ICT365

Software Development Frameworks

Dr Afaq Shah

Murdoch UNIVERSITY

ASP.net

Murdoch
UNIVERSITY

Topics

ASP.NET:

a set of classes and tools for creating web applications.

Web Services

are internet based applications that use XML messages (**SOAP** messages) for communications.

ASP.net

Building Distributed Applications

ASP.NET Architecture

ASP.NET Page Composition

ASP.NET Page Lifecycle

Web Technologies

- HTTP / HTTPS
- Client-side:

HTML / XHTML (Extensible HyperText Markup Language)

JavaScript / VBScript (client-side scripting)

Applets / ActiveX controls

- Server-side:

PHP

Python

JSP (Java Server Pages)

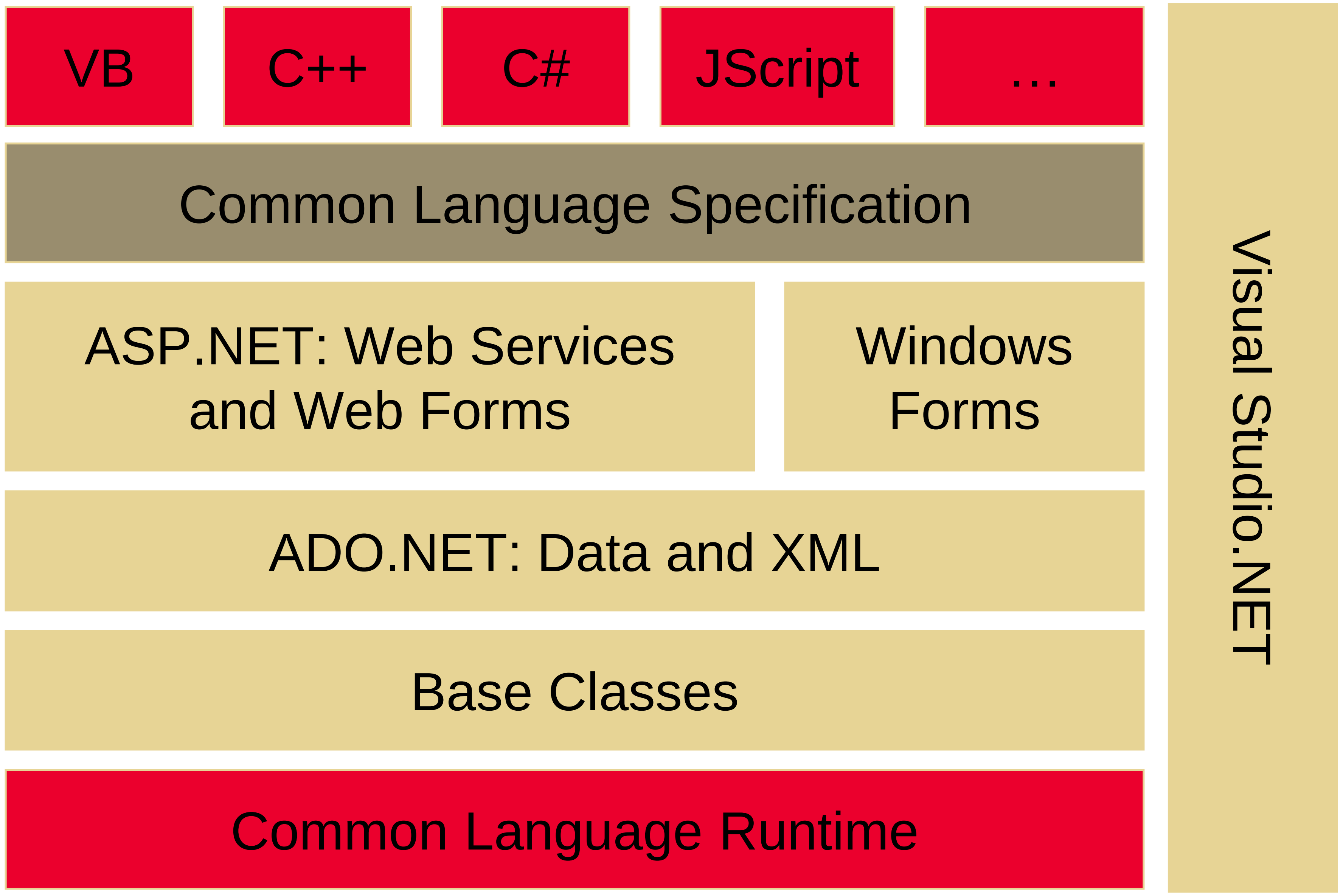
ASP (Active Server Pages)

ASP.NET (next generation of ASP)

ASP.NET Overview and Features

- ASP.NET provides services to allow the creation, deployment, and execution of Web Applications and Web Services
- Web Applications are built using Web Forms
- Web Forms are designed to make building web-based applications as easy as building Visual Basic applications
- Built on .NET Framework: any .NET programming language can be used (C#, Visual Basic)
- Complete object model
- Separation of code and UI
- Session management
- Caching, Debugging, Extensibility

