Microsoft

OFFICIAL MICROSOFT LEARNING PRODUCT

10267A

Introduction to Web Development with Microsoft® Visual Studio® 2010

Companion Content

Product Number: 10267A

Released: 07/2010

MICROSOFT LICENSE TERMS

OFFICIAL MICROSOFT LEARNING PRODUCTS COURSEWARE – STUDENT EDITION – Pre-Release and Final Versions

These license terms are an agreement between Microsoft Corporation and you. Please read them. They apply to the licensed content named above, which includes the media on which you received it, if any. The terms also apply to any Microsoft

- updates,
- supplements,
- Internet-based services, and
- support services

for this licensed content, unless other terms accompany those items. If so, those terms apply.

By using the licensed content, you accept these terms. If you do not accept them, do not use the licensed content.

If you comply with these license terms, you have the rights below.

1. **OVERVIEW.**

   Licensed Content. The licensed content includes software, printed materials, academic materials (online and electronic), and associated media.

   License Model. The licensed content is licensed on a per copy per device basis.

2. **INSTALLATION AND USE RIGHTS.**

   a. **Licensed Device.** The licensed device is the device on which you use the licensed content. You may install and use one copy of the licensed content on the licensed device.

   b. **Portable Device.** You may install another copy on a portable device for use by the single primary user of the licensed device.

   c. **Separation of Components.** The components of the licensed content are licensed as a single unit. You may not separate the components and install them on different devices.

   d. **Third Party Programs.** The licensed content may contain third party programs. These license terms will apply to your use of those third party programs, unless other terms accompany those programs.

3. **PRE-RELEASE VERSIONS.** If the licensed content is a pre-release ("beta") version, in addition to the other provisions in this agreement, then these terms also apply:

   a. **Pre-Release Licensed Content.** This licensed content is a pre-release version. It may not contain the same information and/or work the way a final version of the licensed content will. We may change it for the final, commercial version. We also may not release a commercial version. You will clearly and conspicuously inform any Students who participate in an Authorized Training Session and any Trainers who provide training in such Authorized Training Sessions of the foregoing; and, that you or Microsoft are under no obligation to provide them with any further content, including but not limited to the final released version of the Licensed Content for the Course.

   b. **Feedback.** If you agree to give feedback about the licensed content to Microsoft, you give to Microsoft, without charge, the right to use, share and commercialize your feedback in any way and for any purpose. You also give to third parties, without charge, any patent rights needed for their products, technologies and services to use or interface with any specific parts of a Microsoft software, licensed content, or service that includes the feedback. You will not give feedback that is subject to a license that requires Microsoft to license its software or documentation to third parties because we include your feedback in them. These rights survive this agreement.

   c. **Confidential Information.** The licensed content, including any viewer, user interface, features and documentation that may be included with the licensed content, is confidential and proprietary to Microsoft and its suppliers.

      i. **Use.** For five years after installation of the licensed content or its commercial release, whichever is first, you may not disclose confidential information to third parties. You may disclose confidential information only to your employees and consultants who need to know the information. You must have written agreements with them that protect the confidential information at least as much as this agreement.

      ii. **Survival.** Your duty to protect confidential information survives this agreement.

iii. **Exclusions**. You may disclose confidential information in response to a judicial or governmental order. You must first give written notice to Microsoft to allow it to seek a protective order or otherwise protect the information. Confidential information does not include information that

- becomes publicly known through no wrongful act;
- you received from a third party who did not breach confidentiality obligations to Microsoft or its suppliers; or
- you developed independently.

d. **Term**. The term of this agreement for pre-release versions is (i) the date which Microsoft informs you is the end date for using the beta version, or (ii) the commercial release of the final release version of the licensed content, whichever is first ("beta term").

e. **Use**. You will cease using all copies of the beta version upon expiration or termination of the beta term, and will destroy all copies of same in the possession or under your control.

f. **Copies**. Microsoft will inform Authorized Learning Centers if they may make copies of the beta version (in either print and/or CD version) and distribute such copies to Students and/or Trainers. If Microsoft allows to such distribution, you will follow any additional terms that Microsoft provides to you for such copies and distribution.

