please enter your name", "GalAnt" );
48 }
```

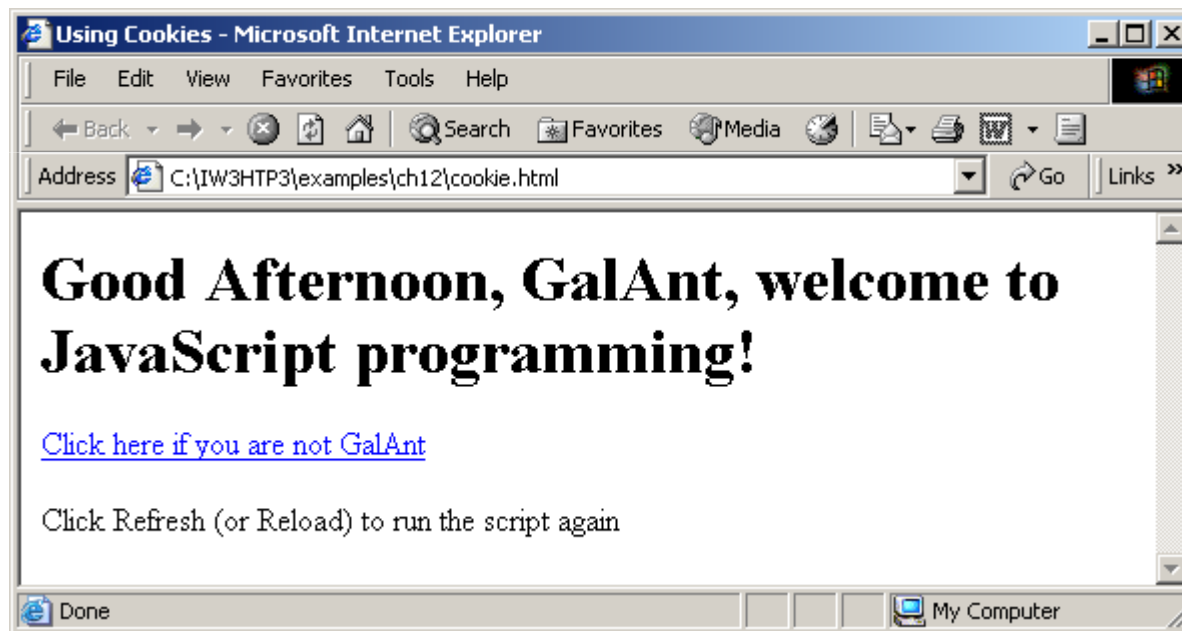
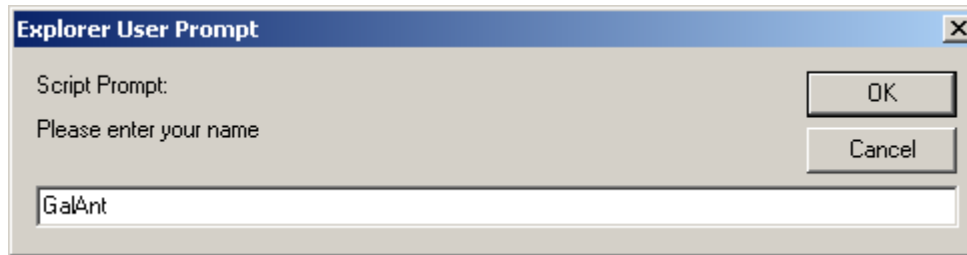
cookie.html
2 of 4

```
49     // escape special characters in the name string
50     // and add name to the cookie
51     document.cookie = "name=" + escape( name );
52 }
53
54 document.writeln(
55     name + ", welcome to JavaScript programming! </h1>" );
56 document.writeln( "<a href= \" JavaScript:wrongPerson() \" > " +
57     "Click here if you are not " + name + "</a>" );
58
59 // reset the document's cookie if wrong person
60 function wrongPerson()
61 {
62     // reset the cookie
63     document.cookie= "name=null;" +
64         " expires=Thu, 01-Jan-95 00:00:01 GMT";
65
66     // after removing the cookie reload the page to get a new name
67     location.reload();
68 }
69
70 // -->
71 </script>
72 </head>
73
```

cookie.html
3 of 4

```
74 <body>
75     <p>Click Refresh (or Reload) to run the script again</p>
76 </body>
77 </html>
```

cookie.html
4 of 4



12.10 Final JavaScript Example

- Combines concepts discussed previously