ASP.NET Architecture



Programming Model

ASP.NET Object Model

- Controls are objects, available in server-side code

  Derived from `System.Web.UI.Control`

- The web page is an object too

  Derived from `System.Web.UI.Page`

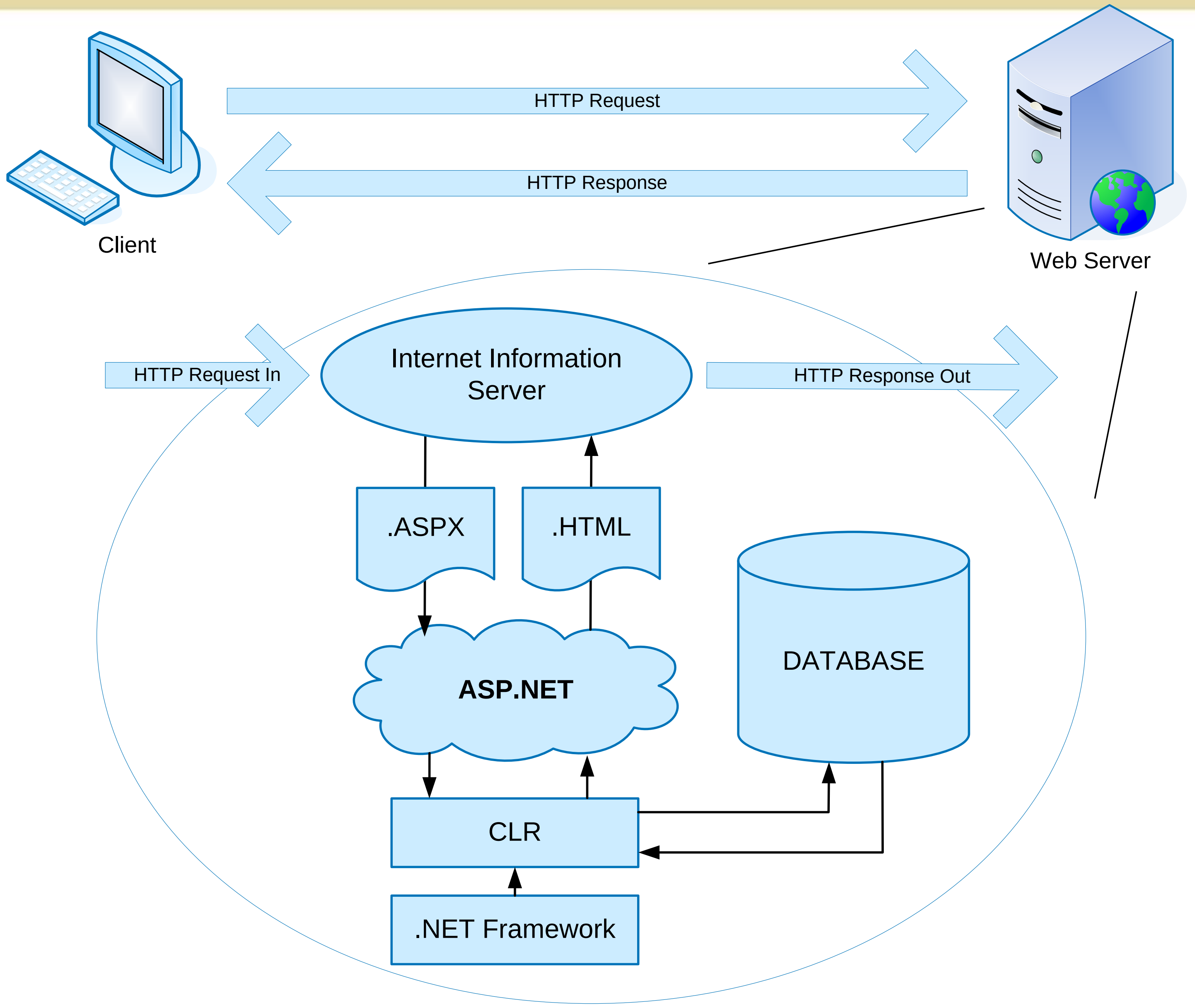
- User code executes on the web server in page or control event handlers

Distributed Applications

Divide Responsibility Accordingly

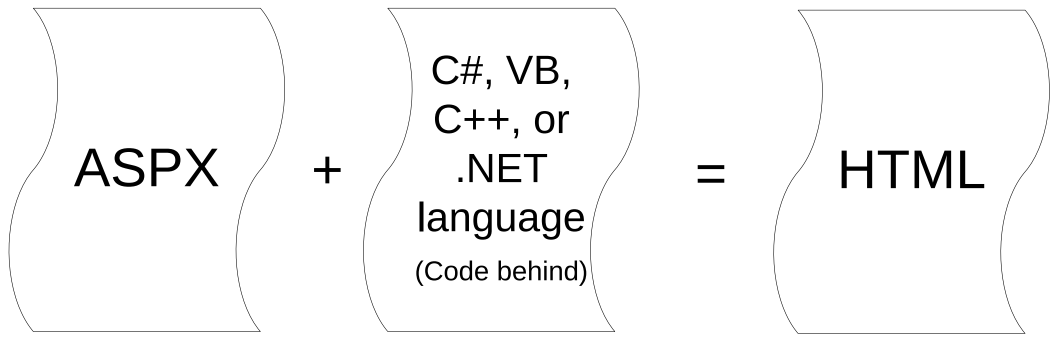
- CSS
- HTML
- ASP
- Code Behind
- Business Logic
- Themes
- MasterPage
- Web.config
- Database

ASP.NET Architecture

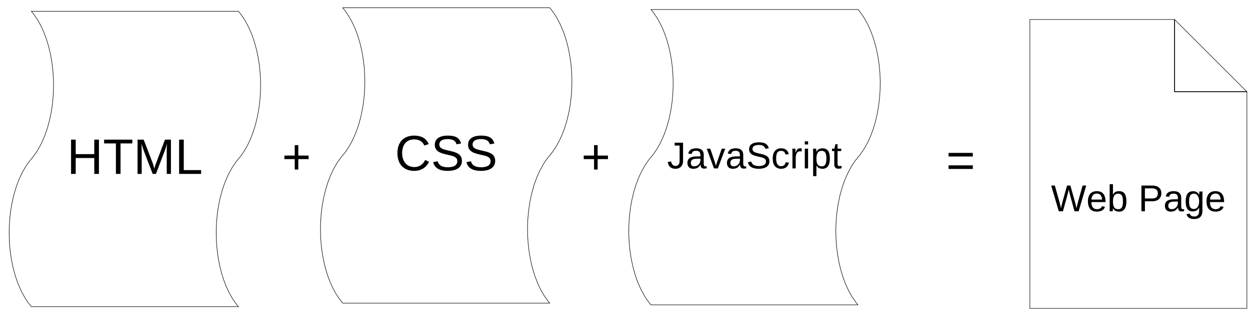


Page Composition Parts

Server Side:



Client Side:



ASP.Net Controls

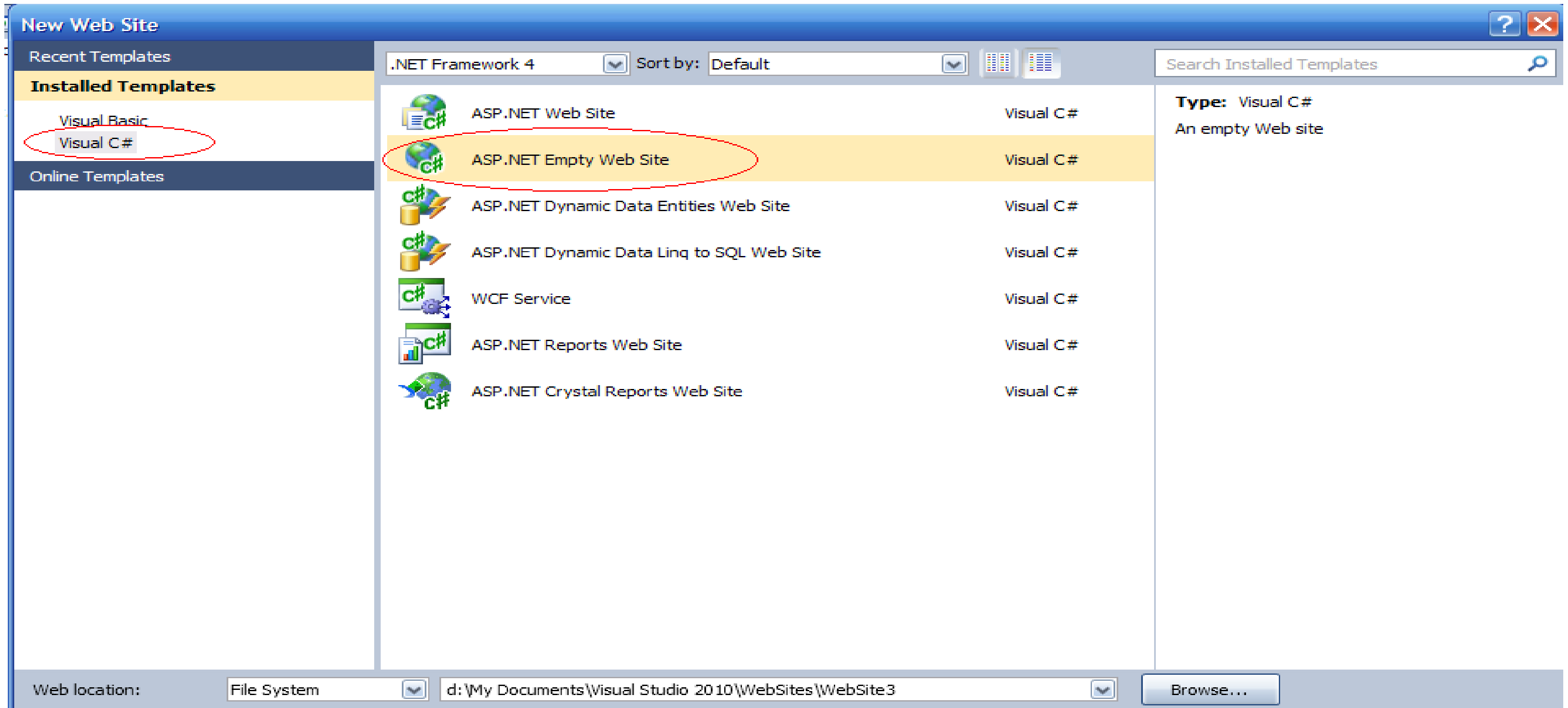
- Button Controls
- Text Boxes and Labels
- Check Boxes and Radio Buttons
- List Controls

WebTime.aspx Example
Creating an ASP.NET Web Application using Visual Studio

Murdoch UNIVERSITY

Step 1: Creating the Web Application Project

- Select **File > New Web Site...** and choose **ASP.NET Empty Web Site** in the **Templates** pane.
- Select **File System** from the drop-down list closest to **Location**.
-



ASP.NET Web Site	Visual C#
ASP.NET Empty Web Site	Visual C#
ASP.NET Dynamic Data Entities Web Site	Visual C#
ASP.NET Dynamic Data Linq to SQL Web Site	Visual C#
WCF Service	Visual C#
ASP.NET Reports Web Site	Visual C#
ASP.NET Crystal Reports Web Site	Visual C#

Type: Visual C#
An empty Web site

Web location: File System d:\My Documents\Visual Studio 2010\WebSites\WebSite3

WebTime.aspx Example

- Add an ASPX file (i.e., Web Form), default named Default.aspx is created for each new project.
- Visual Web Developer creates a code-behind file named Default.aspx.cs.
- The **View Designer** button opens the Web Form in **Design** mode.
- The **Copy Web Site** button allows you to copy the project's files to another location, such as a remote web server.
- Finally, the **ASP.NET Configuration** button takes you to the **Web Site Administration Tool.**
- Look at **Toolbox** displayed in the IDE when the project loads.

Standard and **Data** list of web controls.

Editing the WebTime.aspx

- When the project loads for the first time, the Web Forms Designer displays the autogenerated ASPX file in **Source** mode.
- **Design** mode indicates the XHTML element where the cursor is currently located.
- You can also view both the markup and the web-page design at the same time by using **Split** mode
- Right click the ASPX file in the **Solution Explorer** and select **View Code** to open the code-behind file.

WebTime.aspx Example

- Let’s create our first ASP.NET page using Visual Studio
  1. Modify title of the page
  2. Add a heading <h2>
  3. Look at the page in Design and Split modes
  4. Add a **Label** control from the *Toolbox*
  5. Change ID of the **Label** control
  6. Change some physical properties of the **Label** control
  7. Go to WebTime.aspx.cs file and add Page_Init function to set Text property of the **Label** control

WebTime.aspx Example

Changing the Title of the Page

- We change the page’s title from the default Untitled Page to “A Simple Web Form Example”.
- Open the ASPX file in **Source** mode and modify the text between the <title> tags.
- Alternatively, you can modify the Web Form’s **Title** property in the **Properties** window.
- To view the Web Form’s properties, select DOCUMENT from the drop-down list in the **Properties** window.

Designing the Page

- To add controls to the page, you can drag and drop them from the **Toolbox** onto the Web Form in **Design** mode.
- Like the Web Form itself, each control is an object that has properties, methods and events.
- You can type text directly on a Web Form at the cursor location or insert XHTML elements using menu commands.