4. **ADDITIONAL LICENSING REQUIREMENTS AND/OR USE RIGHTS.**

a. Media Elements and Templates. You may use images, clip art, animations, sounds, music, shapes, video clips and templates provided with the licensed content solely for your personal training use. If you wish to use these media elements or templates for any other purpose, go to www.microsoft.com/permission to learn whether that use is allowed.

b. Academic Materials. If the licensed content contains academic materials (such as white papers, labs, tests, datasheets and FAQs), you may copy and use the academic materials. You may not make any modifications to the academic materials and you may not print any book (either electronic or print version) in its entirety. If you reproduce any academic materials, you agree that:

- The use of the academic materials will be only for your personal reference or training use
- You will not republish or post the academic materials on any network computer or broadcast in any media;
- You will include the academic material's original copyright notice, or a copyright notice to Microsoft's benefit in the format provided below:

Form of Notice:

c. Distributable Code. The licensed content may contain code that you are permitted to distribute in programs you develop if you comply with the terms below.

i. Right to Use and Distribute. The code and text files listed below are "Distributable Code."

- REDIST.TXT Files. You may copy and distribute the object code form of code listed in REDIST.TXT files.
- Sample Code. You may modify, copy, and distribute the source and object code form of code marked as "sample."
- Third Party Distribution. You may permit distributors of your programs to copy and distribute the Distributable Code as part of those programs.

ii. Distribution Requirements. For any Distributable Code you distribute, you must

- add significant primary functionality to it in your programs;
- require distributors and external end users to agree to terms that protect it at least as much as this agreement;
- display your valid copyright notice on your programs; and
- indemnify, defend, and hold harmless Microsoft from any claims, including attorneys' fees, related to the distribution or use of your programs.

iii. **Distribution Restrictions.** You may not

- alter any copyright, trademark or patent notice in the Distributable Code;
- use Microsoft's trademarks in your programs' names or in a way that suggests your programs come from or are endorsed by Microsoft;
- distribute Distributable Code to run on a platform other than the Windows platform;
- include Distributable Code in malicious, deceptive or unlawful programs; or
- modify or distribute the source code of any Distributable Code so that any part of it becomes subject to an Excluded License. An Excluded License is one that requires, as a condition of use, modification or distribution, that
  - the code be disclosed or distributed in source code form; or
  - others have the right to modify it.

5. **INTERNET-BASED SERVICES.** Microsoft may provide Internet-based services with the licensed content. It may change or cancel them at any time. You may not use these services in any way that could harm them or impair anyone else's use of them. You may not use the services to try to gain unauthorized access to any service, data, account or network by any means.
6. **SCOPE OF LICENSE.** The licensed content is licensed, not sold. This agreement only gives you some rights to use the licensed content. Microsoft reserves all other rights. Unless applicable law gives you more rights despite this limitation, you may use the licensed content only as expressly permitted in this agreement. In doing so, you must comply with any technical limitations in the licensed content that only allow you to use it in certain ways. You may not
   - disclose the results of any benchmark tests of the licensed content to any third party without Microsoft's prior written approval;
   - work around any technical limitations in the licensed content;
   - reverse engineer, decompile or disassemble the licensed content, except and only to the extent that applicable law expressly permits, despite this limitation;
   - make more copies of the licensed content than specified in this agreement or allowed by applicable law, despite this limitation;
   - publish the licensed content for others to copy;
   - transfer the licensed content marked as 'beta' or 'pre-release' to any third party;
   - allow others to access or use the licensed content;
   - rent, lease or lend the licensed content; or
   - use the licensed content for commercial licensed content hosting services.
   - Rights to access the server software that may be included with the Licensed Content, including the Virtual Hard Disks does not give you any right to implement Microsoft patents or other Microsoft intellectual property in software or devices that may access the server.
7. **BACKUP COPY.** You may make one backup copy of the licensed content. You may use it only to reinstall the licensed content.
8. **TRANSFER TO ANOTHER DEVICE.** You may uninstall the licensed content and install it on another device for your personal training use. You may not do so to share this license between devices.
9. **TRANSFER TO A THIRD PARTY.** You may not transfer those versions marked as 'beta' or 'pre-release' to a third party. For final versions, these terms apply: The first user of the licensed content may transfer it and this agreement directly to a third party. Before the transfer, that party must agree that this agreement applies to the transfer and use of the licensed content. The first user must uninstall the licensed content before transferring it separately from the device. The first user may not retain any copies.
10. **EXPORT RESTRICTIONS.** The licensed content is subject to United States export laws and regulations. You must comply with all domestic and international export laws and regulations that apply to the licensed content. These laws include restrictions on destinations, end users and end use. For additional information, see www.microsoft.com/exporting.
11. **NOT FOR RESALE SOFTWARE/LICENSED CONTENT.** You may not sell software or licensed content marked as "NFR" or "Not for Resale."

