

# *JavaScript*

## *Chapter 5*

### *Validating Form Data with JavaScript*

*Lecturer: Thiều Quang Trung*

# Objectives

- Study form elements and objects
- Use JavaScript to manipulate and validate form elements
- Learn how to submit and reset forms
- Learn how to validate submitted form data

# Overview of Forms

- Many Web sites use **forms**
  - Collect information from users and transmit to a server for processing
- Forms frequently found on Web pages gather search criteria from a user
  - Data collected is transmitted to a server-side scripting language program on a Web server
- Popular server-side scripting languages
  - PHP, Common Gateway Interface (CGI), Active Server Pages (ASP), and Java Server Pages (JSP)

# Overview of Forms (continued)

**Figure 5-1**

MySpace.com sign-up form

The screenshot shows a Mozilla Firefox browser window with the title "Myspace.com - Mozilla Firefox". The address bar displays "http://signup.myspace.com/index.cfm?fusea...". The page content includes a sign-up form titled "JOIN MYSPACE HERE!". The form fields are: Email Address (text input), First Name (text input), Last Name (text input), Password (text input), Confirm Password (text input), Country (dropdown menu showing "United States"), Postal Code (text input), Gender (radio buttons for "Female" and "Male"), Date Of Birth (Month, Day, and Year dropdown menus), Preferred Site & Language (dropdown menu showing "www.myspace.com - English"), and a checkbox for "Allow others to see when it's my birthday" which is checked. At the bottom of the form, there is a checkbox for "By checking the box you agree to the MySpace Terms of" which is unchecked. To the right of the form, there is a sidebar with the heading "why JOIN M..." and a list of links: "» Create a...", "» Upload Pi...", "» Send Mail...", "» Write Blo...", "» Comment...", and "» It's FREE". Below the links, there is a box with the text "MySpace unders...", "privacy is the ke...", "Already a memb...", and "Please read our...". The status bar at the bottom of the browser window shows "Transferring data from 04.myspace.presence.userplane.com..."

Already a member? Click Here to Log In

**JOIN MYSPACE HERE!**

Email Address:

First Name:

Last Name:

Password:

Confirm Password:

Country:

Postal Code:

Gender: ☐ Female ☐ Male

Date Of Birth:

☒ Allow others to see when it's my birthday

Preferred Site & Language:

☐ By checking the box you agree to the MySpace Terms of

why JOIN M...

- » Create a
- » Upload Pi
- » Send Mail
- » Write Blo
- » Comment
- » It's FREE

MySpace unders  
privacy is the ke  
Already a memb  
Please read our

Transferring data from 04.myspace.presence.userplane.com...

# Overview of Forms (continued)

**Figure 5-2**

Barnes & Noble.com  
advanced search page

The screenshot shows the Barnes & Noble.com advanced search page in a Mozilla Firefox browser window. The address bar shows the URL `http://www.barnesandnoble.com/search.asp?z=y`. The page features a navigation bar with links to various categories like HOME, BOOKS, USED & OUT OF PRINT, TEXTBOOKS, DVD, MUSIC, PC & VIDEO GAMES, CHILDREN'S, TOYS & GAMES, HOME & OFFICE, GIFT CARDS, and B&N MEMBER PROGRAM. A search bar with a dropdown menu set to 'Keyword' and a 'SEARCH' button is present. Below the search bar, there's a section titled 'Search: Books' with a prompt 'Fill in one or more of the fields below:'. This section includes input fields for 'Title of Book', 'Authors Name' (which contains the text 'Don Gosselin'), and 'Keywords'. There are 'Search' and 'Clear Fields' buttons, along with a link to 'Search Tips'. Below this, a section titled 'You can narrow your search by selecting one or more options below:' contains two dropdown menus for 'Price' and 'Format'. The page also includes a sidebar with a 'Barnes & Noble Gift Cards' advertisement and a footer with a 'Done' button.

