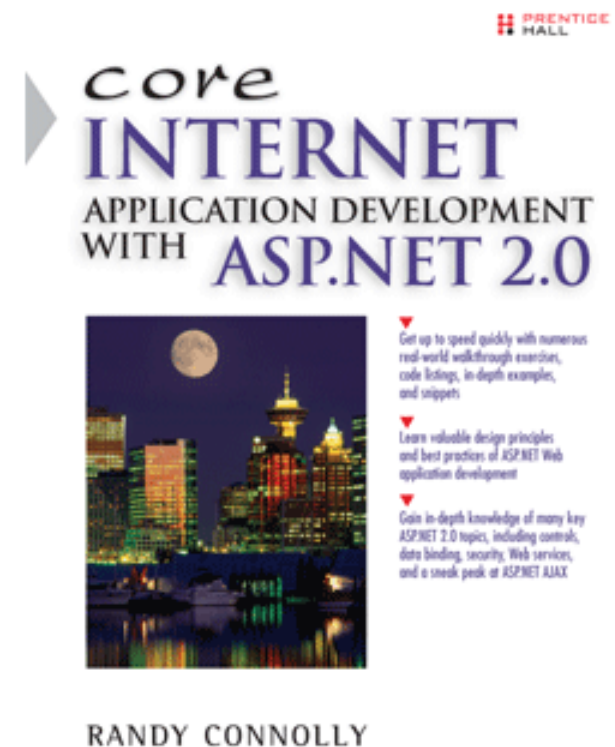


# Chapter 1

## Introducing ASP.NET 2.0

*The true beginning of our end.*

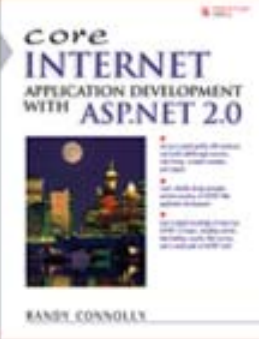
William Shakespeare,  
*A Midsummer Night's Dream.*



# Overview

- Introduces the nature and structure of ASP.NET.
- Illustrates how to use Microsoft Visual Studio 2005
- Tutorial walkthroughs for creating some sample ASP.NET Web forms using Visual Studio.
  - in-depth coverage of how ASP.NET works is left to the next chapter.

2 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

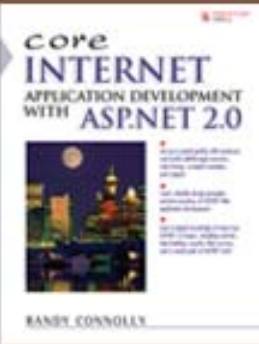
Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# What is ASP.NET?

- ASP.NET 2.0 is the current version of ASP.NET, Microsoft's powerful technology for creating dynamic Web content.
- ASP.NET is one of the key technologies of Microsoft's .NET Framework.
- It is the successor to Active Server Pages (ASP).

## 3 Introducing ASP.NET 2.0

COPYRIGHT © 2007 RANDY CONNOLLY



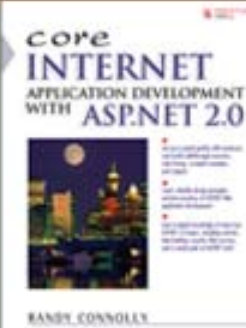
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Static vs Dynamic Web Pages

- Most Web pages that you view are not static HTML pages.
- Instead they are output from programs that run on servers.
  - These programs can interact with server resources like databases and XML Web services.

4 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY

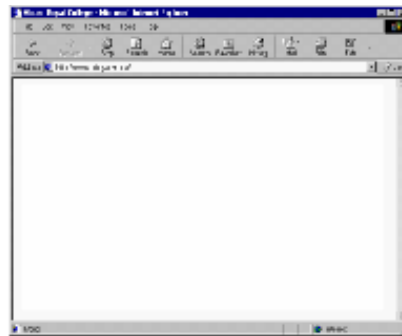


CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

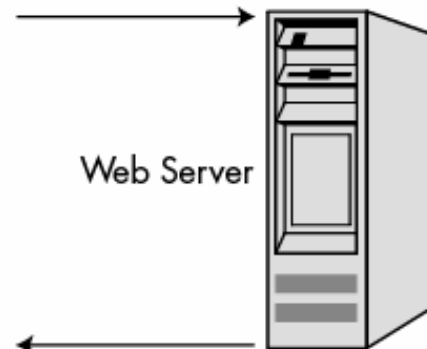
Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Static Web Content

5 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY



1. Browser requests index.htm from server

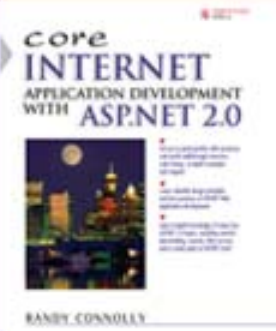


2. Server responds by sending content of index.htm to browser

3. Browser renders (displays) requested content

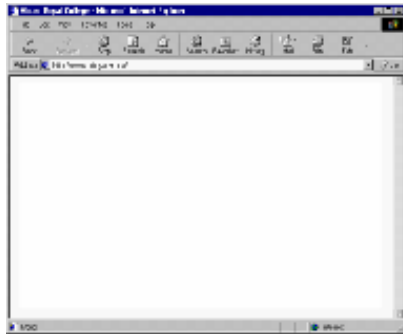
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

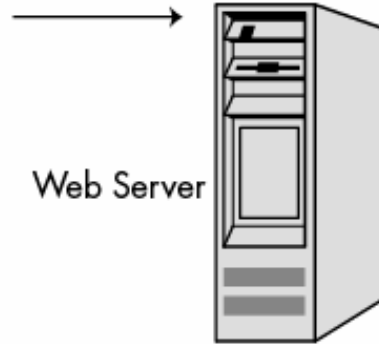


# Dynamic Web Content

6 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY

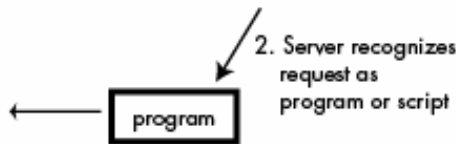


1. Browser requests program from server



Web Server

2. Server recognizes request as program or script



program

3. Program runs, gets information about the request from the server, interacts with server resources such as databases, and generates response (HTML and Javascript) that is sent back to browser



4. Browser renders content

CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

Core  
INTERNET  
APPLICATION DEVELOPMENT  
WITH ASP.NET 2.0



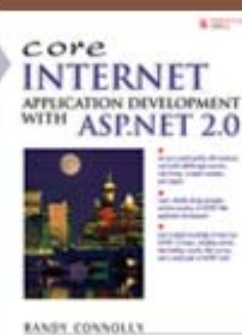
RANDY CONNOLLY

# Dynamic Web Technologies

- There are quite a number of different technologies for dynamically generating Web content.
  - ASP.NET
  - ASP
  - CGI
  - ColdFusion
  - JSP
  - PHP
  - Ruby on Rails
- All of these technologies share one thing in common:
  - Using programming logic, they generate HTML on the server and send it back to the requesting browser.