12. **ACADEMIC EDITION.** You must be a "Qualified Educational User" to use licensed content marked as "Academic Edition" or "AE." If you do not know whether you are a Qualified Educational User, visit www.microsoft.com/education or contact the Microsoft affiliate serving your country.

13. **ENTIRE AGREEMENT.** This agreement, and the terms for supplements, updates, Internet-based services and support services that you use, are the entire agreement for the licensed content and support services.

14. **APPLICABLE LAW.**

    a. **United States.** If you acquired the licensed content in the United States, Washington state law governs the interpretation of this agreement and applies to claims for breach of it, regardless of conflict of laws principles. The laws of the state where you live govern all other claims, including claims under state consumer protection laws, unfair competition laws, and in tort.

    b. **Outside the United States.** If you acquired the licensed content in any other country, the laws of that country apply.

15. **LEGAL EFFECT.** This agreement describes certain legal rights. You may have other rights under the laws of your country. You may also have rights with respect to the party from whom you acquired the licensed content. This agreement does not change your rights under the laws of your country if the laws of your country do not permit it to do so.

16. **DISCLAIMER OF WARRANTY. THE LICENSED CONTENT IS LICENSED "AS-IS." YOU BEAR THE RISK OF USING IT. MICROSOFT GIVES NO EXPRESS WARRANTIES, GUARANTEES OR CONDITIONS. YOU MAY HAVE ADDITIONAL CONSUMER RIGHTS UNDER YOUR LOCAL LAWS WHICH THIS AGREEMENT CANNOT CHANGE. TO THE EXTENT PERMITTED UNDER YOUR LOCAL LAWS, MICROSOFT EXCLUDES THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT.**

17. **LIMITATION ON AND EXCLUSION OF REMEDIES AND DAMAGES. YOU CAN RECOVER FROM MICROSOFT AND ITS SUPPLIERS ONLY DIRECT DAMAGES UP TO U.S. $5.00. YOU CANNOT RECOVER ANY OTHER DAMAGES, INCLUDING CONSEQUENTIAL, LOST PROFITS, SPECIAL, INDIRECT OR INCIDENTAL DAMAGES.**

    This limitation applies to

    - anything related to the licensed content, software, services, content (including code) on third party Internet sites, or third party programs; and
    - claims for breach of contract, breach of warranty, guarantee or condition, strict liability, negligence, or other tort to the extent permitted by applicable law.

    It also applies even if Microsoft knew or should have known about the possibility of the damages. The above limitation or exclusion may not apply to you because your country may not allow the exclusion or limitation of incidental, consequential or other damages.

Please note: As this licensed content is distributed in Quebec, Canada, some of the clauses in this agreement are provided below in French.

Remarque : Ce le contenu sous licence étant distribué au Québec, Canada, certaines des clauses dans ce contrat sont fournies ci-dessous en français.

EXONÉRATION DE GARANTIE. Le contenu sous licence visé par une licence est offert « tel quel ». Toute utilisation de ce contenu sous licence est à votre seule risque et péril. Microsoft n'accorde aucune autre garantie expresse. Vous pouvez bénéficier de droits additionnels en vertu du droit local sur la protection dues consommateurs, que ce contrat ne peut modifier. La ou elles sont permises par le droit locale, les garanties implicites de qualité marchande, d'adéquation à un usage particulier et d'absence de contrefaçon sont exclues.