# Understanding Form Elements and Objects

- Learn about the primary form elements and how to access them with JavaScript

# The `<form>` Element

- **`<form>` element**
  - Designates a form within a Web page and contains all the text and elements that make up a form
- Can set up a form to send data to an e-mail address
  - Replace the Web server script's URL in the `<form>` element's action attribute with the `mailto` protocol

# The `<form>` Element (continued)

Attribute	Description
<code>accept-charset</code>	Specifies a comma-separated list of possible character sets that the form supports
<code>action</code>	Required attribute that specifies a URL to which form data is submitted If this attribute is excluded, the data is sent to the URL that contains the form Typically you would specify an e-mail address or the URL of a program on a server
<code>enctype</code>	Specifies the MIME type of the data being submitted The default value is <code>application/x-www-form-urlencoded</code>
<code>method</code>	Determines how form data is submitted: the two options for this attribute are "get" and "post"; the default option, "get", appends form data as one long string to the URL specified by the <code>action</code> attribute; the "post" option sends form data as a transmission separate from the URL specified by the <code>action</code> attribute; although "get" is the default, "post" is considered the preferred option, because it allows the server to receive the data separately from the URL

**Table 5-1** Attributes of the `<form>` element



# Form Controls

- Primary elements used within the `<form>` element
  - `<input>`, `<button>`, `<select>`, and `<textarea>`
- `<input>` and `<button>` elements
  - Create input fields with which users interact
- `<select>` element
  - Displays choices in drop-down menu or scrolling list
- `<textarea>` element
  - Create a text field in which users can enter multiple lines of information

# Form Controls (continued)

- **Field**
  - Any form element into which a user can enter data or that a user can select or change
- `<input>`, `<textarea>`, and `<select>` elements can include `name` and `value` attributes
  - `name` attribute defines a name for an element
  - `value` attribute defines a default value

# Using JavaScript with Forms

- **Form object**
  - Represents a form on a Web page
  - Used in JavaScript to access form controls and verify form information
  - Part of the browser object model
- Referencing Forms and Form Elements
  - `Document` object includes a `forms[]` array that contains all the forms on a Web page
  - The `<form>` element's `name` attribute is deprecated in XHTML
  - `Form` object has an `elements[]` array

# Using JavaScript with Forms (continued)

- Referencing Forms and Form Elements (continued)
  - **elements[] array**
    - Contains objects representing each control in a form
  - Reference the index number of the form in the `forms[] array`
    - Followed by the appropriate element index number from the `elements[] array`
- The `Form` Object
  - See Tables 5-2, 5-3, and 5-4
- Example: The Gosselin Gazette Web page

# Using JavaScript with Forms (continued)

Property	Description
<code>acceptCharset</code>	Returns a comma-separated list of possible character sets that the form supports
<code>action</code>	Returns the URL to which form data is submitted
<code>elements[]</code>	Returns an array of a form's elements
<code>enctype</code>	Sets or returns a string representing the MIME type of the data being submitted
<code>length</code>	Returns an integer representing the number of elements in the form
<code>method</code>	Sets or returns a string representing one of the two options for submitting form data: "get" or "post"
<code>name</code>	Sets or returns the value assigned to the form's name attribute
<code>target</code>	Sets or returns the target window where responses are displayed after submitting the form

**Table 5-2** Form object properties

# Using JavaScript with Forms (continued)

Event	Description
reset	Executes when a form's reset button is clicked
submit	Executes when a form's submit button is clicked

**Table 5-3** Form object events

Method	Description
reset()	Resets a form without the use of a reset button
submit()	Submits a form without the use of a submit button

**Table 5-4** Form object methods

# Working with Input Fields

- Empty `<input>` element
  - Generate **input fields** that create interface elements
    - Such as text boxes, radio buttons, and so on
- **Minimized form**
  - When a Boolean attribute is not assigned a value
  - Illegal in XHTML
- **Full form** of a Boolean attribute
  - Created by assigning the name of the attribute itself as the attribute's value

