

Tutorial 5

Working with Forms and Regular Expressions

Objectives

- Understand how to reference form element objects
- Extract data from input fields, selection lists, and option button groups
- Create a calculated field
- Understand the principles of form validation
- Perform a client-side validation

Objectives

- Work with string objects
- Work with regular expressions
- Apply regular expressions to zip code fields
- Validate credit card numbers

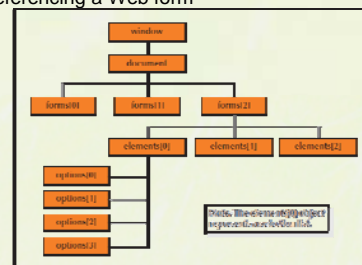
Working with Forms and Fields

Working with Forms and Fields

- Referencing a Web form
 - You have to work with the properties and methods of the form object and the elements it contains
 - JavaScript supports an object collection for forms
`document.forms[idref]` or
`document.forms["form1"]`

Working with Forms and Fields

- Referencing a Web form



Working with Forms and Fields

- Referencing a form element
 - The elements within a form are organized into an elements collection
 - Can reference a form element either by its position in the collection or by its name or id attributes

New Perspectives on JavaScript and AJAX, 2nd Edition

Working with Forms and Fields

- Referencing a form element

Object	Reference
order form	document.form[0]
date field	document.form[0].date
product selection list	document.form[0].prod
quantity selection list	document.form[0].qty
price of the product field	document.form[0].price
group of shipping options	document.form[0].shipType
shipping cost field	document.form[0].ship
subtotal field	document.form[0].sub
tax field	document.form[0].tax
total field	document.form[0].tot
cancel button	document.form[0].cancelb
next button	document.form[0].nextb

New Perspectives on JavaScript and AJAX, 2nd Edition

Working with Input Fields

- Setting the field value
 - To set the value contained in a field such as an input box, you use the value property
 - The general syntax is:


```
formObject.element.value = fieldvalue;
```

New Perspectives on JavaScript and AJAX, 2nd Edition

Working with Input Fields

- Setting the field value

Property	Description
defaultvalue	The default value that is initially displayed in the field
form	References the form containing the field
maxlength	The maximum number of characters allowed in the field
name	The name of the field
size	The width of the input field in characters
type	The type of input field (button, check box, file, hidden, image, password, radio, reset, submit, text)
value	The current value of the field

Method	Description
blur()	Remove the focus from the field
focus()	Give focus to the field
select()	Select the field

New Perspectives on JavaScript and AJAX, 2nd Edition

Working with Input Fields

- Navigating between fields
 - To place the cursor in a particular field on a form, use:


```
formObject.element.focus();
```
 - To remove the focus from this field, use:


```
formObject.element.blur();
```

New Perspectives on JavaScript and AJAX, 2nd Edition

Working with Selection Lists

- To reference a particular option in the collection, use:


```
selection.options[idref]
```

object	object.text	object.value
document.form[0].prod.options[0]	Products from GPS-ware	0
document.form[0].prod.options[1]	GoMap 1.0 (\$19.95)	19.95
document.form[0].prod.options[2]	Drive Planner 2.0 (\$29.95)	29.95
document.form[0].prod.options[3]	Hiker 1.0 (\$29.95)	29.95
document.form[0].prod.options[4]	G-Receiver 1 (\$149.50)	149.50
document.form[0].prod.options[5]	G-Receiver 1 (\$199.50)	199.50
document.form[0].prod.options[6]	G-Receiver 1 (\$249.50)	249.50

New Perspectives on JavaScript and AJAX, 2nd Edition

Working with Selection Lists

selection list	Property	Description
	length	The number of options in the list
	name	The name of the selection list
	options	The collection of options in the list
selection list option	selectedIndex	The index number of the currently selected option in the list
	defaultSelected	A Boolean value indicating whether the option is selected by default
	index	The index value of the option
	selected	A Boolean value indicating whether the option is currently selected
	text	The text associated with the option
	value	The value associated with the option

13

New Perspectives on
JavaScript and AJAX, 2nd Edition

Working with Selection Lists

```
function calculate() {
    price = document.forms[0].price.value;
    product = product.selectedIndex;
    productPrice = product.options[product].value;
    quantity = document.forms[0].qty.value;
    quantityOrdered = quantity * productPrice;
    document.forms[0].price.value = productPrice * quantityOrdered;
}
```

code to return the price of the selected product

code to return the index of the selected quantity

and is equal to the product price multiplied by the quantity ordered

14

New Perspectives on
JavaScript and AJAX, 2nd Edition

Working with Option Buttons and Check Boxes

- Using option buttons
 - Option buttons have the reference `options[idref]`
 - Where options is the reference to the group of option buttons and idref is either the index number or id of the individual option button