Renaming the WebTime.aspx

Renaming the ASPX File

- Right click the ASPX file in the **Solution Explorer** and select **Rename**.
- Enter the new file name WebTime.aspx and press *Enter*. Both the ASPX file and the code-behind file are updated.

Renaming the Class in the Code-Behind File and Updating the ASPX File

- Visual Studio’s refactoring tool, which automatically updates the existing references to this class in the rest of the project to reflect this change.
- Right click the class name in the partial class’s declaration and select **Refactor > Rename…** to open the **Rename** dialog.

Visual Studio generates the markup shown when you create the GUI.

```
<%-- WebTime.aspx --%>
<%-- A page that displays the current time in a Label. --%>
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="WebTime.aspx.c
   Inherits="WebTime" EnableSessionState="False" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title>A Simple Web Form Example</title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <h2>Current time on the web server:</h2>
```

ASP.NET comments begin with <%-- and terminate with --%>, and can span multiple lines.

The Page directive specifies information needed by ASP.NET to process this file.

The document type declaration, which specifies the document element name and the PUBLIC URI for the DTD that defines the XHTML vocabulary.

XHTML documents have the root element html and markup information about the document in the head element.

The form that contains our XHTML text and controls is set to execute on the server, which generates equivalent XHTML.

The body contains the main content that the browser displays.

ASPX file that displays the web server’s time.

WebTime.aspx Example

Examining an ASPX File

- The Page directive’s **Language** attribute specifies the code-behind file’s language.
- The **CodeFile** attribute specifies the code-behind filename.
- When **AutoEventWireup** is true, ASP.NET automatically treats a method of name Page_*eventName* as an event handler.
- When AutoEventWireup is set to false, you specify event handlers using attributes in the Page directive just as you would any other web control.
- The **Inherits** attribute (line 4) specifies the class in the code-behind file from which this ASP.NET class inherits.

WebTime.aspx Example

- The document type declaration, which specifies the document element name and the PUBLIC URI for the DTD that defines the XHTML vocabulary.
- XHTML documents have the root element html and markup information about the document in the head element.
- Setting the **runat** attribute to **"server"** indicates that ASP.NET processes the element and its nested elements and generates the corresponding XHTML.
- The body contains the main content that the browser displays.
- The form that contains our XHTML text and controls is set to execute on the server, which generates equivalent XHTML.

Visual Studio generates the markup (Contd…)

```
        <p>
            <asp:Label ID="timeLabel" runat="server" BackColor="Black"
               Font-Size="XX-Large" ForeColor="Yellow"
               EnableViewState="False"></asp:Label>
        </p>
    </div>
    </form>
</body>
</html>
```

The **asp: tag prefix** indicates that the label is an ASP.NET web control, not an XHTML element.

Markup for a label web control.

- In an ASPX file a **directive** is delimited by **<%@** and **%>**.

ASPX file that displays the web server's time. (Part 2 of 2.)

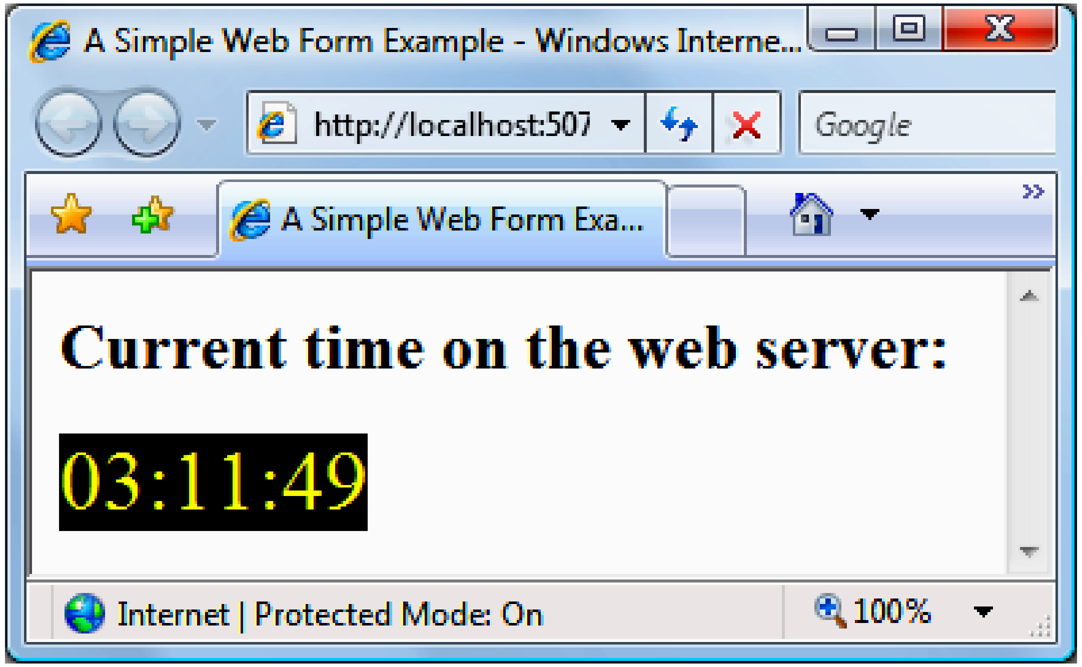
WebTime.aspx Example

- The **ID** attribute assigns a name to a control, used as an identifier in the code-behind file.
- The **asp: tag prefix** indicates that the label is an ASP.NET web control, not an XHTML element.
- Each web control maps to a corresponding XHTML element or group of elements.

WebTime.aspx Example

- The asp:Label control is written as an XHTML **span** element.
- A span element contains text with formatting styles.
- This control is processed on the server so that the server can translate the control into XHTML.
- If this is not supported, the asp:Label element is written as text to the client.

WebTime.aspx Example Run


A Simple Web Form Example - Windows Interne...
http://localhost:507
Google
A Simple Web Form Exa...
Internet | Protected Mode: On
100%

Current time on the web server:

03:11:49

- The **Page_Init** method handles the page’s **Init** event, which indicates that the page is ready to be initialized.