## 7 Introducing ASP.NET 2.0

COPYRIGHT © 2007 RANDY CONNOLLY



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

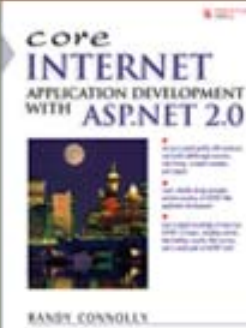
Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# ASP.NET Advantages

- ASP.NET provides a number of advantages compared to Microsoft's earlier, "classic" ASP technology.
  - Better performance
  - More powerful development environment
  - Easier maintenance
  - Smoother deployment and configuration

8 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY

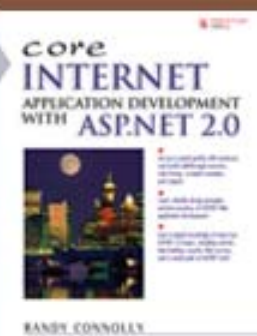


CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# The .NET Framework

- Many of the advantages that ASP.NET provides in comparison to other dynamic Web technologies are a result of its integration into Microsoft's *.NET Framework*.
- The .NET Framework is a **development framework** that provides a new programming interface to Windows services and APIs, and integrates a number of technologies that emerged from Microsoft during the late 1990s.
- The .NET Framework is also a **software platform** for the running and deployment of Windows-based software systems



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

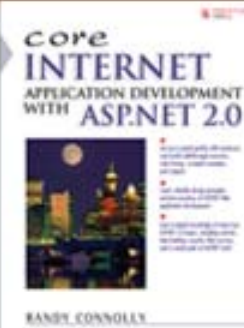
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Core Features of .NET

- Language interoperability
- Fully object-oriented languages
- Common runtime engine shared by all languages
- Base class library usable by all languages
- Simplified deployment
- Better security
- Better performance

10 Introducing ASP.NET 2.0

COPYRIGHT © 2007 RANDY CONNOLLY



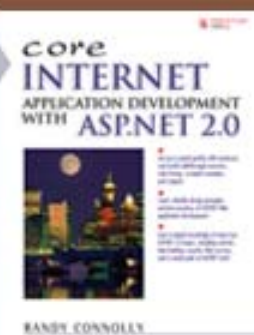
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# .NET Framework 3.0

- In June 2006, Microsoft combined a number of new Windows development technologies and branded them as the .NET Framework 3.0
- This .NET Framework 3.0 includes:
  - Windows CardSpace,
  - Windows Communication Foundation (formerly "Indigo"),
  - Windows Presentation Foundation (formerly "Avalon"),
  - Windows Workflow Foundation (formerly "WinFx")
  - .NET Framework 2.0.
- Thus .NET Framework 3.0 rather confusedly contains the current version of the .NET Framework, which is at present still version 2.0.



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

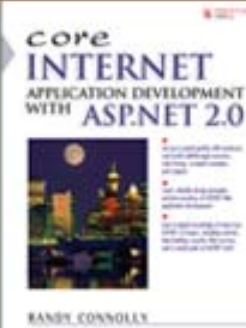
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# .NET Architecture

- The .NET Framework "sits" on top of the Windows operating system.
- Consists of the following components:
  - Language compilers
  - Common Language Runtime
  - .NET Framework Base Class Library

Source: Lam and Tai, *.NET Framework Essentials, 3rd Edition* (O'Reilly, 2003).

12 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY



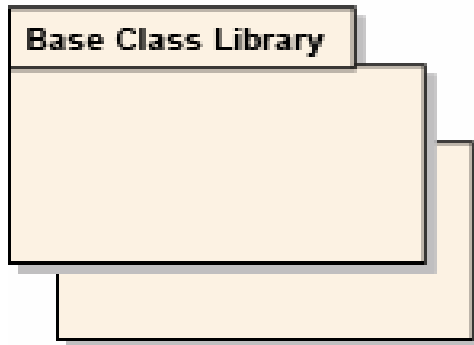
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# .NET Components

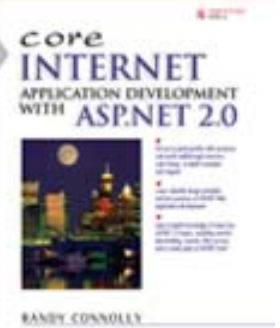
## 13 Introducing ASP.NET 2.0

COPYRIGHT © 2007 RANDY CONNOLLY



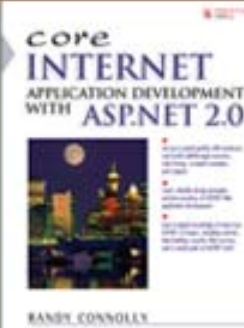
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)



# Language Compilers

- .NET languages can **interoperate**.
  - This means that you can use multiple .NET languages within a single .NET application.
  - The Microsoft provided languages are:
    - Visual Basic.NET (VB.NET), C#, JScript.NET, C++, J++
  - Other .NET languages are available from third-parties.
- Nonetheless, most .NET applications are written using a single language
  - Doing so generally makes an application easier to maintain and support in the long run.
- Language interoperability is, however, often utilized whenever a .NET application makes use of someone else's compiled class libraries.



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

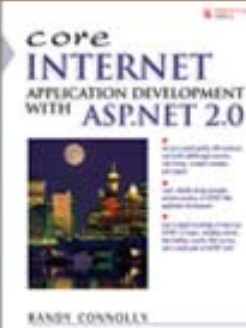
Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

## How is language interoperability achieved?

- All .NET languages must follow the rules in the **Common Language Specification (CLS)**.
  - These rules define a subset of common data types and programming constructs that all .NET languages must support.
- All .NET languages are compiled into a common format.
  - All code is compiled into **Microsoft Intermediate Language (MSIL)**, also called Common Intermediate Language (CIL or simply IL), rather than binary.

15 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY

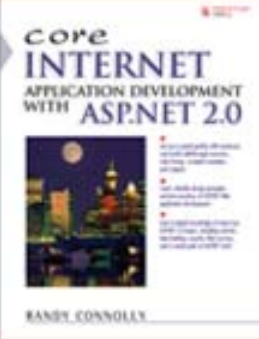


CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# MSIL

- MSIL is a CPU-independent virtual machine language analogous to Java's bytecode.
  - It consists of CPU-independent instructions for loading/storing information and calling methods.
- MSIL is not itself interpreted or executed.
  - The Common Language Runtime (CLR, covered shortly) converts the MSIL into managed native binary code at runtime using the Just-In-Time compiler as methods are called.