LIMITATION DES DOMMAGES-INTÉRÊTS ET EXCLUSION DE RESPONSABILITÉ POUR LES DOMMAGES. Vous pouvez obtenir de Microsoft et de ses fournisseurs une indemnisation en cas de dommages directs uniquement à hauteur de 5,00 $ US. Vous ne pouvez prétendre à aucune indemnisation pour les autres dommages, y compris les dommages spéciaux, indirects ou accessoires et pertes de bénéfices.

Cette limitation concerne:

- tout ce qui est relié au le contenu sous licence , aux services ou au contenu (y compris le code) figurant sur des sites Internet tiers ou dans des programmes tiers ; et
- les réclamations au titre de violation de contrat ou de garantie, ou au titre de responsabilité stricte, de négligence ou d'une autre faute dans la limite autorisée par la loi en vigueur.

Elle s'applique également, même si Microsoft connaissait ou devrait connaître l'éventualité d'un tel dommage. Si votre pays n'autorise pas l'exclusion ou la limitation de responsabilité pour les dommages indirects, accessoires ou de quelque nature que ce soit, il se peut que la limitation ou l'exclusion ci-dessus ne s'appliquera pas à votre égard.

EFFET JURIDIQUE. Le présent contrat décrit certains droits juridiques. Vous pourriez avoir d'autres droits prévus par les lois de votre pays. Le présent contrat ne modifie pas les droits que vous confèrent les lois de votre pays si celles-ci ne le permettent pas.

Module 1

Exploring Microsoft® ASP.NET Web Applications in Microsoft Visual Studio® 2010

Contents:

Lesson 1

Introduction to the .NET Framework

Contents:

Question and Answers

Introduction to Microsoft .NET

Question: What is the biggest advantage to using the .NET Framework to develop your application?

Answer: .NET Framework is a platform-independent and device-independent environment that works across environment barriers.

Benefits of the .NET Framework

Question: What are the main advantages of using the .NET Framework to develop Web applications?

Answer: The main advantages of using the.NET Framework to develop Web applications are:

- It provides an entire framework (ASP.NET) for developing Web-based applications.
- You can develop Web applications, Web sites, or WCF services.
- You can create anything from small, personal Web sites, to large, enterprise-level Web applications.

Components of the .NET Framework

Question: What are the two main components of the .NET Framework 4?

Answer: Two main components are CLR, and the .NET Framework class library (BCL).

Key Features of Visual Studio 2010

Question: Why is Visual Studio 2010 considered to be a complete set of development tools?

Answer: Visual Studio provides a complete set of development tools for building ASP.NET Web applications, services, desktop applications, and cloud-based and mobile applications. Visual Basic, Visual C#, and Visual C++ all use the same integrated development environment (IDE), which enables tool sharing, and eases the creation of mixed-programming language solutions. In addition, these languages use the .NET Framework functionality, which provides access to key technologies that simplify the ASP.NET Web applications development.

Additional Reading

Introduction to Microsoft .NET

The following list includes links to a number of servers capable of hosting the .NET Framework:

- Windows Server 2008 R2. Build and run sophisticated Web and server applications with the .NET Framework and Windows Server 2008
- Windows Azure Platform. The Windows Azure platform offers a flexible, familiar environment for developers to create cloud applications and services
- SQL Server 2008. You can create data-centric solutions for mobile devices, desktops, Web servers, and enterprise servers with SQL Server at the core of your comprehensive programming framework.
- SharePoint 2010. Built entirely on ASP.NET, Microsoft SharePoint Server 2010 uses the same .NET Framework, languages, class libraries, and development tools

Components of the .NET Framework

> For information about the supported operating systems for the .NET Framework version 4.0, see the System Requirements for Version 3.5 page on the Microsoft Web site.

For more information about CLS, see the Common Language Specification page on the Microsoft Web site.

> For more information about Windows Forms, see the Windows Forms page on the Microsoft Web site.

For more information about the .NET Framework Class Library, see the Base Class Libraries page on the Microsoft Web site.

