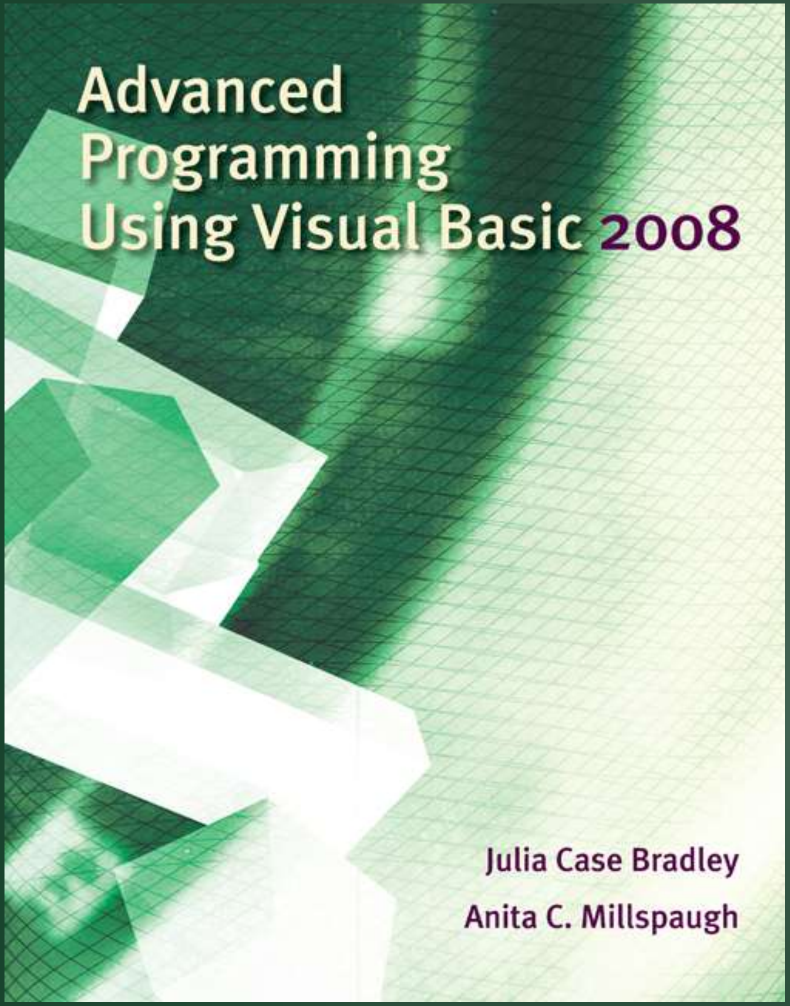

Advanced Programming Using Visual Basic 2008

Julia Case Bradley

Anita C. Millspaugh

Chapter 7

Web Applications

Objectives - 1

- Discuss the concepts of Web-based applications
- Understand the types of files that make up a Web project
- Distinguish among the various types of button controls
- Understand the event structure used by Web applications
- Include hyperlinks and link buttons on a page
- Navigate from one Web page to another

Objectives - 2

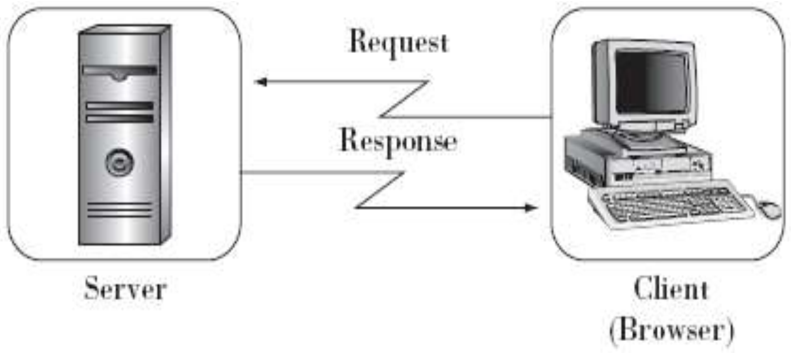
- Design a consistent layout using ASP.NET master pages
- Create and apply cascading style sheets
- Validate Web input using the Validation controls
- Maintain state (data values) from one page to the next
- Incorporate Login controls for both new and existing users
- Use AJAX controls and perform partial-page updates

Web Applications

- Developing an application for the Internet is different from creating a Windows program
- Windows Forms applications run on systems with Windows operating systems
- Web Forms are the gateway to cross-platform development
  - Displays in a browser application
  - Connect via an Internet Service Provider (ISP)

Client/Server Web Applications

- The Web *server* sends Web pages to the client where pages display inside a browser application



- The server can be on a remote machine or on the same machine as the client

Web Servers

- To publish Web applications, use either a remote *Web server* or make the local machine a Web server
  - – Visual Studio includes a built-in development Web server
  - – Install Internet Information Services (IIS) to make the development machine a Web server
  - – Use any available server that supports ASP.NET 3.5

Web Clients

- Browsers display pages written in hypertext markup language (HTML)
  - Pages may also contain programming logic in the form of script
    - JavaScript, VBScript, Jscript, or Java applets
- The browser renders the page and displays it on the local system
- Most common browser applications are Internet Explorer, Mozilla Firefox, Opera, Safari, and Netscape Navigator
- Test and check all output on multiple browsers

Browser Support

- ASP.NET applications run best in Internet Explorer
- ASP.NET is aware of the browser running the application
  - The HTML that it sends to the client is customized for the capabilities of the browser
    - If the browser can handle cascading style sheets, the font style is formatted using styles
      - Otherwise the font formatting is sent another way, such as with a Font tag

Web Pages

- HTML Web pages are stateless
  - A page does not store any information about its contents from one invocation to the next
- Maintain state by storing “cookies” on the local machine and sending state information as part of the URL
- Server may send a preformatted HTML file, or a program on the server may dynamically generate HTML to render a page

ASP.NET

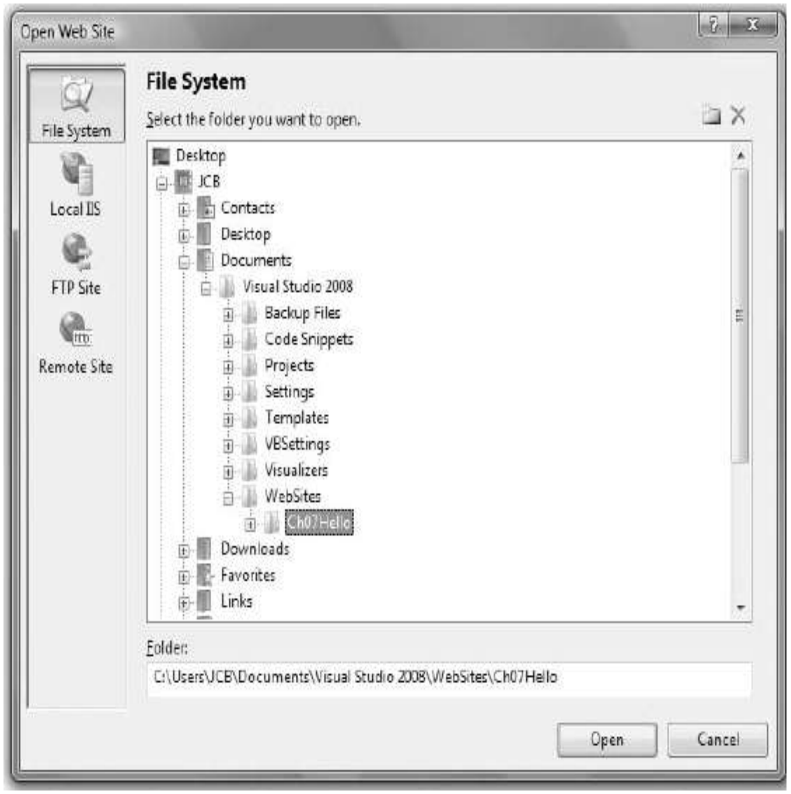
- The latest Web programming technology from Microsoft is ASP.NET 3.5
  - Provides libraries, controls, and programming support
  - Allows programs to be written that interact with the user, maintain state, render controls, display data, and generate appropriate HTML
- Web forms in Visual Basic or Visual Web Developer use ASP.NET 3.5
- Use VB and ASP.NET to create object-oriented event-driven Web applications with multiple classes that use inheritance