# Working with Input Fields (continued)

Attribute	Description
accept	Determines the MIME type of a document that is uploaded with a file box
alt	Provides alternate text for an image submit button
checked	Determines whether or not a radio button or a check box is selected; a Boolean attribute
disabled	Disables a control
maxlength	Accepts an integer value that determines the number of characters that can be entered into a field
name	Designates a name for the element; part of the name=value pair that is used to submit data to a Web server
readonly	Prevents users from changing values in a control
size	Accepts an integer value that determines the width of a text box in characters
src	Specifies the URL of an image
type	Specifies the type of element to be rendered; <code>type</code> is a required attribute; valid values are text, password, radio, check box, reset, button, submit, image, file, and hidden
value	Sets an initial value in a field or a label for buttons; part of the name=value pair that is used to submit data to a Web server

**Table 5-5** Attributes of the `<input>` element



# Input Field Objects

- For controls created with an `<input>` element
  - Each control is represented by an object that is similar to the name of the control
    - `Input`
    - `Radio`
    - `Checkbox`

# Input Field Objects (continued)

Property	Description	Form controls
accept	Sets or returns a comma-separated list of MIME types that can be uploaded	File boxes
accessKey	Sets or returns a keyboard shortcut that users can press to jump to a control, or select and deselect a control	Check boxes, radio buttons, reset buttons, submit buttons, image submit buttons, text boxes, password boxes, file boxes, hidden text boxes
alt	Sets or returns alternate text for an image	Image submit buttons
checked	Sets or returns the checked status of a check box or radio button	Check boxes, radio buttons
defaultChecked	Determines the control that is checked by default in a check box group or radio button group	Check boxes, radio buttons
defaultValue	Sets or returns the default text that appears in a form control	Text boxes, password boxes, file boxes
disabled	Sets or returns a Boolean value that determines whether a control is disabled	Check boxes, radio buttons, reset buttons, submit buttons, image submit buttons, text boxes, password boxes, file boxes, hidden text boxes
form	Returns a reference to the form that contains the control	Check boxes, radio buttons, reset buttons, submit buttons, image submit buttons, text boxes, password boxes, file boxes, hidden text boxes
maxLength	Sets or returns the maximum number of characters that can be entered into a field	Text boxes, password boxes
name	Sets or returns the value assigned to the element's name attribute	Check boxes, radio buttons, reset buttons, submit buttons, image submit buttons, text boxes, password boxes, file boxes, hidden text boxes
readOnly	Sets or returns a Boolean value that determines whether a control is read only	Text boxes, password boxes
size	Sets or returns how many characters wide a field is	Text boxes, password boxes
src	Sets or returns the URL of an image	Image submit buttons
tabIndex	Sets or returns a control's position in the tab order	Check boxes, radio buttons, reset buttons, submit buttons, image submit buttons, text boxes, password boxes, file boxes, hidden text boxes
type	Returns the type of input element: button, check box, file, hidden, image, password, radio, reset, submit, or text	Check boxes, radio buttons, reset buttons, submit buttons, image submit buttons, text boxes, password boxes, file boxes, hidden text boxes
useMap	Sets or returns the name of an image map	Image submit buttons
value	Sets or returns the value of form controls	Check boxes, radio buttons, reset buttons, submit buttons, image submit buttons, text boxes, password boxes, file boxes, hidden text boxes

**Table 5-6** Input field object properties and their associated form controls

# Input Field Objects (continued)

Method	Description	Form controls
<code>blur()</code>	Removes focus from a form control	Check boxes, radio buttons, reset buttons, submit buttons, text boxes, text areas, password boxes, file boxes
<code>click()</code>	Activates a form control's click event	Check boxes, radio buttons, reset buttons, submit buttons
<code>focus()</code>	Changes focus to a form control	Check boxes, radio buttons, reset buttons, submit buttons, text boxes, password boxes, file boxes
<code>select()</code>	Selects the text in a form control	Text boxes, password boxes, file boxes