```

1 <?xml version = "1.0"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3   "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
4
5 <!-- Fig. 12.16: final.html -->
6 <!-- Putting It All Together -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9   <head>
10     <title>Putting It All Together</title>
11
12     <script type = "text/javascript">
13       <!--
14       var now = new Date(); // current date and time
15       var hour = now.getHours(); // current hour
16
17       // array with names of the images that will be randomly selected
18       var pictures =
19         [ "CPE", "EPT", "GPP", "GUI", "PERF", "PORT", "SEO" ];
20
21       // array with the quotes that will be randomly selected
22       var quotes = [ "Form ever follows function.<br/>" +
23         " Louis Henri Sullivan", "E pluribus unum." +
24         " (One composed of many.) <br/> Virgil", "Is it a" +
25         " world to hide virtues in?<br/> William Shakespeare" ];

```

final.html
1 of 6

```
26
27 // write the current date and time to the web page
28 document.write( "<p>" + now.toLocaleString() + "<br/></p>" );
29
30 // determine whether it is morning
31 if ( hour < 12 )
32     document.write( "<h2>Good Morning, " );
33 else
34 {
35     hour = hour - 12; // convert from 24 hour clock to PM time
36
37     // determine whether it is afternoon or evening
38     if ( hour < 6 )
39         document.write( "<h2>Good Afternoon, " );
40     else
41         document.write( "<h2>Good Evening, " );
42 }
43
44 // determine whether there is a cookie
45 if ( document.cookie )
46 {
47     // convert escape characters in the cookie string to their
48     // english notation
49     var myCookie = unescape( document.cookie );
50
```

final.html
2 of 6

```
51 // split the cookie into tokens using = as delimiter
52 var cookieTokens = myCookie.split( "=" );
53
54 // set name to the part of the cookie that follows the = sign
55 name = cookieTokens[ 1 ];
56 }
57 else
58 {
59 // if there was no cookie then ask the user to input a name
60 name = window.prompt( "Please enter your name", "GalAnt" );
61
62 // escape special characters in the name string
63 // and add name to the cookie
64 document.cookie = "name =" + escape( name );
65 }
66
67 // write the greeting to the page
68 document.writeln(
69 name + ", welcome to JavaScript programming!</h2>" );
70
71 // write the link for deleting the cookie to the page
72 document.writeln( "<a href = \" JavaScript:wrongPerson() \" > " +
73 "Click here if you are not " + name + "</a><br/>" );
74
```

final.html
3 of 6

```

75 // write the random image to the page
76 document.write ( "<img src = \"\" +
77     pictures[ Math.floor( Math.random() * 7 ) ] +
78     ".gif\" width= \" 105 \" height= \" 100 \" /> <br/>" );
79
80 // write the random quote to the page
81 document.write ( quotes[ Math.floor( Math.random() * 3 ) ] );
82
83 // create a window with all the quotes in it
84 function allQuotes()
85 {
86     // create the child window for the quotes
87     quotewindow = window.open( "", "", "resizable=yes, toolbar" +
88         "=no, menubar=no, status=no, location=no," +
89         " scrollBars=yes" );
90     quotewindow.document.write( "<p>" )
91
92     // loop through all quotes and write them in the new window
93     for ( var i = 0; i < quotes.length; i++ )
94         quotewindow.document.write( ( i + 1 ) + ".) " +
95         quotes[ i ] + "<br/><br/>");
96

```

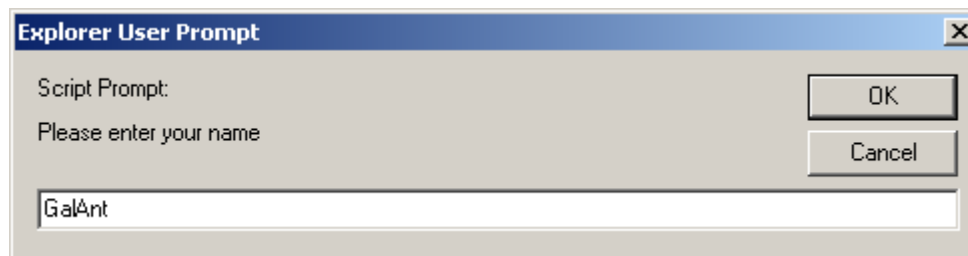
final.html
4 of 6