Visual Basic and ASP.NET

- Each Web form has two distinct parts
  1. HTML and instructions to render the page
     - Generates a file with .aspx extension
     - HTML is generated automatically by the Visual Studio IDE
  2. Visual Basic code
     - Generates a file with **.**aspx.vb extension
     - Program logic to respond to events — the “code-behind” file
     - Code is not compiled into an .exe file

Types of Web Sites

- Web applications are referred to as *Web sites* in Visual Studio
- VS provides four types of Web sites
  - File System
  - Local IIS
  - FTP Site
  - Remote Site



File System

Select the folder you want to open.

```
Desktop
  JCB
    Contacts
    Desktop
    Documents
      Visual Studio 2008
        Backup Files
        Code Snippets
        Projects
        Settings
        Templates
        VBSettings
        Visualizers
        WebSites
          Ch07Hello
    Downloads
    Favorites
    Links
```

Folder: C:\Users\JCB\Documents\Visual Studio 2008\WebSites\Ch07Hello

File System Web Sites

- Stores the Web pages and associated files in any folder on the local computer or other computer on the network
- Web pages are tested using the Visual Studio Web server
- Several advantages for Web developers over using IIS
  - Does not expose the computer to security vulnerabilities
  - Does not require administrative rights to create and debug a Web project
  - Runs on the Home Edition of Windows Vista or Windows XP

IIS Web Sites

- IIS is Microsoft's production Web server and is part of the operating system
- Includes a Web server, FTP server, e-mail server, and other services
  - Need to take extra steps to secure when running on a local computer
- Must have administrative rights to create IIS Web projects
  - If security on the network does not allow the proper permissions, IIS Web applications cannot be created

Remote Sites and FTP Sites

- Administrative rights must be granted on the host computer to utilize a remote Web server
- FTP sites cannot be used to create a new Web site
  - A previously created FTP Web site can be opened in Visual Studio

Creating Web Sites

- Select *File/New Web Site* to create a new Web application
  - Select *ASP.NET Web Site* for the template
  - Select *File System* for the location
  - Select *Visual Basic* for the language
  - The *Location* field is set to the Visual Studio 2008\WebSites folder

Web Page Files

- A new Web site automatically contains one Web page, Default.aspx
- Default.aspx.vb, the code-behind file, holds the VB code for the project
- The visual elements are created with HTML tags, not VB code
- ASP.NET provides two models for managing controls and code
  - *Code separation model* contains separate code and visual elements (described above)
  - *Single-file model* combines the visible elements and the code in a single file

Web Forms in the Visual Studio IDE

- When starting a Visual Basic Web application, a *Web Form* displays, also called a *Web page*
- Depending on the default setting, the Web Form may open showing the HTML source for the page
- To change the default behavior to always display the *Design* tab first, select *Tools/Options/Show all settings/HTML Designer/Start Pages in Design View*

Naming a Web Form

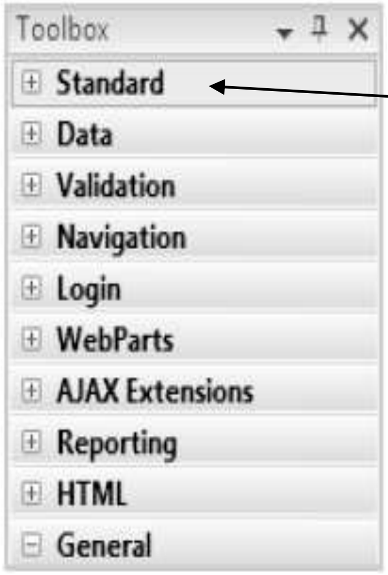
- In a new Web site the first page is called Default.aspx
- For most Web sites, the first page to display is called either *default* or *index*
- Additional pages and links need to have meaningful names

Opening an Existing Web Project

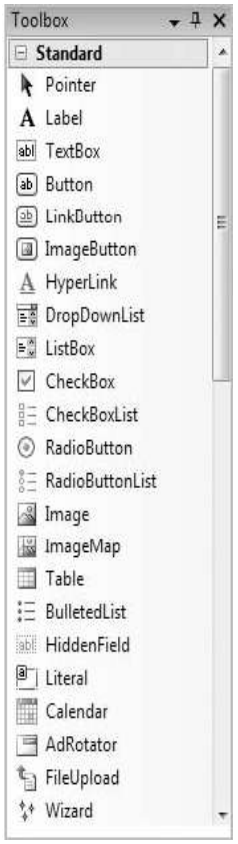
- Open the IDE and select *File/Open Web Site*
- Browse to the folder that holds the Web page files and click *Open*

Control Types - 1

- The toolbox has controls arranged into several groups

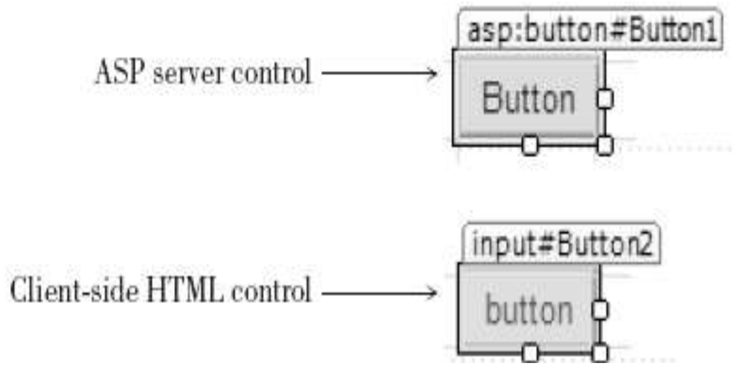


The Standard section of the toolbox holds the ASP.NET server controls which are used primarily



Control Types - 2

- In Design View the difference between client-side HTML controls and server-side controls can be seen in popup DataTips for each control
  - The type of control and its ID are indentified



Event Handling - 1

- VB code for the events of Web controls is written in the same way as Windows controls
  - Events occur on either the client or the server
  - Code is always executed on the server
  - Capturing an event, sending it to the server, and executing required methods is done automatically