**Table 5-7** Input field object methods and their associated form controls


# Text Boxes

- **Text box**
  - An `<input>` element with a type of “text”
  - Accepts a single line of text
- `value` attribute
  - Specifies text to be used as the default value at the moment a form first loads
- Example: The Gosselin Gazette Web page
  - Add text `<input>` elements to the Subscription form to collect basic customer data

# Text Boxes (continued)

**Figure 5-4**

Form with several text  
<input> elements



The screenshot shows a Mozilla Firefox browser window with the title "Text Boxes - Mozilla Firefox". The browser's address bar displays "file:///C:/Course". The form contains the following elements:

- Name:** A single text box containing "Office of the Mayor".
- Address:** A single text box containing "City Hall".
- City, State, Zip:** Three separate text boxes. The first contains "New York", the second contains "NY", and the third contains "38116".

The status bar at the bottom of the browser window shows "Done".

# Text Boxes (continued)

- Most form validation with JavaScript takes place when you submit the form
- You can use JavaScript's built-in `isNaN()` function
  - Determines if value entered by the user is a number
- Example: The Gosselin Gazette Web page
  - Add function to `Subscription.html`

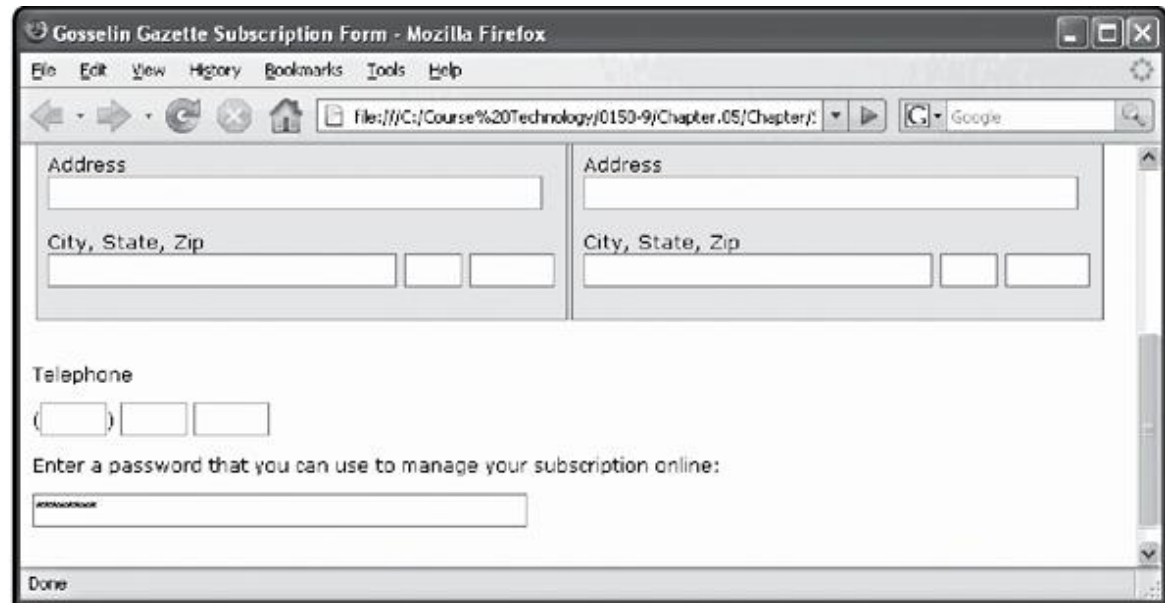
# Password Boxes

- **Password box**
  - An `<input>` element with a type of “password”
  - Entering passwords or other types of sensitive data
  - Character typed appears as an asterisk or bullet
- **Example: The Gosselin Gazette Web page**
  - Add a password `<input>` element to `Subscription.html`