Property	Description
checked	A Boolean value indicating whether the option button is currently selected
defaultChecked	A Boolean value indicating whether the option button is selected by default
name	The name of the option button
value	The value associated with the option button

15

New Perspectives on
JavaScript and AJAX, 2nd Edition

Working with Option Buttons and Check Boxes

- Using the "this" keyword
 - The **this** keyword is a JavaScript object reference that refers to the currently selected object, whatever that may be

```
document.forms[0].price.value = productPrice * quantityOrdered;
function calculate() {
    document.forms[0].qty.value = this.value;
}
```

the this keyword references the currently selected object

16

New Perspectives on
JavaScript and AJAX, 2nd Edition

Working with Option Buttons and Check Boxes

- Working with check boxes
 - Work the same way as option buttons
 - In addition, the value associated with a check box is stored in the value property of the check box object
 - This value is applied only when the check box is checked
 - When unchecked, its field has no value assigned to it

17

New Perspectives on
JavaScript and AJAX, 2nd Edition

Creating Calculated Fields

- Converting between text strings and numeric values
 - Explicitly indicate that you want to convert `parseFloat(text)`

18

New Perspectives on
JavaScript and AJAX, 2nd Edition

Working with Form Validation

- **Form validation** is a process by which the server or user's browser checks a form for data entry errors
- With **server-side validation**, a form is sent to the Web server for checking
- In **client-side validation**, the Web browser checks the form, which is not submitted to the server until it passes inspection

19

New Perspectives on
JavaScript and AJAX, 2nd Edition

Working with Form Validation

- Submitting a Form
 - To control this submission process, JavaScript provides the `onsubmit` event handler
- ```
formobj.onsubmit = function;
```
- If the function returns a value of `false`, the submit event is cancelled, while a value of `true` allows the submit event to continue unabated

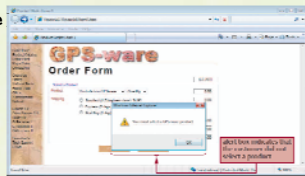
20

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Working with Form Validation

- Alerting the user
  - An **alert box** is a dialog box that displays an informative message to the user along with an OK button

```
alert(message)
```



21

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Working with Text Strings

- The string object
    - JavaScript treats each text string as an object called a **string object**
    - The most common way to create a string object is to assign a text string to a variable
    - You can also create a string object using the object constructor
- ```
stringVariable = new String("text")
```

22

New Perspectives on
JavaScript and AJAX, 2nd Edition

Working with Text Strings

- Calculating the length of a text string
 - The following code calculates the number of characters in the `stringVar` variable, storing the value 17 in the `lengthValue` variable
- ```
stringVar = "GPS-ware Products";
lengthValue = stringVar.length
```

23

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Working with Text Strings

- Working with the string object methods
  - To determine the number of characters in a text string, use the object property `string.length`
  - To extract a character from a text string, use the method `string.charAt(i)`
  - To extract a substring from a text string, use the method `string.slice(start, end)`

24

New Perspectives on  
JavaScript and AJAX, 2nd Edition



## Working with Text Strings

- Working with the string object methods
  - To split a string into several substrings, use the command `strArray = string.split(str)`
  - To search a string, use the method `string.indexOf(str, start)`