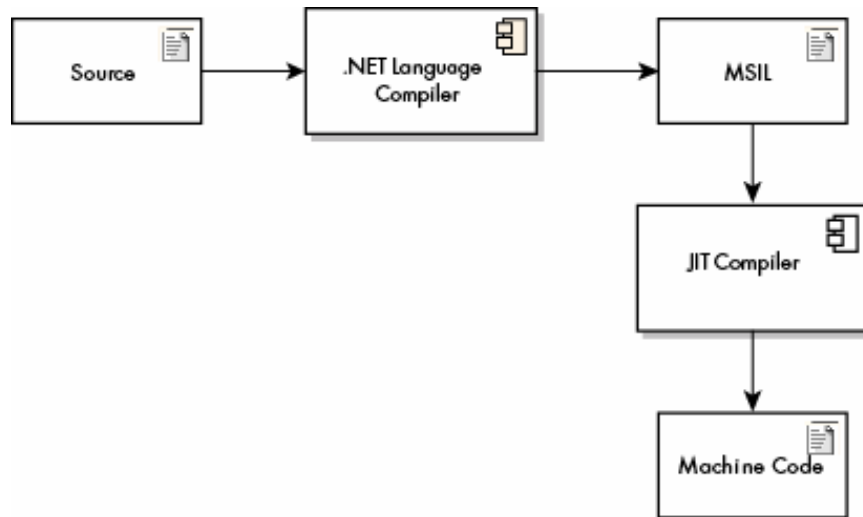


CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

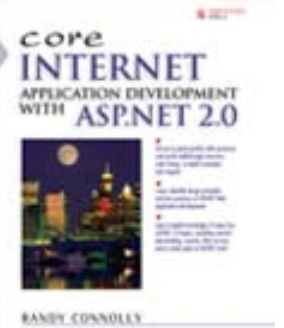
# .NET Compilation Process

17 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY



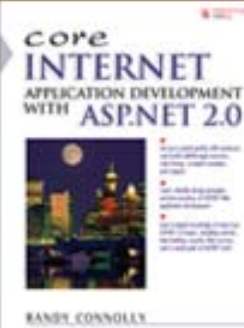
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)



# Assemblies

- MSIL is physically stored within special containers called **assemblies**.
- While these assemblies have familiar extensions (e.g., DLL or EXE), they are quite different from traditional Windows DLL or EXE files.

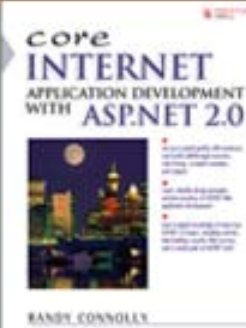


CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Assemblies

- A .NET assembly contains
  - MSIL instructions
  - metadata that includes
    - type definitions,
    - version information,
    - external assembly references, and other info.
- This metadata allows different components, tools, and runtimes to work together.
  - The CLR uses this metadata for verification, security enforcement, and other tasks.
- For this reason, .NET assemblies are said to contain **managed code**, in that the CLR manages the memory utilization and execution of the code.



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Sample MSIL Assembly

```
...
.assembly extern mscorlib
{
  .publickeytoken = (B7 7A 5C 56 19 34 E0 89 )
  .ver 1:0:5000:0
}
.assembly hello
{
  .hash algorithm 0x00008004
  .ver 0:0:0:0
}

.module hello.exe
// MVID: {F828835E-3705-4238-BCD7-637ACDD33B78}

.class private auto ansi beforefieldinit MainApp
  extends [mscorlib]System.Object
{
  .method public hidebysig static void Main( ) cil managed
  {
    .entrypoint
    .maxstack 1
    ldstr "C# hello world!"
    call void [mscorlib]System.Console::WriteLine(string)
    ret
  } // End of method MainApp::Main

  .method public hidebysig specialname rtspecialname
  instance void .ctor( ) cil managed
  {
    .maxstack 1
    ldarg.0
    call instance void [mscorlib]System.Object::.ctor( )
    ret
  } // End of method MainApp::.ctor
} // End of class MainApp
```

Metadata

MSIL Module

CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

Core  
INTERNET  
APPLICATION DEVELOPMENT  
WITH ASP.NET 2.0



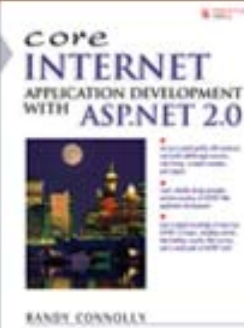
RANDY CONNOLLY

If we use the IL disassembler (ildasm.exe) to turn a binary assembly into a text assembly, we will see something similar to the following:

# Common Language Runtime (CLR)

- The CLR is the execution engine for .NET framework applications.
  - provides a common runtime environment for the execution of code written in any of the .NET languages.
  - It is a software-only, virtual platform that abstracts functionality from the operating system platform.

21 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY

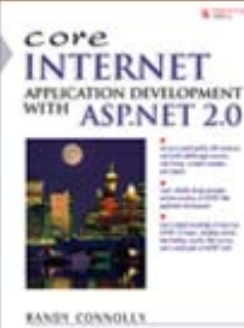


CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# CLR

- Conceptually, the CLR and Java's JVM are similar in that they are both runtime infrastructures that abstract the underlying platform differences.
  - However, while the JVM officially supports only the Java language, the CLR supports multiple languages
    - The JVM executes bytecode, so it too could, in principle, support languages other than Java.
  - Unlike Java's bytecode, though, .NET's MSIL is never interpreted.
  - Another conceptual difference is that Java code runs on any platform with a JVM, whereas .NET code runs only on platforms that support the CLR (currently only Windows).



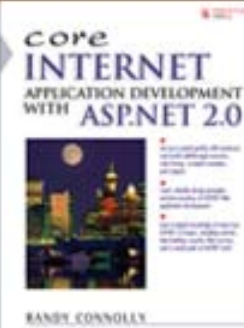
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# .NET Framework Base Class Library

- The .NET Framework base class library (BCL) is a large set of standard classes and other types.
- This library includes classes for:
  - working with the base types and exceptions,
  - representing common data structures and collections,
  - performing data access,
  - constructing Windows and Web interfaces.

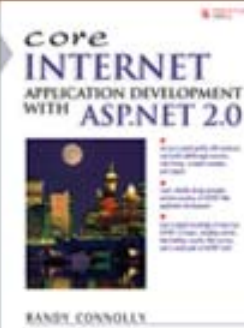


CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# BCL

- These classes are hierarchically organized into **logical** containers called **namespaces**.
- These namespaces are somewhat analogous to Java packages, except that .NET namespaces, unlike Java packages, are not also **physical** containers.
- Namespaces prevent name clashes
  - e.g., two assemblies each with a class named Image.
    - `System.Windows.Forms.Image` VS. `System.Web.UI.Image`
- Recall that compiled .NET code (i.e., MSIL) is physically stored in assemblies.
  - an assembly can contain classes in multiple namespaces, or
  - classes within a namespace can be physically partitioned across multiple assemblies.