```
97 // write a close link to the new window
98 quotewindow.document.write( "</p><br/><a href = \" \" +
99     "JavaScript:window.close()\">" +
100     " close this window </a>" )
101 }
102
103 // reset the document's cookie if wrong person
104 function wrongPerson()
105 {
106     // reset the cookie
107     document.cookie= "name=null;" +
108     " expires=Thu, 01-Jan-95 00:00:01 GMT";
109
110     // after removing the cookie reload the page to get a new name
111     location.reload();
112 }
113
114 // open a new window with the quiz2.html file in it
115 function openQuiz()
116 {
117     window.open( "quiz2.html", "", "resizable = yes, " +
118     "toolbar = no, menubar = no, status = no, " +
119     "location = no, scrollbars = no");
120 }
121 // -->
```

final.html
5 of 6

```
122     </script>
123
124 </head>
125
126 <body>
127     <p><a href = "JavaScript:allQuotes()">View all quotes</a></p>
128
129     <p id = "quizSpot">
130         <a href = "JavaScript:openQuiz()">Please take our quiz</a></p>
131
132     <script type = "text/javascript">
133         // variable that gets the last midification date and time
134         var modDate = new Date( document.lastModified );
135
136         // write the last modified date and time to the page
137         document.write ( "This page was last modified " +
138             modDate.toLocaleString() );
139     </script>
140
141 </body>
142 </html>
```

final.html
6 of 6



Putting It All Together - Microsoft Internet Explorer

File Edit View Favorites Tools Help


← Back → Stop Home Search Favorites Media Print

Address <C:\IW3HTP3\examples\ch12\final.html> Go Links >>

Thursday, July 10, 2003 1:29:44 PM

Good Afternoon, GalAnt, welcome to JavaScript programming!

[Click here if you are not GalAnt](#)



Form ever follows function.
Louis Henri Sullivan

[View all quotes](#)

[Please take our quiz](#)

This page was last modified Thursday, July 10, 2003 12:36:26 PM

Done My Computer

C:\IW3HTP3\examples\ch12\final.html - Mic...

- 1.) Form ever follows function.
Louis Henri Sullivan
- 2.) E pluribus unum. (One composed of many.)
Virgil
- 3.) Is it a world to hide virtues in?
William Shakespeare

[Close this window](#)

```
1 <?xml version = "1.0" encoding = "utf-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
3   "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
4
5 <!-- Fig. 12.14: quiz2.html -->
6 <!-- Online Quiz          -->
7
8 <html xmlns = "http://www.w3.org/1999/xhtml">
9 <head>
10 <title>Online Quiz</title>
11
12 <script type = "text/JavaScript">
13   <!--
14   function checkAnswers()
15   {
16     // determine whether the answer is correct
17     if ( myQuiz.radiobutton[ 1 ].checked )
18       window.opener.quizSpot.innerHTML =
19         "Congratulations, your answer is correct";
20     else // if the answer is incorrect
21       window.opener.quizSpot.innerHTML = "Your answer is incorrect." +
22         " Please try again <br /> <a href= \" JavaScript:openQuiz()\" +
23         \" \" > Please take our quiz</a>";
24
25     window.opener.focus();
```

quiz2.html
1 of 3


```
26     window.close();
27 } // end checkAnswers function
28 //-->
29 </script>
30
31 </head>
32
33 <body>
34     <form id = "myQuiz" action = "JavaScript:checkAnswers()">
35         <p>Select the name of the tip that goes with the image shown:<br />
36         <img src = "EPT.gif" width = "108" height = "100"
37             alt = "mystery tip"/>
38         <br />
39
40         <input type = "radio" name = "radiobutton" value = "CPE" />
41         <label>Common Programming Error</label>
42
43         <input type = "radio" name = "radiobutton" value = "EPT" />
44         <label>Error-Prevention Tip</label>
45
46         <input type = "radio" name = "radiobutton" value = "PERF" />
47         <label>Performance Tip</label>
48
49         <input type = "radio" name = "radiobutton" value = "PORT" />
50         <label>Portability Tip</label><br />
```


quiz2.html
2 of 3