25

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Working with Text Strings

- Formatting text strings

| Method                            | Description                                                | HTML Equivalent                                                                                                                                                                                                                         |
|-----------------------------------|------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>string.anchor()</code>      | Creates an anchor with the anchor name text                | <code>&lt;a name="text"&gt;text&lt;/a&gt;</code>                                                                                                                                                                                        |
| <code>string.big()</code>         | Changes the size of the string font to big                 | <code>&lt;big&gt;text&lt;/big&gt;</code>                                                                                                                                                                                                |
| <code>string.blink()</code>       | Changes string to blinking text                            | <code>&lt;blink&gt;text&lt;/blink&gt;</code>                                                                                                                                                                                            |
| <code>string.bold()</code>        | Changes the font weight of string to bold                  | <code>&lt;b&gt;text&lt;/b&gt;</code>                                                                                                                                                                                                    |
| <code>string.fixed()</code>       | Changes the font of string to a fixed-width font           | <code>&lt;code&gt;text&lt;/code&gt;</code>                                                                                                                                                                                              |
| <code>string.fontcolor()</code>   | Changes the color of string to the hexadecimal color value | <code>&lt;span style="color:hex"&gt;text&lt;/span&gt;</code>                                                                                                                                                                            |
| <code>string.fontSize()</code>    | Changes the font size of string to value                   | <code>&lt;span style="font-size:value"&gt;text&lt;/span&gt;</code>                                                                                                                                                                      |
| <code>string.italic()</code>      | Changes string to italic                                   | <code>&lt;i&gt;text&lt;/i&gt;</code>                                                                                                                                                                                                    |
| <code>string.itali()</code>       | Changes string to a line pointing to left                  | <code>&lt;del&gt;text&lt;/del&gt;</code>                                                                                                                                                                                                |
| <code>string.left()</code>        | Changes the size of the string font to small               | <code>&lt;small&gt;text&lt;/small&gt;</code>                                                                                                                                                                                            |
| <code>string.reverse()</code>     | Reverses the string                                        | <code>&lt;div style="writing-mode: vertical-rl; transform: rotate(180deg);"&gt;text&lt;/div&gt;</code>                                                                                                                                  |
| <code>string.substr()</code>      | Changes string to a substring                              | <code>&lt;div style="display: inline-block; width: 100px; height: 100px; background-color: #ccc; border: 1px solid #000; vertical-align: middle; text-align: center; line-height: 100px; margin: 5px 0 0 0;"&gt;text&lt;/div&gt;</code> |
| <code>string.toUpperCase()</code> | Changes string to uppercase letters                        | <code>&lt;span style="text-transform: uppercase;"&gt;text&lt;/span&gt;</code>                                                                                                                                                           |

26

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Introducing Regular Expressions

- A **regular expression** is a text string that defines a character pattern
- One use of regular expressions is **pattern-matching**, in which a text string is tested to see whether it matches the pattern defined by a regular expression

27

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Introducing Regular Expressions

- Creating a regular expression**
  - You create a regular expression in JavaScript using the command  
`re = /pattern/;`
  - This syntax for creating regular expressions is sometimes referred to as a **regular expression literal**

28

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Introducing Regular Expressions

- Matching a substring**
  - The most basic regular expression consists of a substring that you want to locate in the test string
  - The regular expression to match the first occurrence of a substring is `/chars/`

29

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Introducing Regular Expressions

- Setting regular expression flags**
  - To make a regular expression not sensitive to case, use the regular expression literal `/pattern/i`
  - To allow a global search for all matches in a test string, use the regular expression literal `/pattern/g`

30

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Introducing Regular Expressions

- Defining character positions

| Character | Description                                | Example                                                    |
|-----------|--------------------------------------------|------------------------------------------------------------|
| ^         | Indicates the beginning of the text string | /^GPS/ matches "GPS-ware" but not "Products from GPS-ware" |
| \$        | Indicates the end of the text string       | /ware\$/ matches "GPS-ware" but not "GPS-ware Products"    |
| \b        | Indicates the presence of a word boundary  | /bart/ matches "art" and "artists" but not "dart"          |
| \b        | Indicates the absence of a word boundary   | /art\b/ matches "dart" but not "artist"                    |

31

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Introducing Regular Expressions

- Defining character positions

| Character | Description                                                                           | Example                                          |
|-----------|---------------------------------------------------------------------------------------|--------------------------------------------------|
| \d        | A digit (from 0 to 9)                                                                 | /dth/ matches "5th" but not "ath"                |
| \D        | A non-digit                                                                           | /Dth/ matches "at" but not "5t"                  |
| \w        | A word character (an upper- or lowercase letter, a digit, or an underscore)           | /whe/ matches "to" or "A1" but not "\$x" or "++" |
| \W        | A non-word character                                                                  | /W/ matches "\$" or "&" but not "a", "B", or "3" |
| \s        | A white space character (a blank space, tab, new line, carriage return, or form feed) | /s/dh/ matches " 5" but not "5"                  |
| \S        | A non-white space character                                                           | /S/dh/ matches "345" or "a5b" but not "5"        |
| .         | Any character                                                                         | /./ matches anything                             |

32

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Introducing Regular Expressions

- Defining character positions
  - Can specify a collection of characters known as a **character class** to limit the regular expression to only a select group of characters

33

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Introducing Regular Expressions