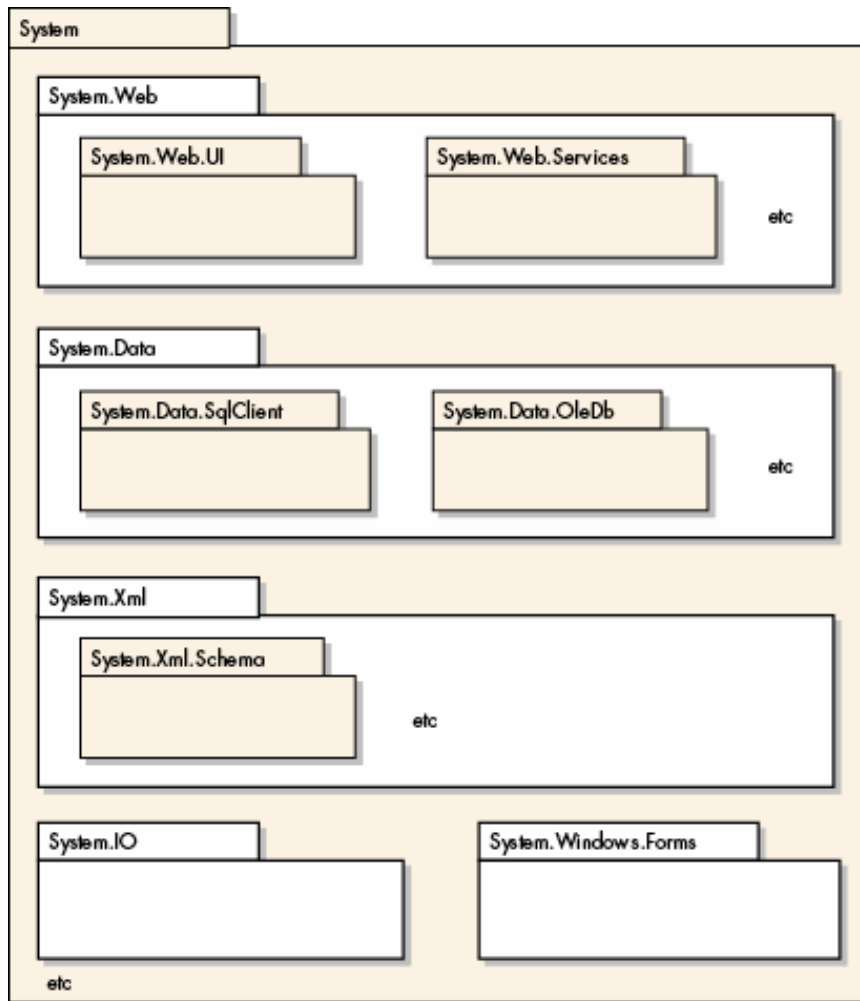
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Partial .NET Namespaces

25 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY



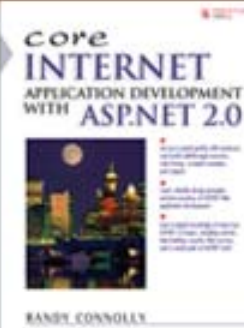
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)



# .NET Execution

- .NET programs written in a CLS-compliant language are compiled by the appropriate language compiler into MSIL.
- The MSIL is persisted into one or more assemblies.
- The CLR is then involved in the execution of the MSIL-based programs. This involves:
  - The CLR will invoke the JIT (Just-In-Time) compiler as needed to convert the MSIL into the appropriate machine code.
  - The CLR will execute the resulting machine code but manage code-level security and garbage collection.

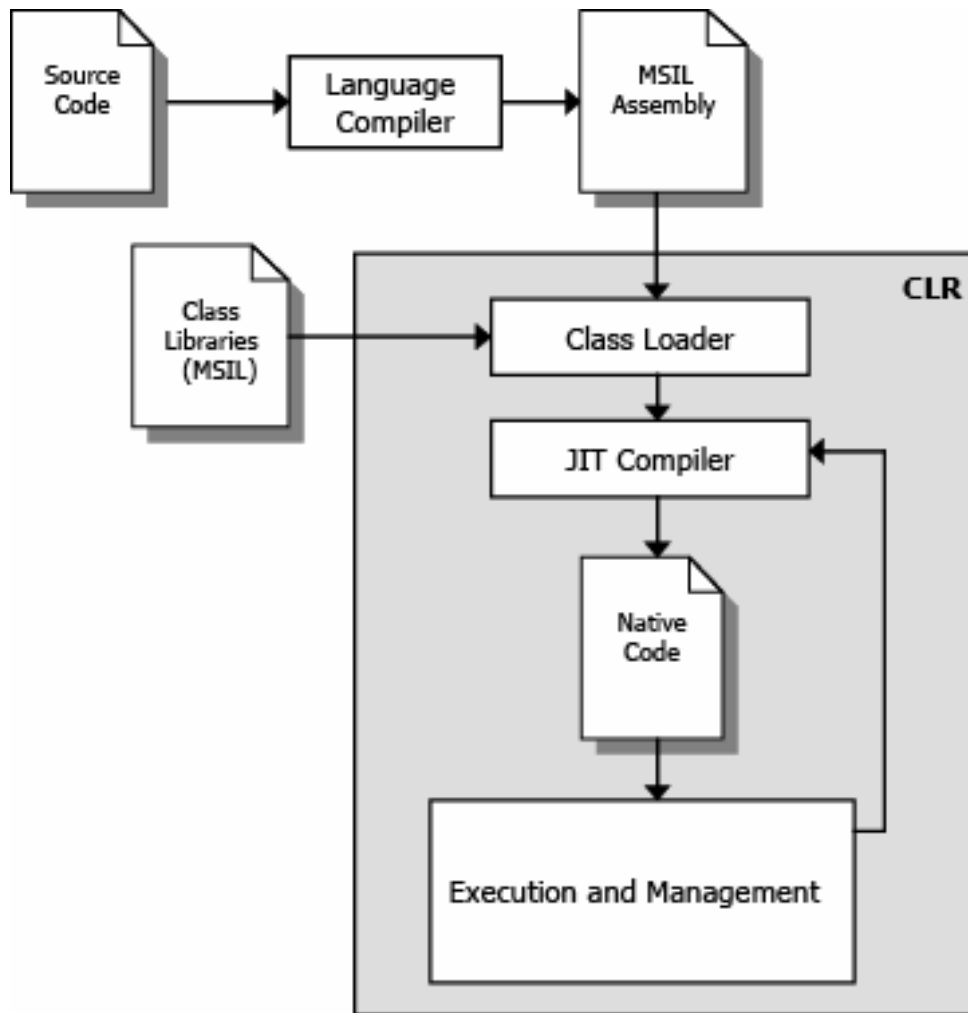


CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# .NET Execution



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

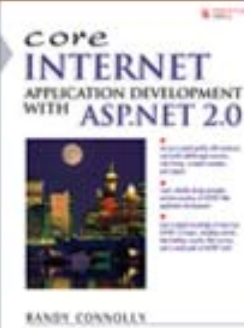
Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)



# ASP.NET Web Forms

- An ASP.NET web application :
  - Consists of any number of web pages, controls, programming classes, web services, and other files
  - Residing within a single web server application directory

28 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY



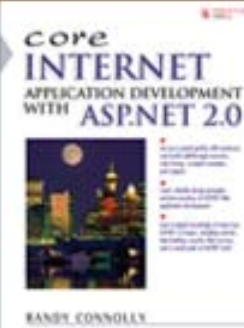
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# ASP.NET Web Forms

- The principal component of an ASP.NET web application are its web pages.
- These are text files with an .aspx extension and are called **web forms**.
- Consist of two parts:
  - The declaratively-defined (i.e., by markup/tags) visual elements.
  - The programming logic.