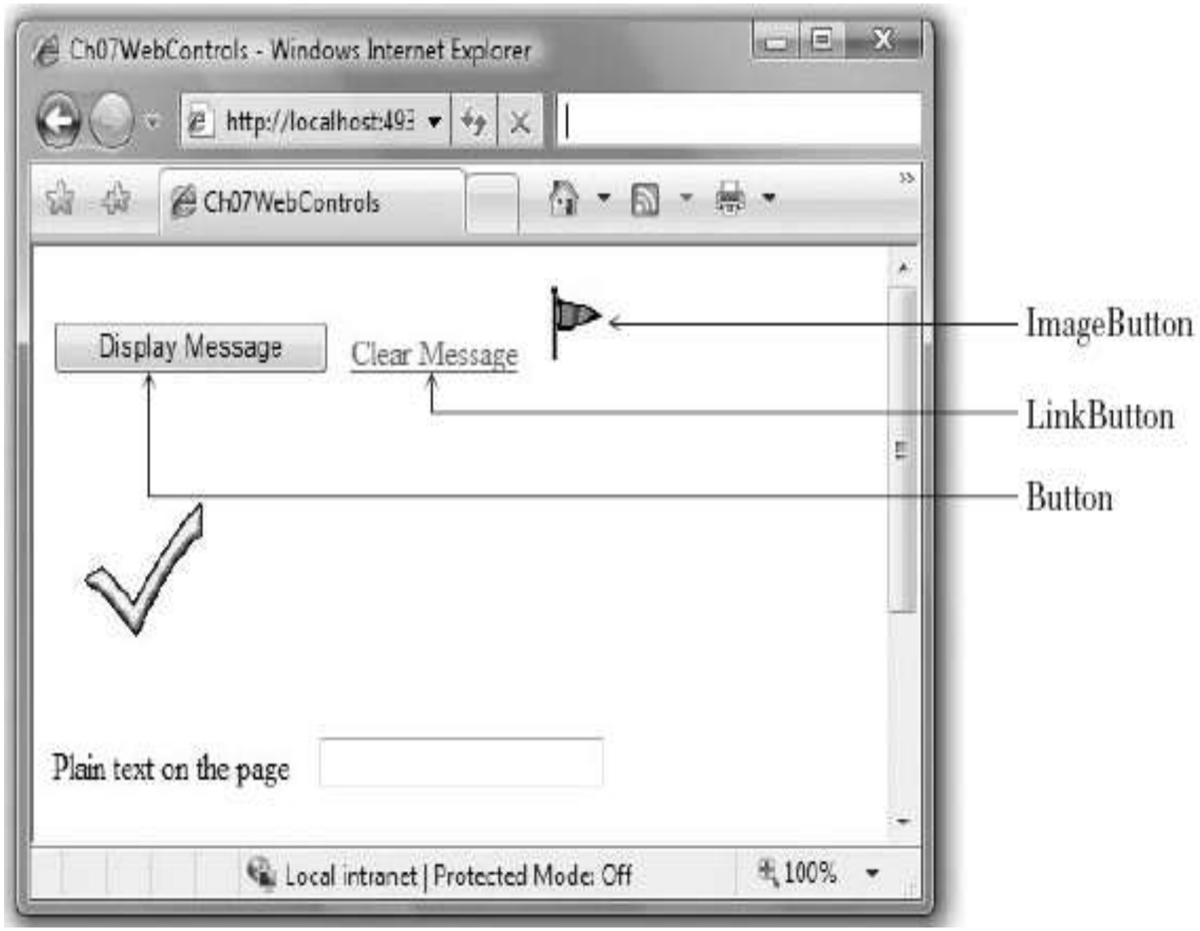
Event Handling - 2

- Events of Web Forms and controls are somewhat different from those of Windows Forms
- Some events may not occur and be handled as expected
  - A button click triggers a *postback,* which is a roundtrip to the server
  - Most events do not trigger a postback
  - When an event is posted to the server, all events since the last postback are processed

Button Controls - 1

- *Button, LinkButton, and ImageButton* are in the Standard toolbox
  - – Differ in appearance but not function
- LinkButton looks like a hyperlink, functions like a button and fires a Click event
- ImageButton can display a graphic image
- Code for the buttons is very similar to that in Windows Forms

Button Controls - 2



Debugging

- Running a Web application in the Visual Studio IDE is different from running a Windows application
- The IDE does not automatically generate the code necessary for debugging a Web application
- To run without debugging choose *Debug/Start Without Debugging* or press Ctrl + F5
- To run with debugging add the following line to the project’s *Web.config file*

```
<compilation debug=“true” />
```

The Hyperlink Control

- Looks like a LinkButton but is used to navigate to another Web page
- Does not have a Click event; it is intended for navigation only
- When a user clicks the hyperlink, browser navigates to the page indicated in the *NavigateUrl property*
  - Page can be any valid HTML page or another Web Form
- The navigation path (URL) value can be set at design time or in code

Choosing the Right Navigation Control

- A hyperlink button and a link button look the same on the page
- The hyperlink button has a NavigateUrl property which holds the URL of the page to which to transfer
- Use the link button to perform any action before navigating to another page
  - When the user clicks the link button, an event is fired and that page is submitted to the server

Linking to Another Page

- Use *Response.Redirect* or *Server.Transfer* to navigate to another Web page in code
  - Use *Server.Transfer* to transfer to another page on the same server
  - Uses one less round-trip to the server than *Response.Redirect*

```
' Tells the browser (client) to request a new page.
Response.Redirect("http://www.microsoft.com/")

' The server loads the new page and begins
' processing without a request from the browser.
Server.Transfer("LoginPage.aspx")
```

Including Images on Web Pages - 1

- Use the Image control to add graphics to a Web page
  - Similar to a PictureBox control on a Windows Form
- Each Image control has an ImageUrl property that specifies the graphic file's location
  - First copy the graphic into the Web site folder to make the project more self-contained and portable

Including Images on Web Pages - 2

- Add an Image control directly on a Web page
  - Place inside a DIV element or in a table cell
  - Click the Property button (…) in the ImageUrl property to open the *Select Image* dialog box
- Set properties of the Web page using the property settings for DOCUMENT
  - Drop down the *Object* list at the top of the Properties window and select *DOCUMENT*

The Calendar Control

- Allows the user to scroll to future dates and back to previous ones and select a date
- The SelectedDate property holds the date selected
- The SelectionChanged event fires when a new date is selected

Layout and Design of Web Forms

- Always be aware that users may have different browsers, screen sizes, and screen resolutions
- ASP.NET generates appropriate HTML to render the page in various browsers
  - ASP.NET cannot be aware of the screen size, resolution, window size, or user-selected font size
- Use styles to control layout of elements on a page (recommended practice)
  - A table can also be used
- Master pages and styles make it easier to update a Web site
  - Modifications are made in a single location and are automatically carried through to all affected pages

Current Standards for Page Layout

- DIV elements and styles
- Advantages over tables
  - Less HTML
  - Performance
  - Separation of structure from design
    - DIV elements define structure of page only
    - Design elements contained in CSS styles

Cascading Style Sheets

- New tools, called *Expression Web*, expand, enhance, and simplify using *cascading style sheets* (CSS)
- Use cascading style sheets to create, modify, and apply styles to
  - Single elements on a page
  - One entire page
  - All pages of an application
  - All applications of an organization

Using Styles - 1

- Rules that specify page layout, position, font, color, alignment, margins, borders, background, and bullet and number formats
  - Create and apply new styles within a page
  - Attach an external .css file and apply the styles
  - Save the styles in a page to an external .css file to use on other pages or Web sites