The code-behind file (WebTime.aspx.cs)

```
// WebTime.aspx.cs
// Code-behind file for a page that displays the current time.
using System;

public partial class WebTime : System.Web.UI.Page
{
   // initializes the contents of the page
   protected void Page_Init( object sender, EventArgs e )
   {
      // display the server's current time in timeLabel
      timeLabel.Text = DateTime.Now.ToString( "hh:mm:ss" );
   } // end method Page_Init
} // end class WebTime
```

The `Page_Init` method handles the page's `Init` event, which indicates that the page is ready to be initialized.

Retrieve the current time and formats it as `hh:mm:ss`.

Code-behind file for a page that displays the web server's time. (Part 1 of 2.)

WebTime.aspx Example *Relationship Between an ASPX File and a Code Behind File*

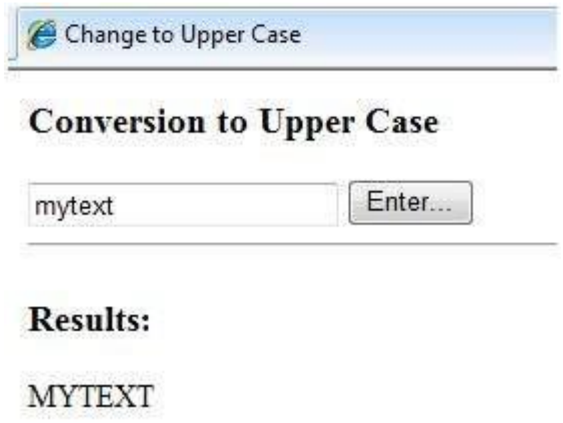
- The code-behind file inherits from Page, which defines the general functionality of a web page.
- The code-behind file contains a partial class.
- ASP.NET generates another partial class that defines the remainder of that class, based on the markup in the ASPX file.
- The first time the web page is requested, this class is compiled, and an instance is created.
- This instance represents our page—it creates the XHTML that is sent to the client.
- Once an instance of the web page has been created, multiple clients can use it to access the page—no recompilation is necessary.

Example: ASP.Net Page Layout

Murdoch UNIVERSITY

```
<!-- directives -->
<% @Page Language="C#" %>

<!-- code section -->
<script runat="server">

   private void convertoupper(object sender, EventArgs e)
   {
      string str = mytext.Value;
      changed_text.InnerHtml = str.ToUpper();
   }
</script>

<!-- Layout -->
<html>
   <head>
      <title> Change to Upper Case </title>
   </head>

   <body>
      <h3> Conversion to Upper Case </h3>

      <form runat="server">
         <input runat="server" id="mytext" type="text" />
         <input runat="server" id="button1" type="submit" value="Enter..."
OnServerClick="convertoupper"/>

         <hr />
         <h3> Results: </h3>
         <span runat="server" id="changed_text" />
      </form>

   </body>

</html>
```

Example


Change to Upper Case

Conversion to Upper Case

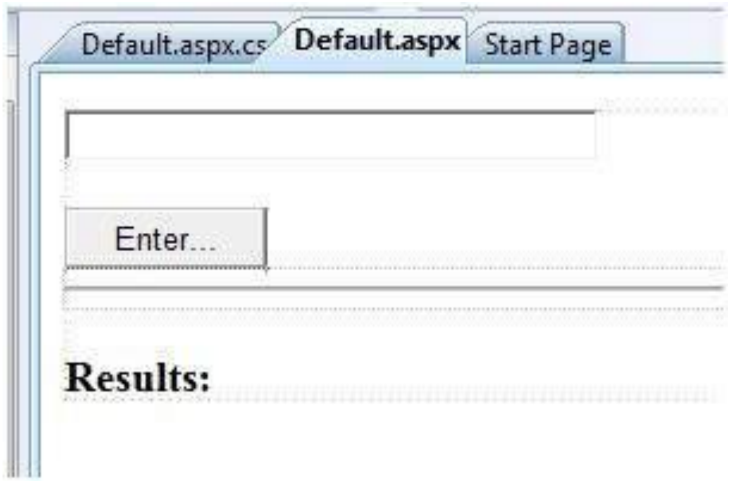
mytext

Enter...

Results:

MYTEXT

Example: Visual Studio IDE

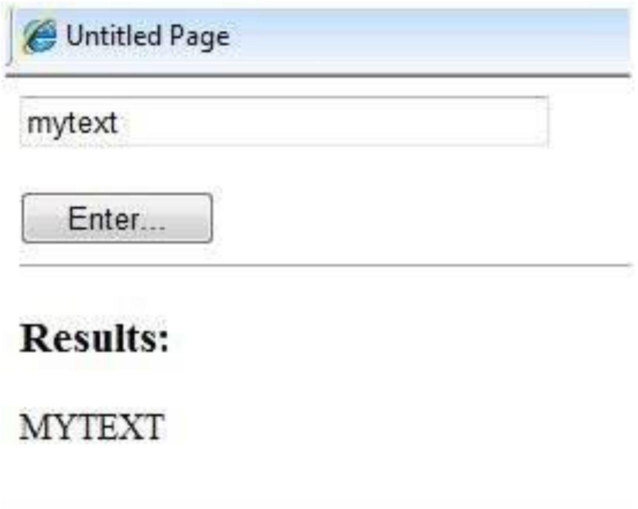


```
protected void Button1_Click(object sender, EventArgs e)
{
string buf = TextBox1.Text;
changed_text.InnerHtml = buf.ToUpper();
}
```

Example: Visual Studio IDE

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
  Inherits="firstexample._Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >

  <head runat="server">
    <title>
      Untitled Page
    </title>
  </head>

  <body>

    <form id="form1" runat="server">
      <div>

        <asp:TextBox ID="TextBox1" runat="server" style="width:224px">
        </asp:TextBox>

        <br />
        <br />

        <asp:Button ID="Button1" runat="server" Text="Enter..." style="width:85px" onclick="Button1_Click" />
        <hr />

        <h3> Results: </h3>
        <span runat="server" id="changed_text" />

      </div>
    </form>

  </body>

</html>
```

Example: Visual Studio IDE



mytext

Results:

MYTEXT

Event Handling

- GUIs are **event driven**.
- When the user interacts with a GUI component, the **event** drives the program to perform a task.
- A method that performs a task in response to an event is called an **event handler**.

Event Handling Example (`HelloWorld`)

- Create another ASP.NET page using Visual Studio

1. Add a `Button` and a `Label` control
2. To create this click event handler, double click the Button on the Form.
3. The following empty event handler is declared:
4. Set the `Text` property of the `Label` control with the current time in this function.