29 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

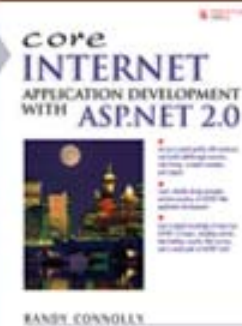
Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Web Form Programming Logic

- A web form's programming logic can exist in either:
  - The same file as the visual elements
    - i.e., the `.aspx` file.
    - This code is contained within a **code-declaration block**.
  - In a separate class file.
    - The file is usually called a **code-behind file**.
    - By convention, its filename is same as `.aspx` file but with a language extension.
      - `HelloWorld.aspx`      <- web form
      - `HelloWorld.aspx.cs`   <- code-behind file

30 Introducing ASP.NET 2.0

COPYRIGHT © 2007 RANDY CONNOLLY



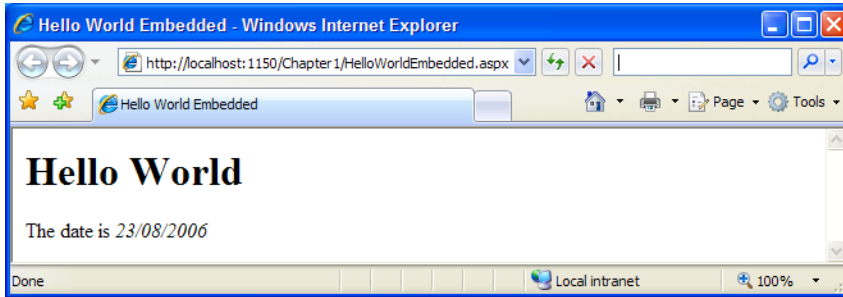
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

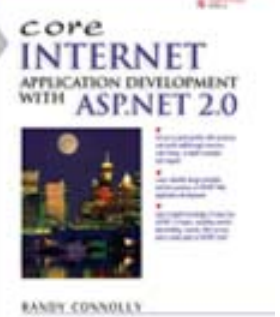
# HelloWorld.aspx Example

31 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)



`<%@ Page Language="C#" %>` ← Page directive

`<!DOCTYPE html PUBLIC ... >`

```
<script runat="server">
    protected void Page_Load(object sender, EventArgs e)
    {
        myDate.Text = DateTime.Now.ToShortDateString();
    }
</script>
```

Code declaration block

```
<html>
<head><title>Hello World Embedded</title></head>
<body>
    <form id="form1" runat="server" >
        <h1>Hello World</h1>
        The date is <em>
            <asp:Label ID="myDate" runat="server"></asp:Label>
        </em>
    </form>
</body>
</html>
```

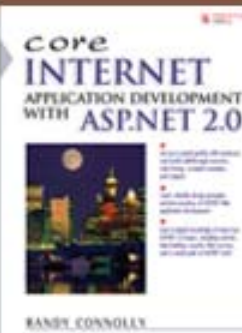
← Necessary to make this a web form

← Web server control

# Page Directive

```
<%@ Page Language="C#" %>
```

- The Page directive is used to define page-specific instructions used by the ASP.NET environment.
  - This directive indicates that the programming code in any script blocks will be C#.
- **Directives** are processing instructions for the parser and compiler when they process an ASP.NET page.
  - Although directives can be located anywhere in an .aspx file, the standard practice is to place them at the beginning of the file.
  - Directive statements are not case sensitive and quotation marks are not required around the attribute values.



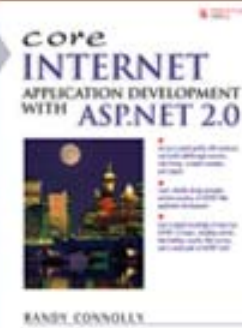
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

## <script> Element

```
<script runat="server">
    protected void Page_Load(object sender, EventArgs e)
    {
        myDate.Text = DateTime.Now.ToShortDateString();
    }
</script>
```

- Notice that the `<script>` element contains a `runat="server"` attribute.
- This tells the ASP.NET environment that the code within the block is to be executed on the server (i.e., it is not client-side Javascript or VBScript).
- The code itself declares a page-level event handler method that sets the `Text` property of a control named `myDate` to a short version of the current date.



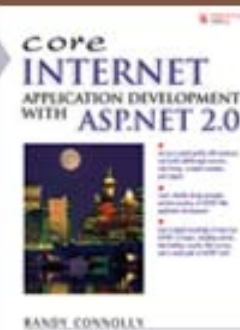
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

## <asp:Label> Web Control

```
<asp:Label ID="myDate" runat="server"></asp:Label>
```

- In the example's markup, there is an element named `<asp:Label>`.
  - This is a predefined ASP.NET Web server control (covered in more detail later).
  - The markup for this `Label` control sets its `ID` property to the value `myDate`.
- The `Label` control must contain the `runat="server"` attribute.
  - If it does not have it, the ASP.NET environment simply passes on the markup to the browser, where it will be ignored.



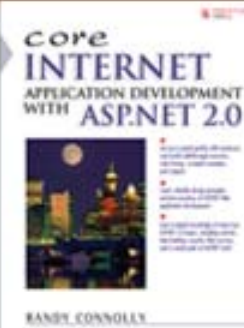
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

## <form> element

```
<form id="form1" runat="server" >  
  ...  
</form>
```

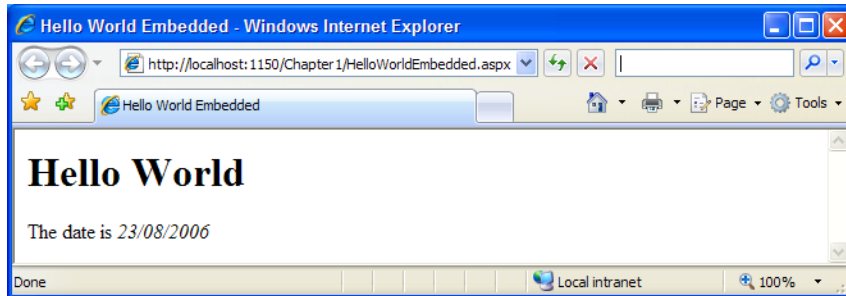
- All ASP.NET pages **must** contain a <form> element that contains this `runat` attribute.
- All body content in a Web form should appear within this special <form> element.