Using Styles - 2

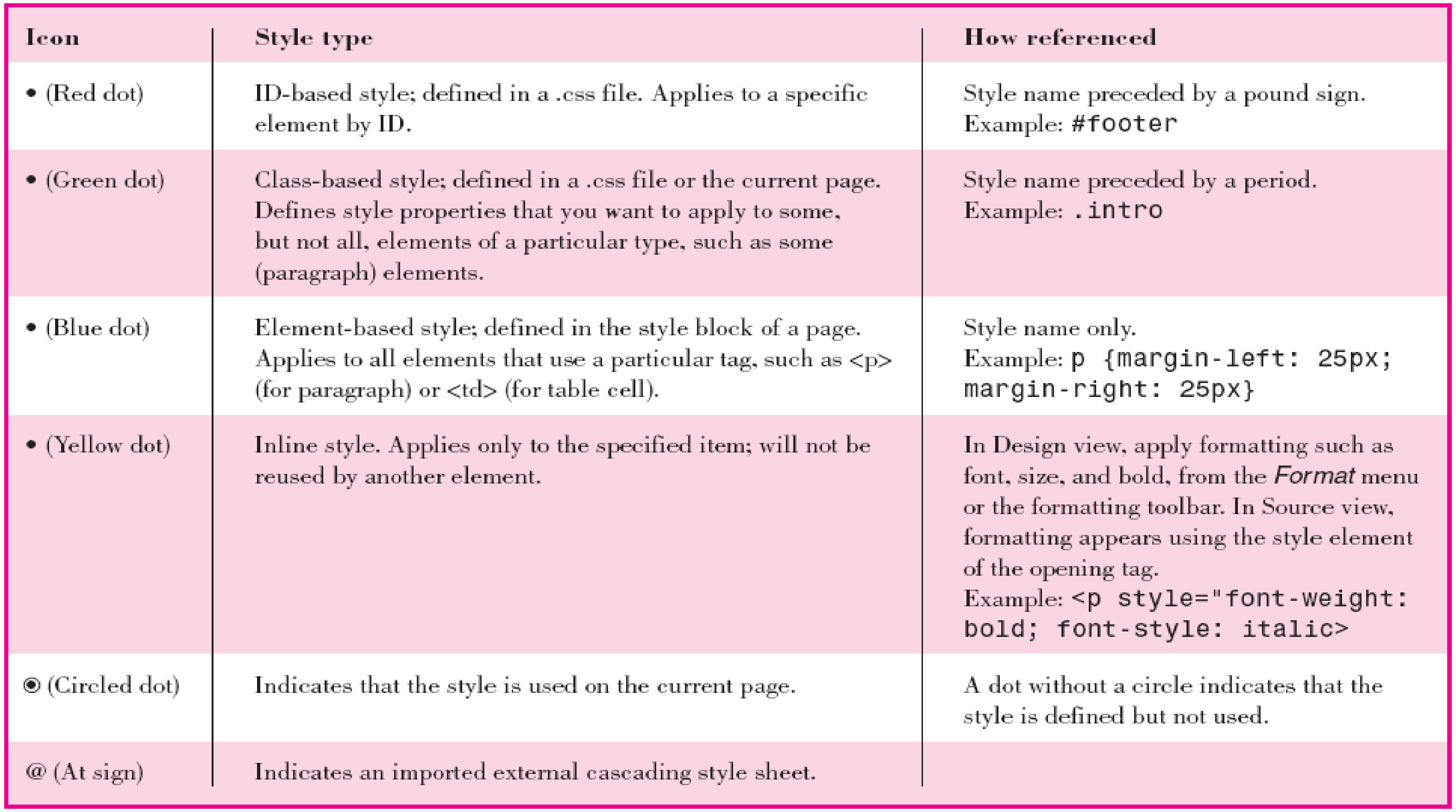
- *Inline styles* are used for elements that appear only once on a page
- *Page styles* are used for elements that may be used in more than one location on a page
- An external style sheet (.css file) is used for elements that may appear on more than one page of a Web site or in multiple Web sites

Using Styles - 3

- “Cascading” refers to the order of precedence of style rules
  - Locally created styles override rules of globally created styles
  - Example
    - Global style sheet sets font, color, size, and alignment
    - Local style for color and size take precedence, font and alignment of global style remain in effect
    - Inline style (more local) for size overrides size of local style, keeps color of local style and font and alignment of global style in effect

Types of Styles

- Cascading style sheet (CSS) style types

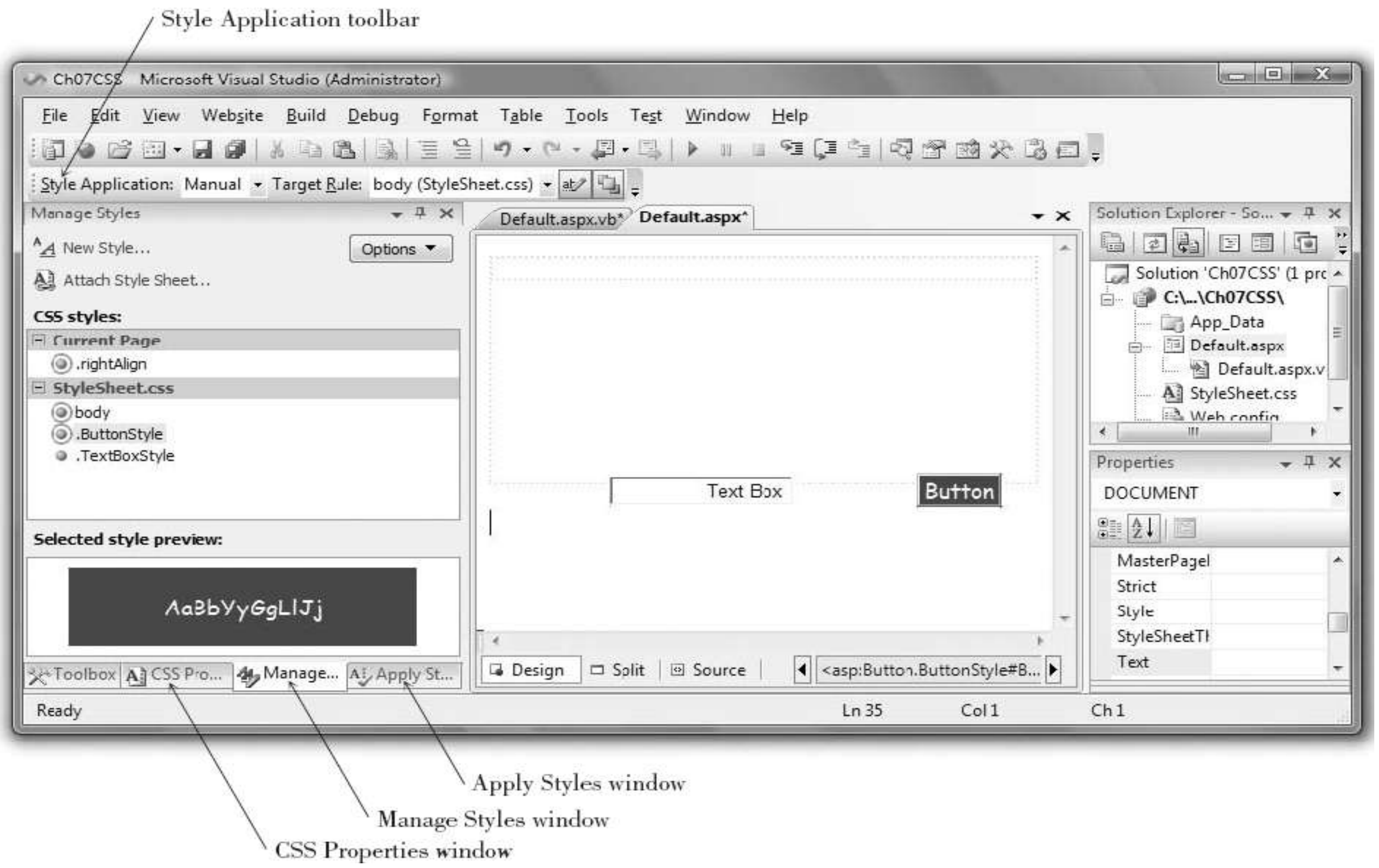


Icon	Style type	How referenced
• (Red dot)	ID-based style; defined in a .css file. Applies to a specific element by ID.	Style name preceded by a pound sign. Example: `#footer`
• (Green dot)	Class-based style; defined in a .css file or the current page. Defines style properties that you want to apply to some, but not all, elements of a particular type, such as some (paragraph) elements.	Style name preceded by a period. Example: `.intro`
• (Blue dot)	Element-based style; defined in the style block of a page. Applies to all elements that use a particular tag, such as <p> (for paragraph) or <td> (for table cell).	Style name only. Example: `p {margin-left: 25px; margin-right: 25px}`
• (Yellow dot)	Inline style. Applies only to the specified item; will not be reused by another element.	In Design view, apply formatting such as font, size, and bold, from the *Format* menu or the formatting toolbar. In Source view, formatting appears using the style element of the opening tag. Example: `<p style="font-weight: bold; font-style: italic>`
◉ (Circled dot)	Indicates that the style is used on the current page.	A dot without a circle indicates that the style is defined but not used.
@ (At sign)	Indicates an imported external cascading style sheet.	