- Defining character positions

| Character      | Description                                              | Example                                                         |
|----------------|----------------------------------------------------------|-----------------------------------------------------------------|
| [chars]        | Match any character in the list of characters, chars     | /[a-z]/ matches "ap" and "z"                                    |
| [^chars]       | Do not match any character in chars                      | /[^a-z]/ matches neither "ap" nor "z"                           |
| [char1-charN]  | Match characters in the range char1 through charN        | /[a-c]/ matches the lowercase letters a through c               |
| [^char1-charN] | Do not match characters in the range char1 through charN | /[^a-c]/ does not match the lowercase letters a through c       |
| [a-z]          | Match lowercase letters                                  | /[a-z]{2}/ matches any two consecutive lowercase letters        |
| [A-Z]          | Match uppercase letters                                  | /[A-Z]{2}/ matches any two consecutive uppercase letters        |
| [a-zA-Z]       | Match letters                                            | /[a-zA-Z]{2}/ matches any two consecutive letters               |
| [0-9]          | Match digits                                             | /[0-9]{2}/ matches the numbers "10" through "99"                |
| [0-9a-zA-Z]    | Match digits and letters                                 | /[0-9a-zA-Z]{2}/ matches any two consecutive letters or numbers |

34

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Introducing Regular Expressions

- Repetition characters

| Repetition Character(s) | Description                                      | Example                                                   |
|-------------------------|--------------------------------------------------|-----------------------------------------------------------|
| *                       | Repeat 0 or more times                           | /s*/ matches 0 or more consecutive white space characters |
| ?                       | Repeat 0 or 1 time                               | /color?/ matches "color" or "colour"                      |
| +                       | Repeat 1 or more times                           | /s+/ matches 1 or more consecutive white space characters |
| {n}                     | Repeat exactly n times                           | /d{9}/ matches a nine-digit number                        |
| {n,}                    | Repeat at least n times                          | /d{9,}/ matches a number with at least nine digits        |
| {n,m}                   | Repeat at least n times but no more than m times | /d{5,9}/ matches a number with 5 to 9 digits              |

35

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Introducing Regular Expressions

- Escape Sequences
  - An **escape sequence** is a special command inside a text string that tells the JavaScript interpreter not to interpret what follows as a character
  - The character which indicates an escape sequence in a regular expression is the backslash character \

36

New Perspectives on  
JavaScript and AJAX, 2nd Edition

## Introducing Regular Expressions

### • Escape Sequences

| Escape Sequence | Represents             | Example                                                            |
|-----------------|------------------------|--------------------------------------------------------------------|
| \               | The \ character        | /d\d/ matches "5/9" or "3/1" but not "59" or "31"                  |
| \\              | The \ character        | /d\\d/ matches "5/9" or "3/1" but not "59" or "31"                 |
| \.              | The . character        | /d\d\./ matches "3.20" or "5.95" but not "320" or "595"            |
| \*              | The * character        | /d\d*/ matches "help" or "jazz"                                    |
| \+              | The + character        | /d\d+/ matches "5+9" or "3+1" but not "59" or "39"                 |
| \?              | The ? character        | /d\d?/ matches "help" or "info"                                    |
| \               | The   character        | /d\d / matches "d" or "5"                                          |
| \[ ]            | The [ and ] characters | /d[3]d/ matches "800" or "555"                                     |
| \{ }            | The { and } characters | /d{2,4}d/ matches "pass" or "info"                                 |
| \^              | The ^ character        | /d\d\d^/ matches "3212" or "453"                                   |
| \\$             | The \$ character       | /d\d\d\$/ matches "559.95" or "519.50"                             |
| \n              | A newline              | /n/ matches the occurrence of a newline in the text string         |
| \r              | A carriage return      | /r/ matches the occurrence of a carriage return in the text string |
| \t              | A tab                  | /t/ matches the occurrence of a tab in the text string             |

New Perspectives on  
JavaScript and AJAX, 2nd Edition

37

## Introducing Regular Expressions

### • The regular expression object

| Method                                     | Description                                                                                                                                                                                                |
|--------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>re.compile(pattern, flags)</code>    | Compiles or recompiles a regular expression <code>re</code> , where <code>pattern</code> is the text string of the new regular expression pattern and <code>flags</code> are flags applied to the pattern. |
| <code>re.exec(string)</code>               | Executes a search on <code>string</code> using the regular expression <code>re</code> ; pattern results are returned in an array and reflected in the properties of the global <code>RegExp</code> object. |
| <code>re.match(string)</code>              | Performs a pattern match in <code>string</code> using the <code>re</code> regular expression; matched substrings are stored in an array.                                                                   |
| <code>string.replace(re, newSubstr)</code> | Replaces the substring defined by the regular expression <code>re</code> in the text string <code>string</code> with <code>newSubstr</code> .                                                              |
| <code>string.search(re)</code>             | Searches <code>string</code> for a substring matching the regular expression <code>re</code> ; returns the index of the match, or -1 if no match is found.                                                 |
| <code>string.split(re)</code>              | Splits <code>string</code> at each point indicated by the regular expression <code>re</code> ; the substrings are stored in an array.                                                                      |
| <code>re.test(string)</code>               | Performs a pattern match on the text string <code>string</code> using the regular expression <code>re</code> , returning the Boolean value true if a match is found and false otherwise.                   |