# Password Boxes (continued)

**Figure 5-7**

Subscription form after  
adding a password  
<input> element



The screenshot shows a Mozilla Firefox browser window titled "Gosselin Gazette Subscription Form - Mozilla Firefox". The address bar displays a file path: "file:///C:/Course%20Technology/0150-9/Chapter.05/Chapter/". The search bar contains "Google". The form itself is divided into two columns. The left column contains an "Address" field, a "City, State, Zip" field (with separate boxes for state and zip), a "Telephone" field (with separate boxes for area code, number, and extension), and a "password" field. The right column contains an "Address" field and a "City, State, Zip" field. The status bar at the bottom of the browser window shows "Done".



# Push Buttons

- **Push button**

- An `<input>` element with a type of “button”
- Similar to OK and Cancel buttons in dialog boxes
- Primary purpose is to execute JavaScript code

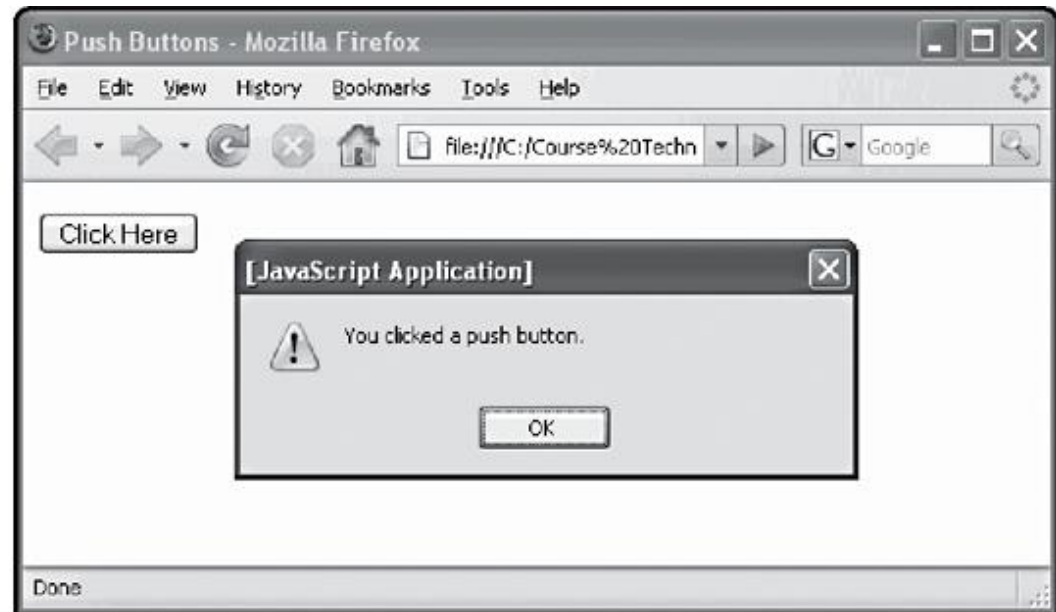
- **Example**

```
<p><input type="button" name="push_button"
  value="Click Here"
  onclick="window.alert('You clicked a push
  button.');" /></p>
```