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

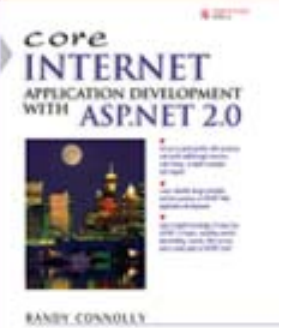
Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Result in the browser



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)



```
<html>
<head><title>Hello World Embedded</title></head>
<body>
  <form name="form1" method="post" action="HelloWorld.aspx" id="form1">
    <input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
value="/wEPDwUJODExMDE5NzY5D2QWAgIDD2QWAgIBDw8WAh4EYGV4dAUKMDgvMDEvMjAwNmRkZDZPhFHJE
R4chf3nmlgfL+uq4W58" />

    <h1>Hello World</h1>
    The date is <em>
    <span id="myDate">23/06/2006</span>
    </em>
  </form>
</body>
</html>
```

**Notice no `<asp:Label>` control.**

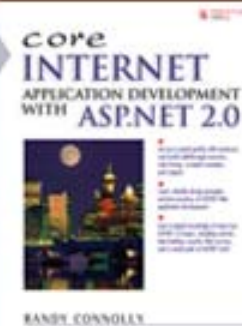
**Notice also the hidden input tag with the name of `__VIEWSTATE`. We will learn more about this view state in Chapter 2.**

## Example using Code-Behind

- Remember that a web form can alternately have its programming code contained in a separate class file called a code-behind file.

37 Introducing ASP.NET 2.0

COPYRIGHT © 2007 RANDY CONNOLLY



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Code Behind Version

```
<%@ Page Language="C#" AutoEventWireup="true"
    CodeFile="HelloWorldCodeBehind.aspx.cs"
    Inherits="HelloWorldCodeBehind" %>
```

Page  
directive

```
<!DOCTYPE ... >

<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
    <title>Hello World Code</title>
</head>
<body>
    <form id="form1" runat="server" >
        <h1>Hello World</h1>
        The date is <em>
        <asp:Label ID="myDate" runat="server"></asp:Label>
        </em>
    </form>
</body>
</html>
```