```
protected void Button1_Click(object sender,
                             EventArgs e)
{


}
```

Event Handling Example (`HelloWorld`)

- To add an event handler, alternatively in markup (aspx) file:

1. Add a `onclick="BClick"` property to the `Button` control.
2. Add a function `BClick` to the page class in the code behind.

```
<%-- Hello World page that also displays the current time. --%>

<%@ Page Language="C#" AutoEventWireup=
    CodeFile="HelloWorld.aspx.cs" Inherits="HelloWorldPage" %>


<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitiona
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional


<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

    <title>Hello World Web Form</title>

</head>

<body>

    <form id="form1" runat="server">

    <asp:Button ID="buttonClick" runat="server" Font-Size="Medium"
         Width="102px" Text="Click Me" onclick="BClick" />

    <br />

    <asp:Label ID="labelHello" runat="server"></asp:Label>

    </form>

</body>  </html>
```

ASP.NET comments begin with <%-- and terminate with --%>, and can span multiple lines.

The Page directive specifies information needed by ASP.NET to process this file.

XHTML documents have the root element `html` and markup information about the document in the `head` element.

The body contains the main content that the browser displays.

The `form` that contains our XHTML text and controls is set to execute on the server, which generates equivalent XHTML.

Markup for label & button web controls.

The `asp:` tag prefix indicates that the label is an ASP.NET web control, not an XHTML element.

ASPX Code Behind File