```
51
52     <input type = "submit" name = "Submit" value = "Submit" />
53     <input type = "reset" name = "reset" value = "Reset" />
54 </p>
55 </form>
56 </body>
57 </html>
```

quiz2.html
3 of 3

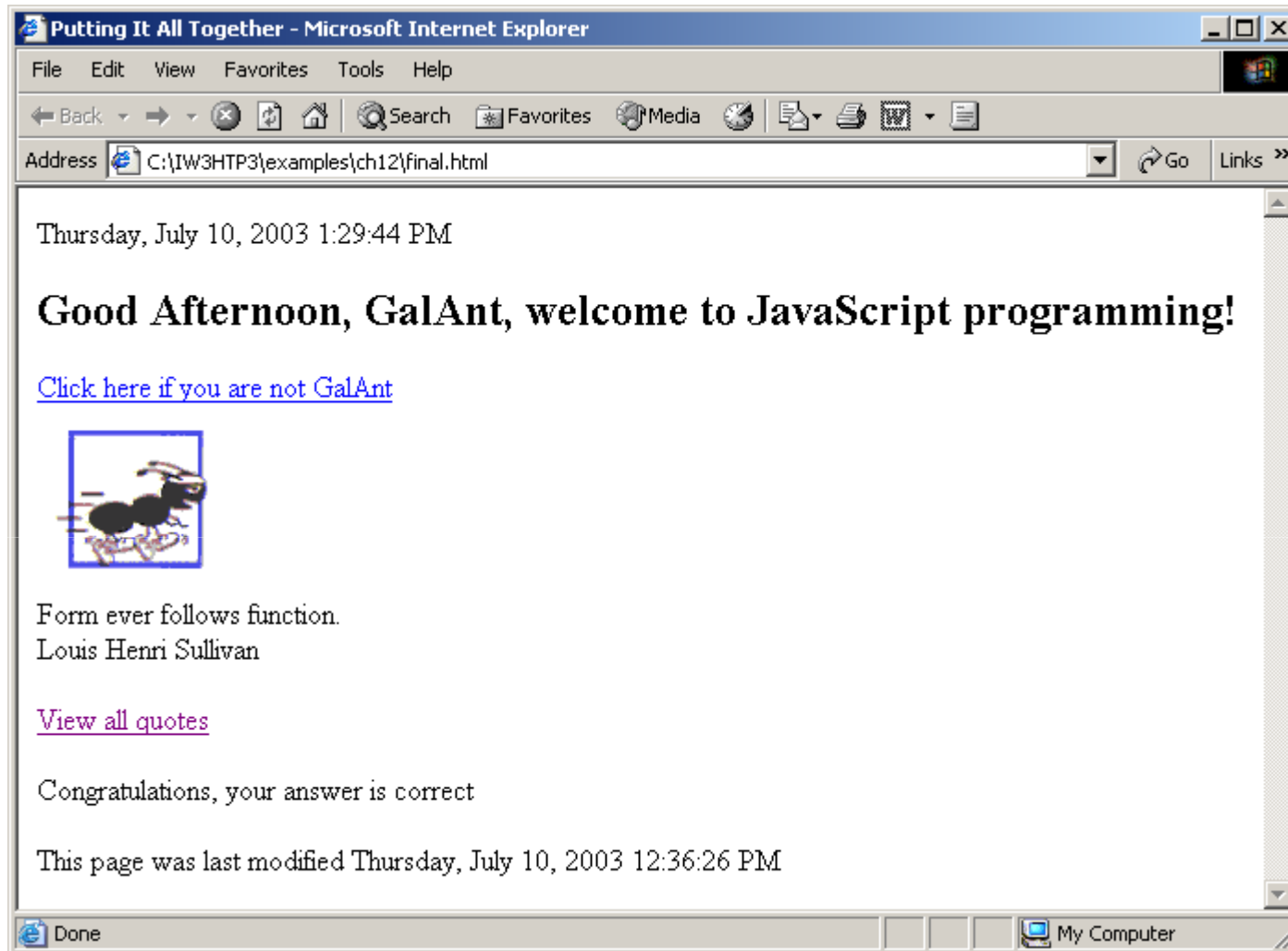
Online Quiz - Microsoft Internet Explorer

Select the name of the tip that goes with the image shown:



Common Programming Error Error-Prevention Tip Performance Tip Portability Tip

Submit Reset



12.11 Web Resources

- www.javascript.com
- www.iboost.com/build/programming/js/tutorial/885.htm
- www.javascriptsearch.com
- www.a1javascripts.com