New Style Tools

- CSS Properties, Manage Styles, and Apply Styles
  - Appear by default in the same area as the toolbox
  - Available from the *View* menu
  - Style Application toolbar appears in default layout of IDE

The Style Application Toolbar



Defining Styles

- Define a new style in the *New Style* dialog box
- Enter the name for the new style or choose the tag for an element type in the *Selector* box
- Choose a location for the new style in the *Define in* box

Managing Styles

- Use the Manage Styles window to preview each style
- Drag styles from one category to another
  - Changes the location of the style definition

Applying Styles

- Apply styles from several locations
  - Apply Styles window
    - Select the element on the page and click the desired style
  - Manage Styles window
    - Select the element on the page
    - Right-click style name, select *Apply Style* from context menu
  - *New Style* dialog box
    - Create a new style, check box for *Apply new style to document selection*

Modifying Styles

- Change the attributes of a style from either the Apply Styles or Manage Styles window
  - Select style name, right-click, select *Modify Style* from context menu
  - Can also modify style attributes in the CSS Properties window

Using DIV Elements to Lay Out a Web Page

- DIV element is a division or section of a page
  - DIV elements are *structure* of the page
  - Styles are the *design* of the page

Design View, Source View, and Split View

- Split view allows
  - Elements to be added in Design view
  - Results to be seen in HTML
  - Elements to be edited and moved in Source view

Creating a Page Layout – Step-by-Step - 1

- Create a new Web site
- Add three more DIVs
- Name the DIVs
- Set up the header style
- Set up the LeftColumn style
- Set up the MainContent style
- Set up the Footer style

Creating a Page Layout – Step-by-Step - 2

- Edit the StyleSheet file
- Examine the page
- Add the header graphic
- Add links to the left column
- Set up the main content and footer areas
- Run the application

Master Pages and Content Pages - 1

- *Master pages* provide the ability to define a standard layout and behavior for all pages in an application
- At run time, the individual *content pages* merge with the master page to produce the final layout for each page
- Ideal location for company logo and navigation controls
  - ContentPlaceHolder reserves space for the content that changes from page to page
  - A Web site can have multiple master pages

Master Pages and Content Pages - 2

- Extra step is required when using Master pages
  - ASP.NET checks to see if a content page is associated with a Master page
  - The page’s URL is the address of the content page
- To use a master page, add a new Master Page item to the project
  - Page has an extension of *.master*
- Master pages can be nested

Creating Master Pages

- Select the project in the Solution Explorer
- Use the *Website* menu or right-click the project name and select *Add New Item*
  - No need to change default name
  - Always check the *Place code in separate file* check box

Creating a Master Page - Step-by-Step

- Create the Web site and master page
- Set up the master page DIV elements
- Import and modify the style sheet
- Set up the master page header and left side
- Set up the master page footer
- Create the default content page
- Add a second content page
- Run the project

Setting the Tab Order

- Different for Web Forms than for Windows Forms
- Manually change the TabIndex property of each control
- By default, each control that is capable of receiving the focus has its TabIndex property set to zero
- Tab key moves focus in the order controls were added to the page
- Set the TabIndex property of each control
- If multiple controls have the same TabIndex, tab moves in the order the controls were added to the page

Setting the Focus to a Control

- Set the focus to an ASP Web control using the *Focus* method of the control or the *SetFocus* method of the page
- Set the initial focus to QuantityTextBox in the Page_Load event handler

```
QuantityTextBox.Focus()
```

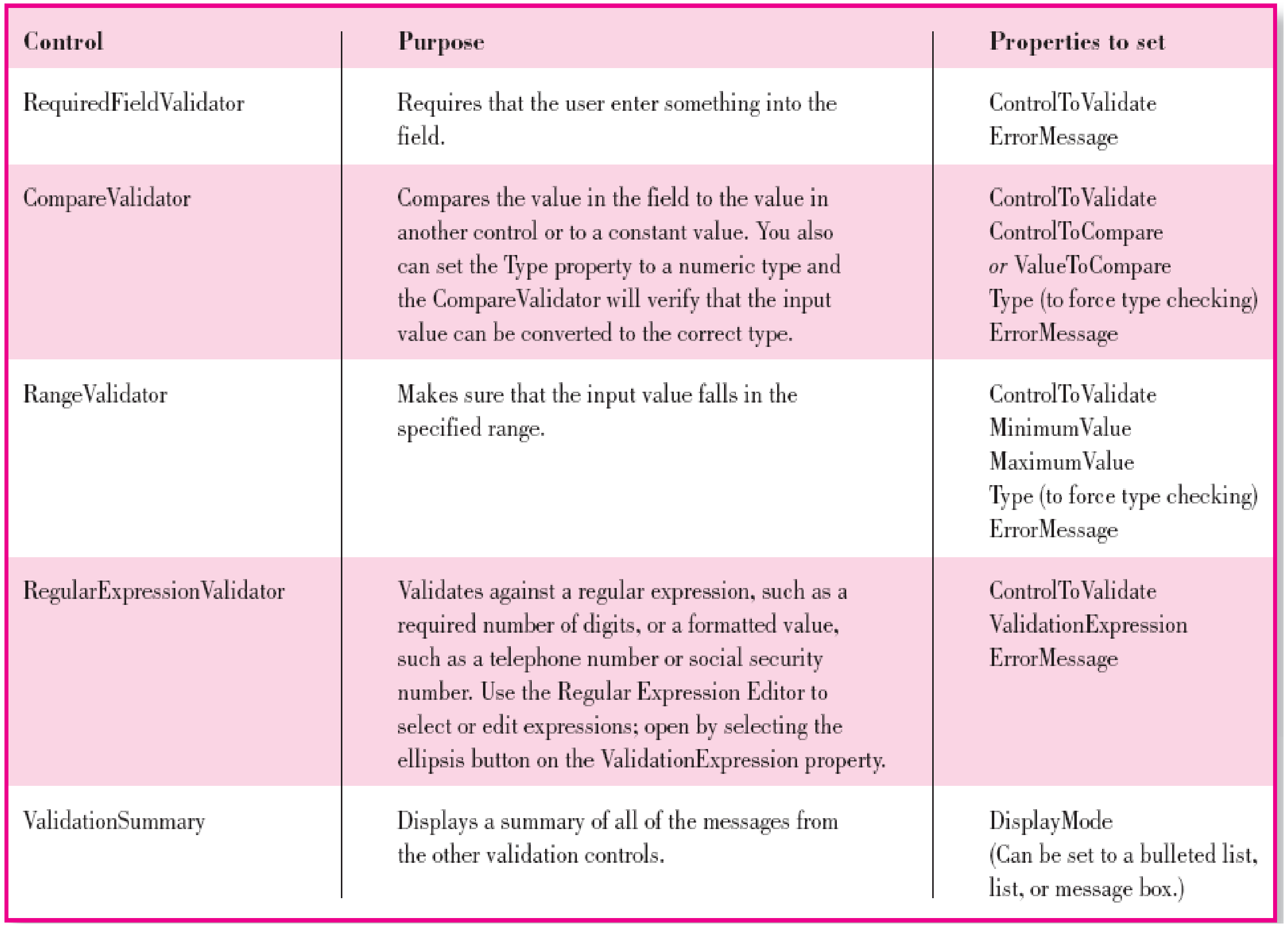
or

```
SetFocus(QuantityTextBox)
```

Using the Validation Controls

- Automatically validates input data
- Attach a *validation control* to an input control and set the error message
- Uplevel browsers (IE 5.5 or above) perform validation on the client without a postback to the server
- Downlevel browsers perform validation on the server when the page is submitted
- A blank entry passes validation for each control except the RequiredFieldValidator