New Perspectives on  
JavaScript and AJAX, 2nd Edition

38

## Working with the Regular Expression Object

### • Validating a zip code

code to call the `checkZipRE()` function

```
else if (document.forms[0].city.value.length == 0) {
 alert("Please enter a city name.");
 return false;
} else if (document.forms[0].zip.value == false) {
 alert("Invalid zip code.");
 return false;
} else return true;
```

```
function checkZipRE(zip) {
 re = /^\d{5}(-\d{4})?$/;
 return re.test(zip);
}
```

regular expression for a U.S. zip code

New Perspectives on  
JavaScript and AJAX, 2nd Edition

39

## Validating Financial Information

### • Removing blank spaces from credit card numbers

```
function testCard(card) {
 var card = card;
 for (var i = 0; i < card.length; i++) {
 if (card.charAt(i) != " ") {
 card = card.replace(" ", "");
 }
 }
 return card;
}
```

regular expression that matches all white space in the text string

replaces each white space character with an empty text string

New Perspectives on  
JavaScript and AJAX, 2nd Edition

40

## Validating Financial Information

### • Validating credit card number patterns

| Credit Card      | Number Pattern                                                                                                   | Regular Expression               |
|------------------|------------------------------------------------------------------------------------------------------------------|----------------------------------|
| American Express | Starts with 34 or 37 followed by 13 other digits                                                                 | /^3[47]\d{13}\$/                 |
| Diners Club      | Starts with 300, 301, 302, 303, 304, or 305 followed by 11 digits, or starts with 36 or 38 followed by 12 digits | /^3[0-5]\d{11}\$ ^3[68]\d{12}\$/ |
| Discover         | Starts with 6011 followed by 12 other digits                                                                     | /^6011\d{12}\$/                  |
| MasterCard       | Starts with 51, 52, 53, 54, or 55 followed by 14 other digits                                                    | /^5[1-5]\d{14}\$/                |
| Visa             | Starts with a 4 followed by 12 or 15 other digits                                                                | /^4[0-9]\d{12,15}\$/             |

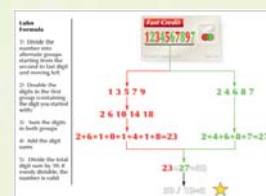
New Perspectives on  
JavaScript and AJAX, 2nd Edition

41

## Validating Financial Information


### • The Luhn Formula

- All credit card numbers must satisfy the **Luhn Formula**, or **Mod10 algorithm**, which is a formula developed by a group of mathematicians in the 1960s to provide a quick validation check on an account number by adding up the digits in the number



New Perspectives on  
JavaScript and AJAX, 2nd Edition

42




## Passing Data Between Forms

- Appending Form Data
  - Text strings can be appended to any URL by adding the ? character to the Web address followed by the text string
    - Use the get method
  - One property of the location object is the location.search property, which contains the text of any data appended to the URL, including the ? character

43

*New Perspectives on  
JavaScript and AJAX, 2nd Edition*




## Passing Data from a Form

- Appending data to a URL
  - There are several limitations to the technique of appending data to a URL
  - URLs are limited in their length
  - Characters other than letters and numbers cannot be passed in the URL without modification
  - URLs cannot contain blank spaces, for example, a blank space is converted to the character code %20

44

*New Perspectives on  
JavaScript and AJAX, 2nd Edition*



## Passing Data from a Form

- Appending and retrieving form data
  - Can use the technique of appending data to the URL with Web forms, too
  - Do this by setting a form's action attribute to the URL of the page to which you want to pass the data, and setting the method of the form to "get"

45

*New Perspectives on  
JavaScript and AJAX, 2nd Edition*



## Passing Data from a Form

- Appending and retrieving form data
  - Use the location.search property and the slice() method to extract only the text string of the field names and values
  - Use the unescape() function to remove any escape sequences characters from the text string
  - Convert each occurrence of the + symbol to a blank space
  - Split the text string at every occurrence of an = or & character, storing the substrings into an array

46

*New Perspectives on  
JavaScript and AJAX, 2nd Edition*