Microsoft also supports two subsets of the .NET Framework, the .NET Compact Framework and the .NET Micro Framework, both supporting the .NET Framework on devices.

Key Features of Visual Studio 2010

> For more information about Visual Studio, see Introducing Visual Studio.

> For more information about programming in Visual Studio, see .NET Framework Programming in Visual Studio.

Lesson 2

Overview of ASP.NET

Contents:

Question and Answers

Components of ASP.NET Web Applications

Question: What are the tools in Visual Studio 2010 that make application development faster, easier, and more reliable?

Answer: Visual Studio tools include:

- Visual designers for Web pages, with drag-and-drop controls, and code (HTML) views with syntax checking.
- Code-aware editors that include statement completion, syntax checking, and other Microsoft IntelliSense® features.
- Integrated compilation and debugging.
- Project management facilities for creating and managing application files, including deployment to local or remote servers.

Additional Reading

What Is ASP.NET?

For more information about ASP.NET Health Monitoring, see the ASP.NET Health Monitoring Overview page.

For more information about the Extensible Designer Environment, see the ASP.NET Control Designers Overview page.

Generating and Rendering Markup and Code

For more information about compiling by using the Web application compilation model, see the Compiling Web Application Projects.

ASP.NET Dynamic Compilation Execution Model

For more information about precompiling a Web site, see the ASP.NET Precompilation Overview.

Lesson 3

Overview of the Lab Application

Contents:

Detailed Demo Steps

Demonstration: Exploring the .NET Framework

Demonstration steps

The main tasks are as follows:

1. Open Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
2. Open the finished lab solution.
    - In the Start page – Microsoft Visual Studio (Administrator) window, on the File menu, click **Open Project.**
    - In the **Open Project** dialog box, in the **File name** box, type **D:\Labfiles\Starter\M16\VB\CustomerManagement.sln** or **D:\Labfiles\Starter\M16\CS\CustomerManagement.sln**, and then click **Open**.
3. Examine some of the .NET Framework namespaces and classes, including **System.Web** and **System.Web.UI**, and examine the **System.Web.UI.Page** class.
4. Explore the Ajax Extensions and the Ajax Library features.
    - Run the solution, and view the About box by clicking **About** on the **Help** menu.

    Note: You will learn more about the features offered by Ajax Extensions and the Ajax Library in Module 11, "Creating a Microsoft® ASP.NET Ajax-Enabled Web Forms Application."
5. Explore how Dynamic Data works.
    - Run the solution.
    - On the **Customers** menu, click **All**.
    - On the **Countries** menu, click **All**.
6. If time permits, view code for some of the Module 6 and later lab exercises.

    Note: You will learn more about how Dynamic Data works in Module 10, "Managing Data by Using Microsoft® ASP.NET Dynamic Data."

    Result: At the end of this demonstration, you will have a basic understanding of the.NET Framework 4 features in the context of Web application development, including some the new functionalities, such as Ajax, and Dynamic Data.

Demonstration: Exploring the Views and Controls of Visual Studio 2010

The main tasks are as follows:

1. Open Microsoft Visual Studio 2010.

- On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.

2. Create a new ASP.NET Web site with the following settings:
    - Name: **WebSite1**
    - Location: **File system**
    - Language: **Visual Basic** or Visual **C#**
3. In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **New Web Site**.
4. In the **New Web Site** dialog box, in the left pane, under **Installed Templates**, click **Visual Basic** or **Visual C#**.
5. In the middle pane, ensure that **.NET Framework 4** is selected in the target framework list.
6. In the **New Web Site** dialog box, in the middle pane, click **ASP.NET Empty Web Site**.
7. In the **Web location** list, ensure that **File System** is selected, and then click **OK**.
8. Add the **Default.aspx** Web Form to the **CustomerManagement** Web site.
    - In Solution Explorer, right-click the Web site, and then click **Add New Item**.
    - In the **Add New Item** dialog box, in the middle pane, click **Web Form**, in the **Name** box, type **Default.aspx**, and then click **Add**.
9. Open the Web Form in both the Design view and the Source view.
    - In the Default.aspx window, click **Design**.
    - In the Default.aspx window, click **Source**.
    - In the Default.aspx window, click **Design**.
10. Add a **Button** control to the Web Form.
    - In the Default.aspx window, select the **div** element.
    - In the CustomerManagement – Microsoft Visual Studio (Administrator) window, point to **Toolbox**.
    - In **Toolbox**, expand **Standard**, and then double-click the **Button** control.
11. Add code to the **Button Click** event.
    - In the InsertCustomer.aspx window, click the **Button** control.
    - In the Properties window, click the **Events** icon, and then double-click the box next to the **Click** event.