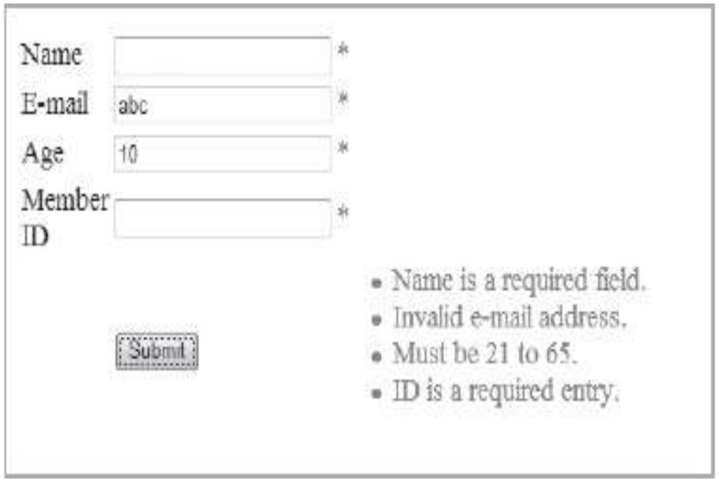
The ASP.NET Validation Controls



Control	Purpose	Properties to set
RequiredFieldValidator	Requires that the user enter something into the field.	ControlToValidate ErrorMessage
CompareValidator	Compares the value in the field to the value in another control or to a constant value. You also can set the Type property to a numeric type and the CompareValidator will verify that the input value can be converted to the correct type.	ControlToValidate ControlToCompare *or* ValueToCompare Type (to force type checking) ErrorMessage
RangeValidator	Makes sure that the input value falls in the specified range.	ControlToValidate MinimumValue MaximumValue Type (to force type checking) ErrorMessage
RegularExpressionValidator	Validates against a regular expression, such as a required number of digits, or a formatted value, such as a telephone number or social security number. Use the Regular Expression Editor to select or edit expressions; open by selecting the ellipsis button on the ValidationExpression property.	ControlToValidate ValidationExpression ErrorMessage
ValidationSummary	Displays a summary of all of the messages from the other validation controls.	DisplayMode (Can be set to a bulleted list, list, or message box.)

Displaying Asterisks

- Common technique used on many Web sites
- Display an asterisk next to the field and make the actual message appear in another location
- Set the validation control's ErrorMessage Text property to an asterisk



Name		*
E-mail	abc	*
Age	10	*
Member ID		*

- Name is a required field.
- Invalid e-mail address.
- Must be 21 to 65.
- ID is a required entry.

Testing for Validity

- Each of the validation controls has an IsValid property that returns *true* if the control assigned to the validator passes
- The Page object has an IsValid property that is set to *true* when all controls on the page pass their validation

```
If RequiredFieldValidator1.IsValid Then
        ' Perform some action.
End If

If Page.IsValid Then
        NavigateHyperlink.Enabled = True
End If
```

The Web Application Objects

- Web applications have access to the server objects:
  - Request, Response, Session, Application, and Server
  - Used without creating an instance
- Request passes from client to server
  - *Request object* holds information about current user, data entered, and arguments
    - Used to retrieve cookies
- Response goes from server to client
  - *Response object* used to create cookies

State Management - 1

- HTML pages are *stateless*
  - Do not retain values
  - Each time page is rendered, all controls are re-created
  - Any values entered by user, the *state*, are lost
- ASP.NET maintains the values in controls during a round-trip to the server or navigation to another page

State Management - 2

- Set *EnableViewState* property *true*
- The control names and values are compressed and encrypted into a single string and assigned to the Value property

```
<input type="hidden" name="__VIEWSTATE"
value="dDw5MTQ4NzEwMjE7Oz4tUQ/8e/xC31fa3oWMMe7CXP+ EAg==" />
```

- When the page is redisplayed, ViewState data are decrypted and used to fill the controls

Overview of State Management Techniques Server Side

- *The Session and Application objects*
  - Assign values that are maintained on the server and use them throughout an application
  - To use the Session object, user must accept cookies or modify the Web.config file to specify cookieless operation
- *Database fields*
  - Write data into database fields and read them back when appropriate

Overview of State Management Techniques Client Side

- *Cookies* — Create cookies in memory or on the user’s hard drive
- *Hidden fields* — Assign a hidden field a value that is passed and replaced in the field
- *A string appended to the URL* — Append state information to the URL
- *The Web Form’s ViewState property* — Declare key/value pairs and assign them to the ViewState of a form
- *Control State* — Allow property information for a control to be kept

Application and Session Objects - 1

- The *Application object* stores information as long as the application is running
  - Only one copy exists for all users
  - Stores information about the program, not about users
- One instance of the *Session object* exists for each user
  - Information is stored on the server

Application and Session Objects - 2

- Session values are maintained as long as the session exists
  - Logout option can call the *Session.Abandon* method
- Drawbacks to using the Session object
  - Storing large amounts of data could bog down the server
  - Web sites may split the server load among several systems, referred to as a *Web farm*
    - .NET specifies the name of the machine that stores the session values in the Web.config file

Cookieless Sessions

- You cannot use the Session object if the user refuses to accept cookies
- Declare a session to be cookieless in Web.config file
  - An encrypted session ID is appended to the page's URL every time the page posts to the server or the user navigates to another page

Cookies - 1

- Store state information as a cookie on the user's system
  - Store for just the session or on the hard drive for future trips to the Web site
  - Use the Expires property to make more permanent

Cookies - 2

```
' Temporary Cookie Stored in RAM.
'  Save the cookie in memory for this session.
Response.Cookies("UserName").Value = NameTextBox.Text
or
Response.Cookies.Add(New System.Web.HttpCookie("UserName".NameTextBox.Text))

' Permanent Cookie stored on the hard drive.
'  Store the cookie for 3 years.
With Response.Cookies("UserName")
            .Value = NameTextBox.Text
            .Expires = Today.AddYears(3)
End With
' Retrieve the cookie.
MessageLabel.Text = "Hello " & Request.Cookies("UserName").Value
```

The ViewState

- Use *ViewState* to save and restore the state of ASP.NET controls and other values for a single page
  - ViewState values are not retained when the user navigates to another page

ViewState of Controls

- Each ASP.NET control has an EnableViewState property set to *true* by default
  - For each postback of a page, a control is automatically saved and restored if its EnableViewState property is *true*