# Push Buttons (continued)

**Figure 5-8**

A push button in a Web browser



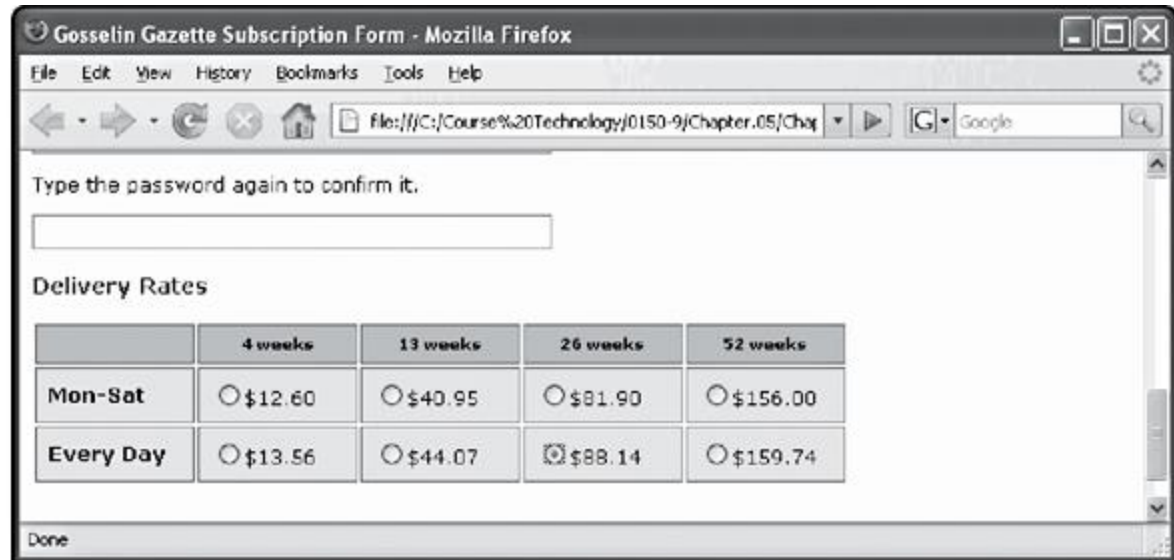
# Radio Buttons

- Group of **radio buttons**, or **option buttons**
  - An `<input>` element with a type of “radio”
  - User can select only one value
- All radio buttons in the group must have the same `name` attribute
- Each radio button requires a `value` attribute that identifies its unique value
- `checked` attribute in a radio `<input>` element
  - Sets an initial value for the group
- Example: The Gosselin Gazette Web page

# Radio Buttons (continued)

**Figure 5-10**

Subscription form after adding radio buttons



The screenshot shows a Mozilla Firefox browser window titled "Gosselin Gazette Subscription Form". The address bar displays a file path: "file:///C:/Course%20Technology/0150-9/Chapter.05/Chap...". The page content includes a password confirmation prompt, a "Delivery Rates" section, and a table of subscription options.

Type the password again to confirm it.

Delivery Rates

	4 weeks	13 weeks	26 weeks	52 weeks
Mon-Sat	<input type="radio"/> \$12.60	<input type="radio"/> \$40.95	<input type="radio"/> \$81.90	<input type="radio"/> \$156.00
Every Day	<input type="radio"/> \$13.56	<input type="radio"/> \$44.07	<input checked="" type="radio"/> \$88.14	<input type="radio"/> \$159.74

Done

# Radio Buttons (continued)

- When multiple form elements share same name
  - JavaScript creates an array out of the elements using the shared name
- Radio buttons share the same name
  - A single name=value pair can be submitted to a server-side script
- `checked` property returns a value of `true` if a check box or radio button is selected
- Example: The Gosselin Gazette Web page
  - Add more radio buttons to the subscription form

# Check Boxes

- **Check boxes**
  - An `<input>` element with a type of “checkbox”
  - Can be set to Yes (checked) or No (unchecked)
  - When you want users to select whether or not to include a certain item
    - Or to allow users to select multiple values from a list
- `checked` attribute
  - Sets the initial value of the check box to Yes
- Group check boxes by giving each check box the same name value

# Check Boxes (continued)

**Figure 5-11**

Form with check boxes



# Check Boxes (continued)

- Each check box can have a different value
- Users can select as many check boxes in a group as they like
- Example: The Gosselin Gazette Web page
  - Add check boxes to the Subscription.html document to allow users to select any other newspapers
  - Use a check box element in Billing Information and Shipping Information at the top of the form in Subscription.html