HelloWorldCodeBehind.aspx.cs

HelloWorldCodeBehind.aspx

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class HelloWorldCodeBehind : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        myDate.Text = DateTime.Now.Date.ToString();
    }
}
```

CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

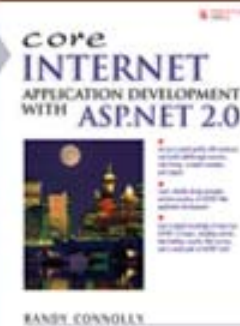
Core  
INTERNET  
APPLICATION DEVELOPMENT  
WITH ASP.NET 2.0



RANDY CONNOLLY

## Why use code-behind?

- The real advantage of separating the code into its own file is that it may lead to more maintainable web forms.
  - One of the main benefits of ASP.NET is that a page's programming logic can be **conceptually** separated from the presentation
  - by using a code-behind file a page's programming logic can also be **physically** separated from the presentation/markup.
- By placing the programming code into its own file, it is also potentially easier to make use of a division of labor in the creation of the site.
- Use whichever model you want
  - However, all the examples in text use code-behind.



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

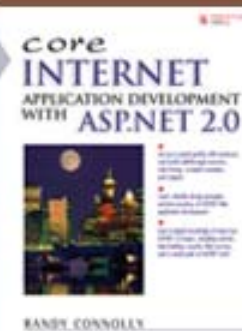
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Web Application Structure

- An ASP.NET web application can simply consist of a folder containing web forms and other files.
- You can, however, add any number of additional nested subfolders within this root folder.
- ASP.NET in fact has a number of reserved application folder names, e.g.
  - App\_Code
  - App\_Data
  - App\_Theme

40 Introducing ASP.NET 2.0

COPYRIGHT © 2007 RANDY CONNOLLY



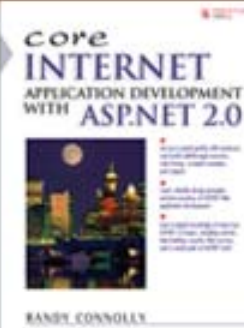
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Configuration File

- Every folder in a web application can contain an XML-format configuration file named `web.config`.
- This configuration file is used to define security, connection strings, and other configuration information.



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

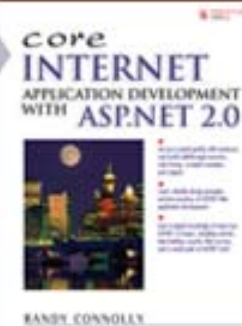
Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

## Visual Studio

- While you can create an ASP.NET application in any text editor, using Visual Studio will make developing ASP.NET applications easier.
  - Can also use the free Visual Web Developer 2005 Express Edition.

42 Introducing ASP.NET 2.0

COPYRIGHT © 2007 RANDY CONNOLLY



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

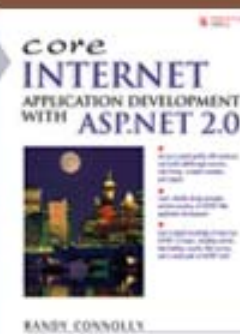
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

## Visual Studio Web Projects

- All files, folders, and settings in a Web application created with Visual Studio are contained within conceptual containers called **solutions** and **projects**.
- Depending upon which way you use Visual Studio to construct your Web application, a solution may contain one or more additional projects as well as additional files and metadata about the project.

43 Introducing ASP.NET 2.0

COPYRIGHT © 2007 RANDY CONNOLLY



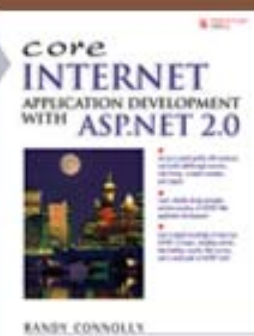
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Visual Studio Web Projects

- Visual Studio provides two ways of constructing a Web application with projects.
  - Web site project
    - This approach uses the content and structure of the site's folder to define the contents of the Visual Studio project.
    - There is no Visual Studio project file; instead, the content of the Web site project is directly inferred by Visual Studio from the folder structure and its contents.
  - Web application project
    - Rather than use the file structure of the site, this approach uses a separate Visual Studio project file (with `.csproj` or `.vbproj` extension) to maintain a list of files that belong to the project.



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Web Server Options

- To test or run ASP.NET Web applications, web server software is necessary.
- Microsoft's production web server software is **Internet Information Services (IIS)**.
  - In order to run IIS, your computer's operating system **must** be one of the following:
    - Windows 2000,
    - Windows XP Professional (not XP Home),
    - Windows Vista Business or Ultimate (not Vista Home Basic),
    - Windows Server 2003.

45 Introducing ASP.NET 2.0

COPYRIGHT © 2007 RANDY CONNOLLY

core  
**INTERNET**  
APPLICATION DEVELOPMENT  
WITH **ASP.NET 2.0**



RANDY CONNOLLY

CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

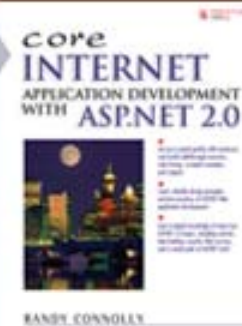
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

## Visual Studio File Server

- One of the advantages of using Visual Studio 2005 for ASP.NET development is that your site can be run and tested with or without using IIS.
- Visual Studio supports a variety of configurations:
  - local IIS,
  - file system,
  - FTP,
  - remote IIS sites

46 Introducing ASP.NET 2.0

COPYRIGHT © 2007 RANDY CONNOLLY



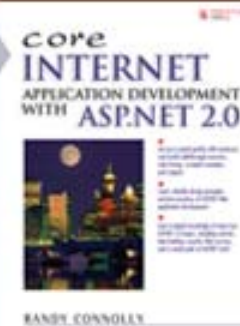
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

## File System Web Sites

- You can test your ASP.NET pages as a **file system web site** if you are using Visual Studio 2005.
- In a file system Web site, you can create and edit files in any folder you like, whether on your local computer or in a folder on another computer that you access via network share, by using the Visual Studio 2005 web server.