The ViewState of a Web Form

- Store text values in the ViewState of a form to maintain settings, values entered by a user, values of variables, or a dataset
- The ViewState information is passed to the server on each postback and returned with the form
  - Data values are not maintained on the server
  - Values are available only to the current form
- The ViewState property uses System.Web.UI.StateBag, a dictionary collection that holds names and values
- Information can be retrieved when reloading the same page from the server

Retaining the Values of Variables

- Local variables in a Web application work like local variables in a Windows application
- Module-level variables are lost when pages are reloaded
- Store the value of a module-level variable in a session variable, a ViewState variable, or a hidden control on the Web page
  - The control's EnableViewState property takes care of holding the value during postback

Saving Module-Level Variables in Session Variables

- Declare the session variable and assign its value in code
  - Convert any numeric variables to string before assigning the value
    - All session variables are string

```
Session("DiscountTotal") = DiscountTotalDecimal.ToString()
```

Checking for Postback

- Page_Load event occurs when an ASP.NET Web application loads
  - Occurs many times, each “round-trip” to the server (each postback)
- The page’s IsPostBack property is set to *false* for the initial page load and to *true* for all page loads following the first
- To make sure an action is performed only on postback and not the initial page load, check for *IsPostBack = true*

Login Features

- ASP.NET includes a group of controls for handling user login and passwords
  - Logins require setting up a new user id and password and changing passwords
  - A database maintains the login information

The Login Controls - 1

- Users can log in and out and recover their passwords
  - No code needs to be written to support these features
  - All validation is automatically included
- Login Control has many useful properties
  - DestinationPageUrl allows a page to be set to link to when a login is successful
  - CreateUserUrl allows a link for new users to be set
  - Error messages can be set to display when an invalid user name or password is entered

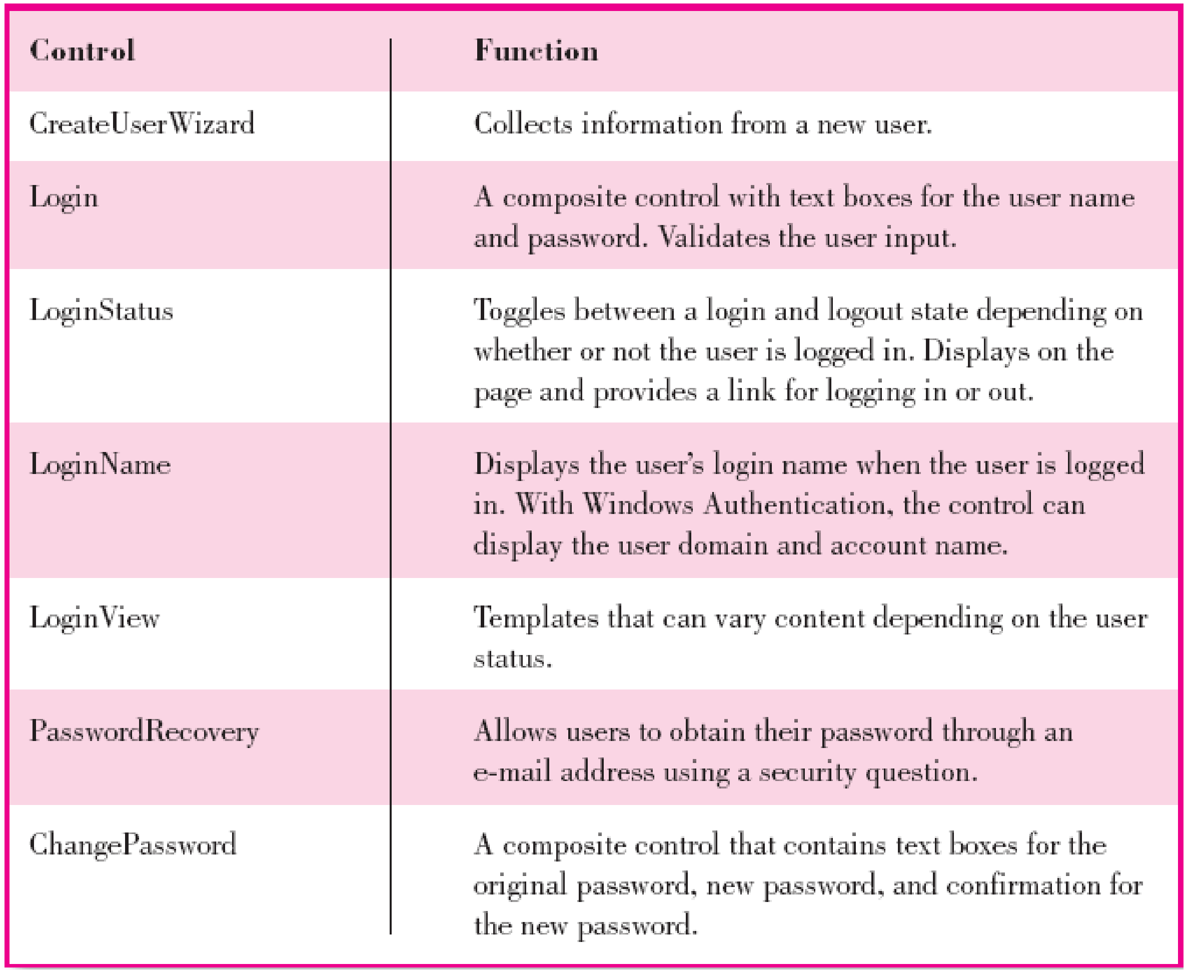
The Login Controls - 2

- CreateUserWizard control provides the ability to enter information for a new user and add it to the database
  - – Requires a strong password
  - – Allows a user to enter a name, password, password confirmation, e-mail address, and a security question
- ChangePassword and PasswordRecovery are two other useful controls

The Login Controls - 3

- LoginStatus control indicates whether user is logged in
  - Provides link to either log in or log out
  - Logout property specifies action to take when user logs out
    - Refresh current page, transfer to login page, or transfer to another page

The ASP.NET Login Controls



Control	Function
CreateUserWizard	Collects information from a new user.
Login	A composite control with text boxes for the user name and password. Validates the user input.
LoginStatus	Toggles between a login and logout state depending on whether or not the user is logged in. Displays on the page and provides a link for logging in or out.
LoginName	Displays the user's login name when the user is logged in. With Windows Authentication, the control can display the user domain and account name.
LoginView	Templates that can vary content depending on the user status.
PasswordRecovery	Allows users to obtain their password through an e-mail address using a security question.
ChangePassword	A composite control that contains text boxes for the original password, new password, and confirmation for the new password.