# Creating Selection Lists

- The `<select>` **element** creates a **selection list**
  - Presents users with fixed lists of options
- Options displayed in a selection list are created with `<option>` elements
- `<select>` element must appear within a block-level element such as the `<p>` element
- A selection list can also include a scroll bar

# Creating Selection Lists (continued)

Attribute	Description
disabled	Disables the selection list
multiple	Specifies whether a user can select more than one option from the list; a Boolean attribute
name	Designates a name for the selection list
size	Determines how many lines of the selection list appear

**Table 5-8** Attributes of the `<select>` element

# Menu Options

- **<option> element**
  - Specifies the options that appear in a selection list
- Each selection list must contain at least one `<option>` element
- Example: The Gosselin Gazette Web page
  - Add a selection list to Subscription.html
    - Subscriber uses to select any magazines to which they are currently subscribed

# Menu Options (continued)

Attribute	Description
disabled	Disables the option
label	Designates alternate text to display in the selection list for an individual option
selected	Determines if an option is initially selected in the selection list when the form first loads; a Boolean attribute
value	Specifies the value submitted to a Web server

**Table 5-9** Attributes of the <option> element

# The `Select` and `Option` Objects

- **Select object**
  - Represents a selection list in a form
  - Includes an `options[]` array containing an `Option` object for each `<option>` element in the selection list
- **Option object**
  - Represents an option in a selection list

# The Select and Option Objects (continued)

Property	Description
disabled	Sets or returns a Boolean value that determines whether a control is disabled
form	Returns a reference to the form that contains the control
length	Returns the number of elements in the <code>options[]</code> array
multiple	Sets or returns a Boolean value that determines whether multiple options can be selected in a selection list
name	Sets or returns the value assigned to the element's name attribute
options[]	Returns an array of the options in a selection list
selectedIndex	Returns a number representing the element number in the <code>options[]</code> array of the first option selected in a selection list; returns -1 if No option is selected
size	Sets or returns the number of options to display
tabIndex	Sets or returns a control's position in the tab order
type	Returns the type of selection list; returns "select-one" if the <code>&lt;select&gt;</code> element does not include the <code>multiple</code> attribute, or it returns "select-multiple" if the <code>&lt;select&gt;</code> element does includes the <code>multiple</code> attribute

**Table 5-10** Properties of the Select object

# The Select and Option Objects (continued)

Method	Description
<code>add(<i>element</i>, <i>before</i>)</code>	Adds a new option to a selection list
<code>blur()</code>	Removes focus from a form control
<code>focus()</code>	Changes focus to a form control
<code>remove(<i>index</i>)</code>	Removes an option from a selection list

**Table 5-11** Methods of the Select object

Property	Description
<code>defaultSelected</code>	Returns a Boolean value that determines whether the <code>&lt;option&gt;</code> element representing the currently selected item includes the <code>selected</code> attribute
<code>disabled</code>	Sets or returns a Boolean value that determines whether a control is disabled
<code>form</code>	Returns a reference to the form that contains the control
<code>index</code>	Returns a number representing the element number within the <code>options[]</code> array
<code>label</code>	Sets or returns alternate text to display for the option in the selection list
<code>selected</code>	Sets or returns a Boolean value that determines whether an option is selected
<code>text</code>	Sets or returns the text displayed for the option in the selection list
<code>value</code>	Sets or returns the text that is assigned to the <code>&lt;option&gt;</code> element's <code>value</code> attribute; this is the value that is submitted to the server

**Table 5-12** Properties of the Option object

# Adding Options to a Selection List

- ECMAScript recommendations suggest using the `add()` method of the `Select` object
  - To add new options to a selection list
  - Method is not consistently implemented
- Create a new option with `Option()` constructor
  - Then, assign the object to an empty element in an `options[]` array
- Example
  - Add a selection list to `Subscription.html`