- The Visual Studio web server can run locally on all current versions of Windows, including Windows XP Home.
- The Visual Studio Web server accepts only localhost requests.
  - It cannot serve pages to another computer, and is therefore suitable only for testing pages locally.



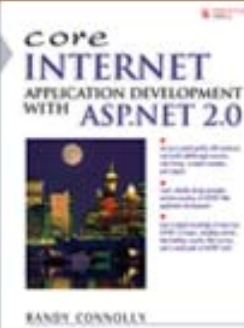
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

## Testing using IIS

- If your computer has Microsoft's IIS installed, then you can run a local IIS website.
- There are two ways of using a local IIS web site with Visual Studio:
  - you can run the Internet Information Services snap-in from the Windows Control Panel and create an IIS **virtual directory**, which is like a pointer that references the physical folder for your web site.
  - you can let Visual Studio create the IIS virtual directory.



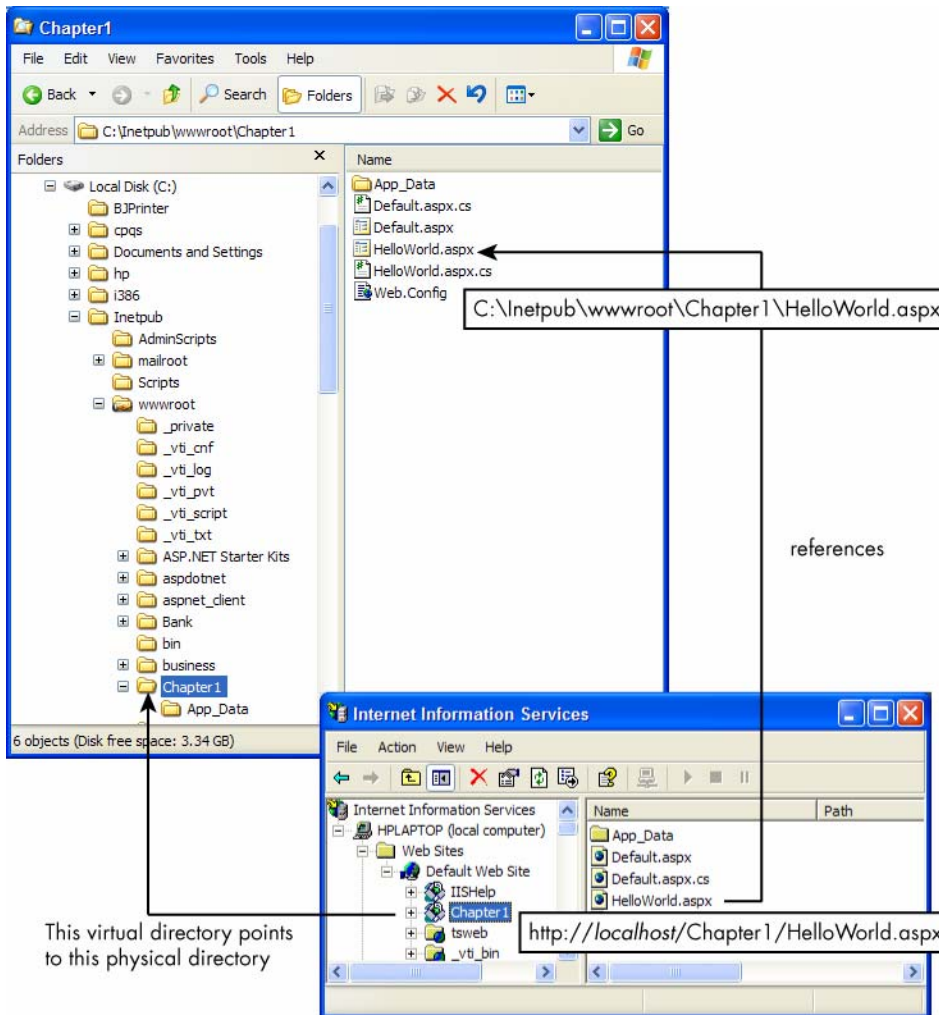
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Virtual Directories vs Physical Folders

49 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)



## Web Server Options

- With Visual Studio 2005, we could say that your computer is a **development server** (available only to yourself).
- Alternately, you might upload your work to a **staging server** running IIS for team testing.
- Of course, if you want others to see this Web application, it must eventually be uploaded to a **production server** (available to your Web application's audience).

50 Introducing ASP.NET 2.0

COPYRIGHT © 2007 RANDY CONNOLLY



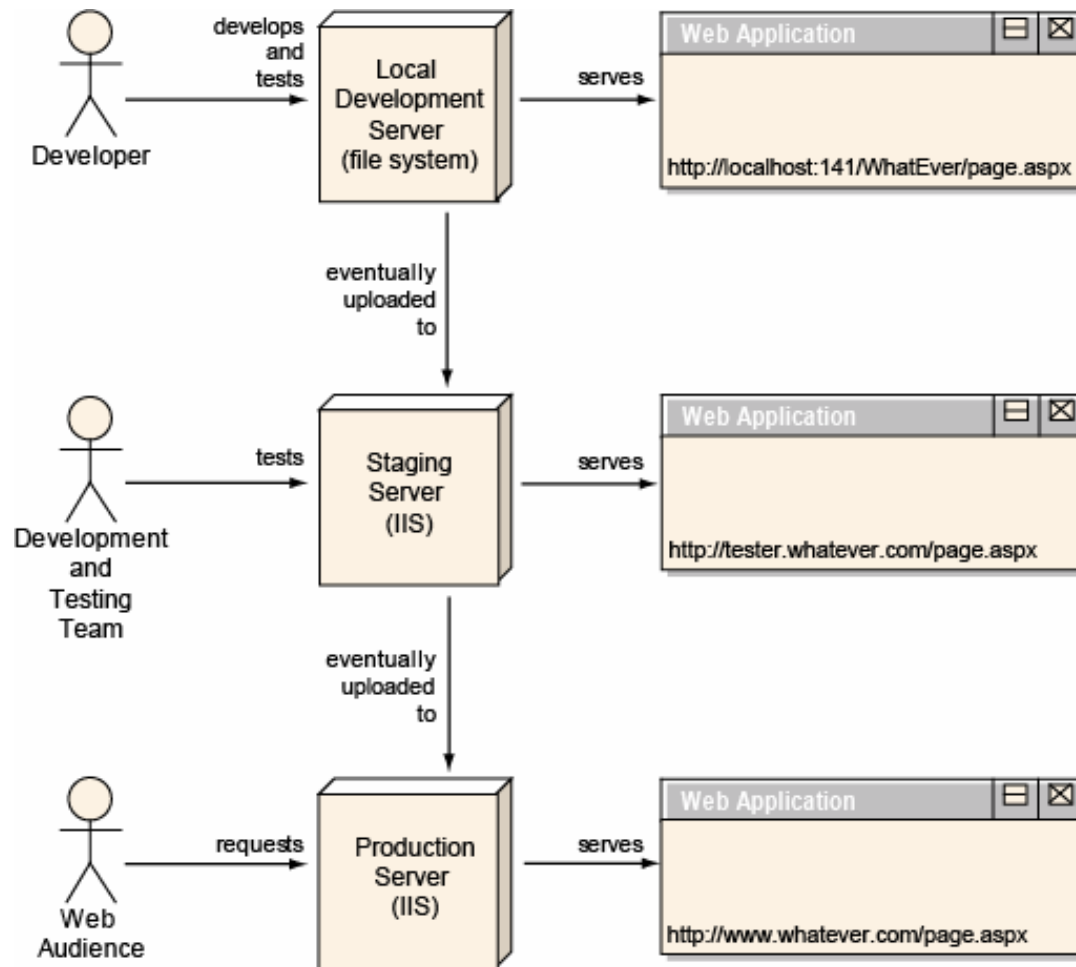
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Web Server Options

51 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY



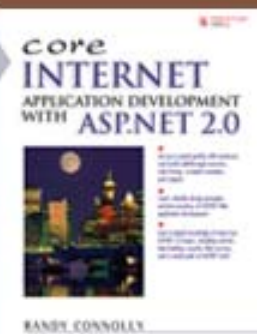
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)



# ASP.NET Tutorial

- Walkthrough 1.1 (p. 30-1)
  - Creating a New Web Site in Visual Studio
- Walkthrough 1.2 (p. 32)
  - Adding a web form to a project
- Walkthrough 1.3 (p. 34-6)
  - Adding markup content to a web form
- Walkthrough 1.4 (p. 37-9)
  - Test viewing a form
- Walkthrough 1.5 (p. 42-3)
  - Adding programming logic to a web form
- Walkthrough 1.6 (p. 43)
  - Adding errors to your page
- Walkthrough 1.7 (p. 45-6)
  - Adding a parser error to your page
- Walkthrough 1.8 (p. 46-50)
  - Using the debugger



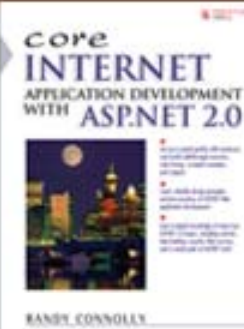
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Programming Logic

- Code can be contained within:
  - Within code-behind class
  - Within code declaration blocks
    - i.e., within the `<script runat="server">` element
  - Within code render blocks within the markup.
    - i.e., `<% inline code here %>`
    - Although the use of code render blocks is familiar to ASP programmers, **their use is very much discouraged in ASP.NET.**
    - In the vast majority of cases, we can replace code render blocks with a combination of server controls and programming within either code declaration blocks or within the page's code-behind file.



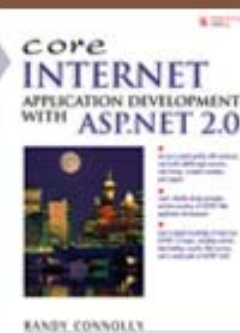
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

## Web Server Controls

- Normal HTML elements such as `<input>`, `<h1>`, and `<select>` are not processed by the server but are sent to and displayed by the browser.
- Server controls, in contrast, are tags that are processed by the server.
- Each ASP.NET server control has an object model containing properties, methods, and events.
- You can use this object model to programmatically interact with the control.



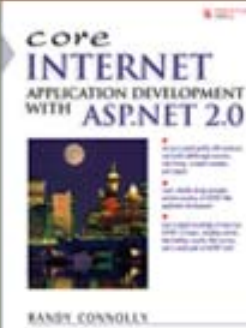
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Web Server Controls

- Web server controls are added to a Web forms page in the same way as any HTML element.
  - That is, you can type the markup code in Source view, or use drag-and-drop from Design view.
  - As well, you can also programmatically add controls at runtime.

55 Introducing ASP.NET 2.0  
COPYRIGHT © 2007 RANDY CONNOLLY



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

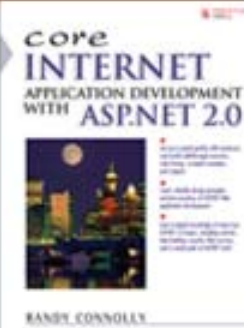
Prentice Hall, 2007  
[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# Web Server Controls

- ASP.NET 2.0 defines over 60 built-in Web server controls, all of which have a tag prefix of `asp`.
- The two possible syntaxes for declaring a server control in your markup are:

```
<tagprefix:tagname ID="myName" runat="server">  
</tagprefix:tagname>
```

```
<tagprefix:tagname ID="myName" runat="server" />
```



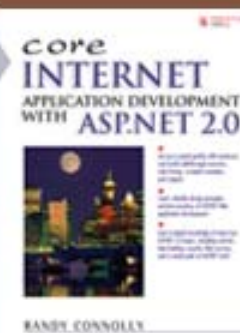
CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)

# HTML Server Controls

- ASP.NET provides a special type of server control called the **HTML server control** that has a different syntax from that shown just previously.
- HTML server controls look like standard HTML tags, except for one thing:
  - They contain a `runat="server"` attribute.
  - HTML server controls are HTML elements that contain attributes that make them programmable on the server.
- Because HTML server controls only have as much functionality as the available HTML elements, this book does not in fact spend any time working with them.
  - Instead, it focuses on using the much more powerful Web server controls.



CORE INTERNET  
APPLICATION DEVELOPMENT  
WITH  
ASP.NET 2.0

Prentice Hall, 2007

[www.randyconnolly.com/core](http://www.randyconnolly.com/core)