```
public partial class HelloWorldPage : System.Web.UI.Page
{
     protected void BClick(object sender, EventArgs e)
    {
        labelHello.Text = "Hello World! Time is " +
                          DateTime.Now;
    }
}
```

Event Handling

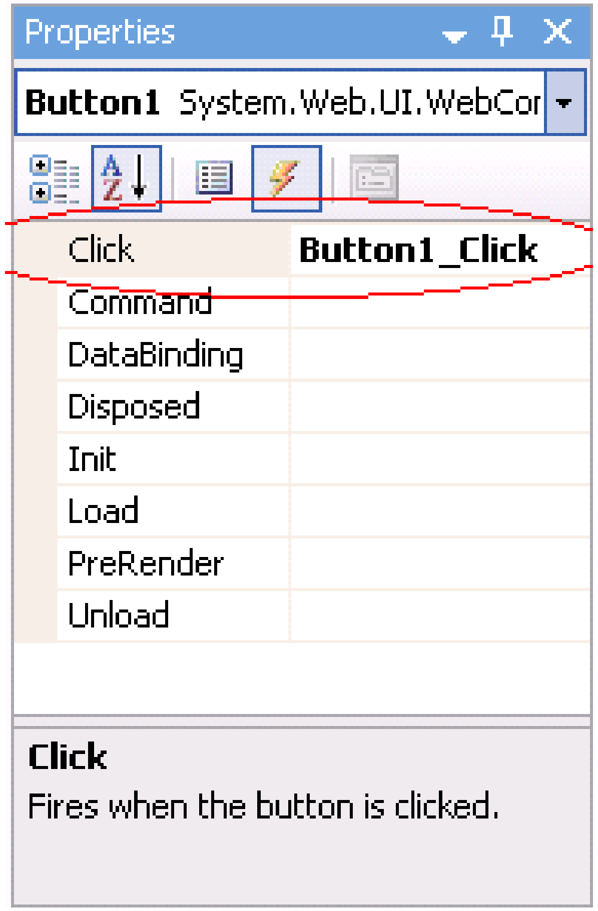
- By convention, C# names the event-handler method as *objectName_eventName* (e.g., `Button1_Click`).
- Each event handler receives two parameters when it is called:

An object reference named sender—a reference to the object that generated the event.

A reference to an object of type `EventArgs`, which contains additional information about the event.

Other Ways to Create Event Handlers

- Typically, controls can generate many different types of events.
- Clicking the **Events** icon (the lightning-bolt icon) in the **Properties** window, displays all the events for the selected control.

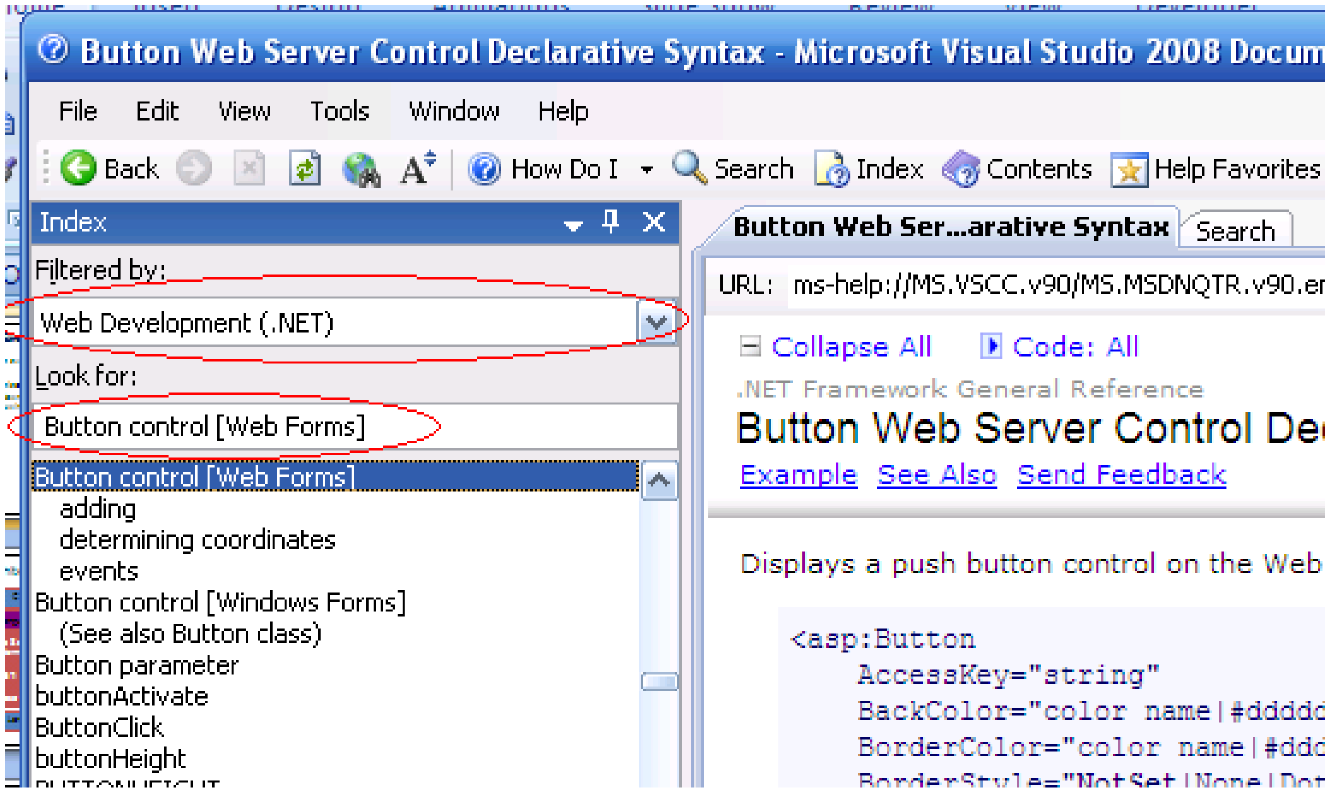


Click	Button1_Click
Command	
DataBinding	
Disposed	
Init	
Load	
PreRender	
Unload	

Click
Fires when the button is clicked.

Locating Event Information

- To learn about the events raised by a control, select **Help > Index**.
- In the window, select **Web Development (.NET)** in the **Filtered by** drop-down list and enter the name of the control's class in the **Index** window.


Button Web Server Control Declarative Syntax - Microsoft Visual Studio 2008 Docum

File Edit View Tools Window Help

Back How Do I Search Index Contents Help Favorites

Index

Filtered by:

Web Development (.NET)

Look for:

Button control [Web Forms]

Button control [Web Forms]
  adding
  determining coordinates
  events
Button control [Windows Forms]
  (See also Button class)
Button parameter
buttonActivate
ButtonClick
buttonHeight

Button Web Ser...arative Syntax | Search

URL: ms-help://MS.VSCC.v90/MS.MSDNQTR.v90.er

Collapse All Code: All

.NET Framework General Reference

Button Web Server Control De

Example See Also Send Feedback

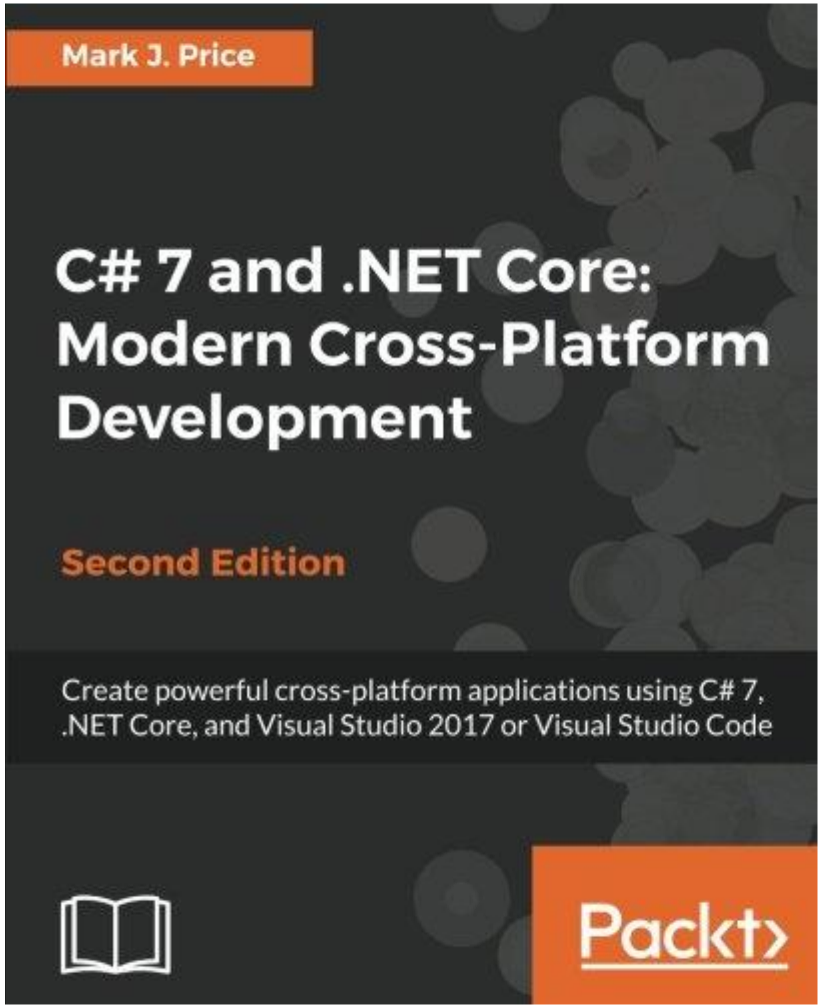
Displays a push button control on the Web