Adding Login Controls to an Application

- Drag and drop the controls on a form
- It is common to have password recovery and change password features available through a link
- To control access, it is best to have the login forms and public pages in the root directory of the Web site
  - Maintain the members-only pages in a separate folder

Using the Web Site Administration Tool

- Use the Administration Tool to set up user logins
  - Accessible from the *Website/ASP.NET Configuration* menu item
  - Create and manage users and roles
  - Establish access rules
  - *Roles* feature enables addition of groups (roles)
    - Use to assign access privileges for groups of users
  - *Access Rules* determine which users (or roles) have access to individual folders

Required Entries

- The Administration Tool provides many options
- To make the login controls work, two settings must be made
  - Authentication type
  - An access rule

Setting Up a Login Application - 1

- Create a Web site and add a Login.aspx page and a NewUser.aspx page
  - Add a Members folder and create a page
- Add links on the Default.aspx page for *Sign In* and *New User*
  - Link *Sign In* to Login.aspx and link *New User* to NewUser.aspx
- On Login.aspx add a Login control, a ChangePassword control, and a Password Recovery control
  - Set the DestinationPageUrl to the members page

Setting Up a Login Application - 2

- On NewUser.aspx add a CreateUserWizard control
- Open the Web Site Administration Tool
  - Select *From the Internet* as the authentication type
  - Select *Create access rules* and set the two options for your Members folder for *All Users* and *Allow*
- Run the program, add a user, try the login

AJAX

- Create interactive Web applications using *Asynchronous JavaScript* and *XML (AJAX)*
- Every AJAX application must have one (and only one) ScriptManager control
- AJAX controls are available on Microsoft's community Web site, CodePlex.com
- Many AJAX controls are called *extenders*
  - Add functionality to existing controls

AJAX Control Toolkit

- Download the AJAX toolkit
  - Add controls to the toolbox
  - Must be done before using AJAX controls

Download and Use AJAX Controls – Step-by-Step

- Download the AJAX control toolkit
- Create a new Web site
- Add the AJAX controls to the toolbox
- Add controls to a Web form
- Add and extend a button control

Use a SlideShow Extender Control – Step-by-Step

- Add and set up an image control
- Set up the images
- Connect the GetSlides function to the extender

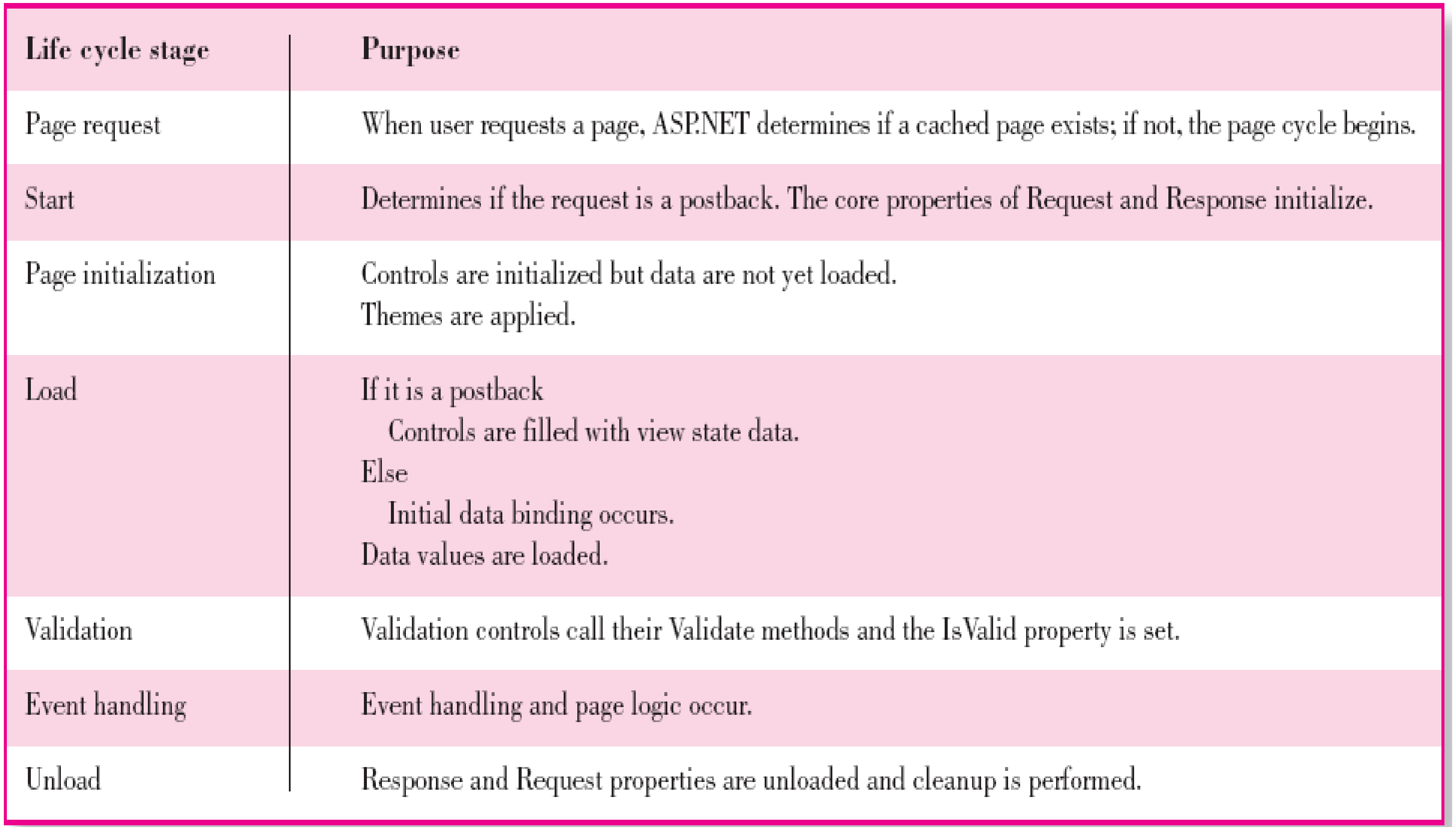
Partial Page Updates

- AJAX allows reloading of only a portion of a Web page, rather than an entire page, on each postback
  - Increases loading speed, only the portion of the page that has changed is downloaded and rendered
- The UpdatePanel is a container for other AJAX controls
  - Placing controls inside an UpdatePanel determines what portion of the page updates on postback

Other AJAX Notes

- AJAX Master Page and AJAX Web Form are templates with a script manager already included
- Text box extenders
  - CalendarExtender changes a text box into a drop-down calendar when the user clicks on the box
  - The type of characters allowed or not allowed in a text box can be set using the FilteredTextBoxExtender

ASP.NET Page Life Cycle



Life cycle stage	Purpose
Page request	When user requests a page, ASP.NET determines if a cached page exists; if not, the page cycle begins.
Start	Determines if the request is a postback. The core properties of Request and Response initialize.
Page initialization	Controls are initialized but data are not yet loaded. Themes are applied.
Load	If it is a postback Controls are filled with view state data. Else Initial data binding occurs. Data values are loaded.
Validation	Validation controls call their Validate methods and the IsValid property is set.
Event handling	Event handling and page logic occur.
Unload	Response and Request properties are unloaded and cleanup is performed.

Location of Files

- Visual Studio IDE saves solution files in the default folder that is selected in *Tools/Options/Projects and Solutions*
- Project files are stored in a separate folder from the Web site files
- Two folders with the same name are created if you store a Web site in the same location as solution files
  - Solution files are stored in the second folder that is created
- Choose to save the solution files inside the Web Site folder to make projects more portable and easier to open

Opening a Saved Web Site

- If .sln and .suo files are not saved in the project folder, then an existing Web site must be opened from inside the IDE
  - Select *File/Open Web Site* and browse to find the folder that holds the Web files (not the solution files)

Moving and Renaming a Web Project

- If the Web project is created as a File System project, it can easily be moved from one location to another
- The project folder can be renamed in Windows Explorer when the project is not open
- Unlike previous editions of Visual Studio, no hard-coded, complete paths are stored in the Web site files
- Moving and renaming an IIS Web site is more complicated
  - Use the Internet Information Services Manager to create a virtual directory

Copying and Publishing Web Sites

- Use the Copy Web tool to copy current Web files to another Web site
  - Copy files between any of the types of Web sites that can be used in Visual Studio
    - File System sites, IIS sites, remote Web sites, and FTP sites
- Visual Studio includes a Copy Web tool and a Publish Web utility
  - Copy Web tool copies all files and can perform synchronization of the files in both locations
  - The Publish Web utility is for publishing a completed Web to a production Web site