# Removing Options from a Selection List

- Pass option's index number in `options[]` array to the `remove()` method of the `Select` object
  - Remaining elements are reordered
- Remove all the options from an options array
  - Set `length` of `options[]` array to zero
- Example
  - Add code to `Subscription.html` that deletes magazine names from the selection list

# Changing Options in a Selection List

- Assign new values to the option's `value` and `text` properties
- Example: The Gosselin Gazette Web Page
  - Add code to `Subscription.html` that modifies magazine names in the selection list

# Submitting and Resetting Forms

- Learn how to submit forms to a server-side script
  - And how to reset form fields to their default values
- Use JavaScript to:
  - Validate submitted data
  - Confirm whether users really want to reset form fields

# Submit Buttons

- **Submit button**
  - An `<input>` element with a type of “submit”
  - Transmits a form’s data to a Web server
- `action` attribute of the `<form>` element
  - Determines to what URL the form is submitted
- Submit buttons do not have values that are submitted to a Web server
- **Image submit button**
  - An `<input>` element with a type of “image”
  - Displays a graphical image and transmits a form’s data to a Web server

# Submit Buttons (continued)

- **Image submit button (continued)**
  - Include the `src` attribute to specify the image to display on the button
- Example: The Gosselin Gazette Web page
  - Add a submit button to `Subscription.html`

# Submit Buttons (continued)

**Figure 5-16**

Subscription form after adding an image submit button



# Reset Buttons

- **Reset button**
  - An `<input>` element with a type of “reset”
  - Clears all form entries and resets each form element to the initial value specified by its `value` attribute
- Text you assign to the reset button's `value` attribute appears as the button label
- Example: The Gosselin Gazette Web page
  - Add a reset button to `Subscription.html`

# Validating Submitted Data

- **onsubmit event handler**
  - Executes when a form is submitted to a server-side script
  - Often used to verify or validate a form's data before it is sent to a server
- **onreset event handler**
  - Executes when a reset button is selected on a form
  - Confirm that a user really wants to reset the contents of a form



# Validating Submitted Data (continued)

- Must return a value of true or false
  - Depending on whether the form should be submitted (true) or reset (false)
- Example: Gosselin Gazette Web page
  - Add `onsubmit` and `onreset` event handlers to `Subscription.html`

# Validating Text and Password Boxes

- Use an `if` statement in the `onsubmit` event handler
  - Check whether field's `value` property contains a value
- Example: The Gosselin Gazette Web page
  - Add code to the `confirmSubmit()` function in the Subscription form that validates the text and password boxes

# Validating Radio Buttons

- Use the `checked` property to determine which element in a group is selected
- Example: The Gosselin Gazette Web page
  - Add code to the `confirmSubmit()` function in the Subscription form that validates the Delivery Rates radio buttons

# Validating Check Boxes

- Use the `checked` property to determine whether an individual check box has been selected

# Validating Selection Lists

- Test whether the selection list's `selectedIndex` property contains a value of -1
  - If it does, then no option is selected
- Example: The Gosselin Gazette Web page
  - Add code to the Subscription form that selects all of the magazines when the form is submitted

# Summary

- Forms collect information from users and transmit that information to a server for processing
- `<form>` element designates a form in a Web page
- Elements to create form controls: `<input>`, `<button>`, `<select>`, and `<textarea>`
- Any form element into which a user can enter data is called a field
- The `Form` object represents a form on a Web page

# Summary (continued)

- Document object includes a `forms[]` array that contains all of the forms on a Web page
- Empty `<input>` element is used to generate input fields
- `<select>` element creates a selection list
- Use `<option>` elements to specify the options that appear in a selection list
- `Select` object represents a selection list in a form

# Summary (continued)

- `Option` object represents an option in a selection list
- Submit button transmits a form's data to a Web server
- A reset button clears all form entries and resets each form element to the initial value specified by its `value` attribute
- `onsubmit` event handler executes when a form is submitted to a server-side script
- `onreset` event handler executes when a reset button is selected on a form