```
<asp:Button
    AccessKey="string"
    BackColor="color name|#dddd
    BorderColor="color name|#ddd
```

Reading/ reference

http://prospero.murdoch.edu.au/record=b2962782~S1

Chapter 14. Building Web Applications Using ASP.NET Core MVC


Mark J. Price
C# 7 and .NET Core: Modern Cross-Platform Development
Second Edition
Create powerful cross-platform applications using C# 7, .NET Core, and Visual Studio 2017 or Visual Studio Code
Packt>

Reading/ reference

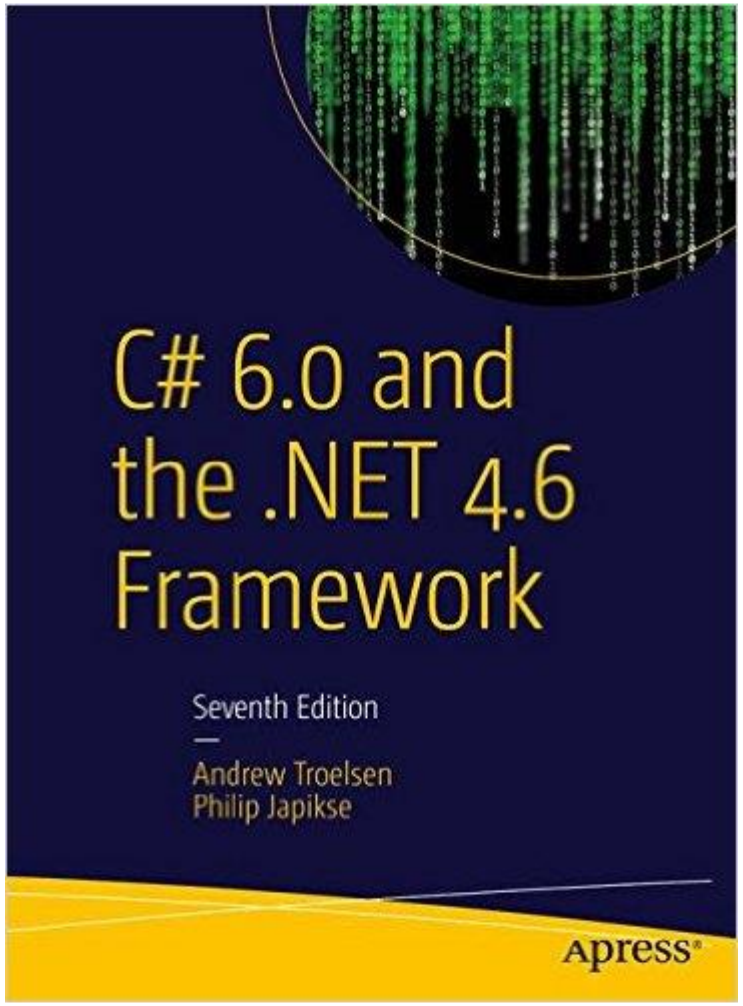
http://prospero.murdoch.edu.au/record=b2962780~S1

Chapter: Introducing ASP.NET Web Forms

Chapter: ASP.NET Web Controls, Master Pages, and Themes

Chapter: ASP.NET State Management Techniques

Chapter: ASP.NET MVC and Web API



Reading/ reference

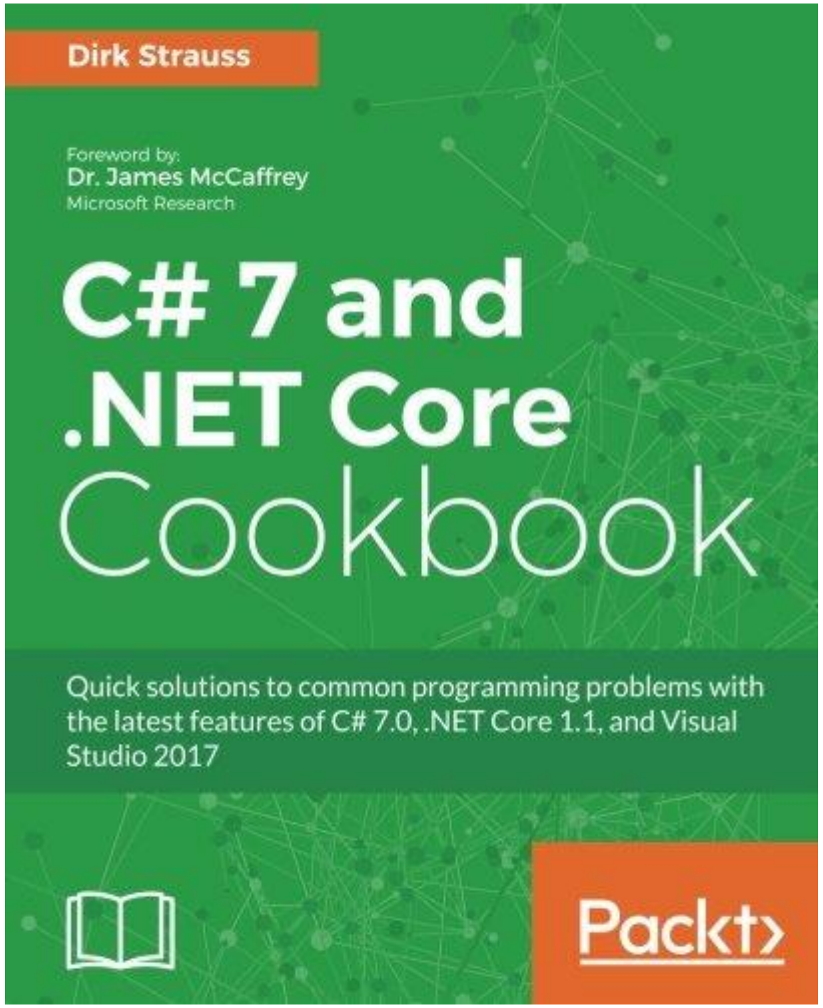
http://prospero.murdoch.edu.au/record=b2962781~S1

Chapter: MAKING APPS RESPONSIVE WITH ASYNCHRONOUS PROGRAMMING

Chapter: COMPOSING EVENT-BASED PROGRAMS USING REACTIVE EXTENSIONS

Chapter: EXPLORING .NET CORE 1.1

Chapter: ASP.NET CORE ON THE MVC FRAMEWORK


Dirk Strauss
Foreword by:
Dr. James McCaffrey
Microsoft Research
C# 7 and .NET Core Cookbook
Quick solutions to common programming problems with the latest features of C# 7.0, .NET Core 1.1, and Visual Studio 2017
Packt>

MSDN etc

http://msdn.microsoft.com/en-us/aa336522.aspx

ASP.NET

http://www.asp.net/

AspFree community

http://www.aspfree.com/

devx.com/dotnet/

http://www.devx.com/dotnet/