Result: At the end of this demonstration, you will understand the advantages of using Visual Studio 2010 for developing ASP.NET applications.

Demonstration: Exploring the Components of an ASP.NET 4.0 Web Application Project

The main tasks are as follows:

1. Examine the solution and project files, and the references.

- In Solution Explorer, notice the solution and project files, and identify where project references go.

2. Examine the ASP.NET folders.
    - In Solution Explorer, review the ASP.NET folders, and if time allows create a new one.
3. Examine the Web Forms, designer code files, code-behind files, and class files.
    - In Solution Explorer, review the various files.
4. Examine the assembly info file and the web.config files.
    - In Solution Explorer, review the various files.

Result: At the end of this demonstration, you will have identified and examined the functionality and importance of the components of an ASP.NET 4.0 Web application.

Module Reviews and Takeaways

Review questions

1. Which components are included in the .NET Framework 4?

   The .NET Framework 4 consists of: the common language runtime component; the.NET Framework class library, data access; WCF services; Web Forms; Windows Forms; WPF; WCF; WF; LINQ; ASP.NET Ajax; ASP.NET Dynamic Data; and ADO.NET Entity Framework.

2. What is the purpose of CLR?

   The CLR component provides an environment in which you can run code. The runtime also provides a compiler, language independence, and portability. Finally, the CLR componentperforms garbage collection.

3. You need to create a programmable Web component that you can share among multiple Web applications. Which type of component should you create?

   You should create a Web-based WCF Service.

4. You need to develop an application that contains projects that target multiple versions of the .NET Framework. Which tool enables you to accomplish this?

   Visual Studio 2010 enables you to develop an application containing projects that target multiple .NET Framework versions.

Real-world issues and scenarios

1. You are unsure of the functionality of a particular class. How can you determine the functionality of the class?

   Use the .NET Framework documentation, Web sites, or online communities.

2. You are unsure whether to use dynamic compilation or precompilation. What are the factors that you need to consider?

   Use dynamic compilation in the following situations:

   - When you want to modify your source code without having to explicitly compile your code before deploying your Web application
   - When you want to extend the ASP.NET build system by creating custom build providers for new file types that are called during compilation
   - Use precompilation in the following situations:
   - When response time is a concern
   - To identify compilation errors before users access the site
   - To deploy the Web site to a production server without the source code

Tools

Tool	Purpose	Where to find it
Visual Studio 2010	Developing Web applications	Start menu

Lab Review Questions and Answers

Review questions and answers

1. What is the major advantage of using ASP.NET?

   Answer: The ASP.NET page framework is a scalable programming model that you can use on the server to dynamically generate Web pages.

2. What is the purpose of the code-behind files in ASP.NET?

   Answer: With code-behind files, you can create a single-file for the ASP.NET page, which contains both the page markup and the .NET Framework code in the same file.

3. What are Web Forms?

   Answer: Web Forms are pages that are processed at the server side, and then rendered as HTML pages to a client.

Module 2

Creating Web Applications by Using Microsoft® Visual Studio® 2010 and Microsoft .NET-Based Languages

Contents:

Lesson 1

Choosing a Programming Language

Contents:

Question and Answers

Features of Visual Basic

Question: How are namespaces used in Visual Basic?

Answer: Namespaces are a logical way of grouping related types, and they help reduce conflicts when using types with the same name. Namespaces are specific to the .NET Framework, but are not specific to individual programming languages. In Visual Basic—like other .NET Framework–based programming languages—each project has a default namespace; however, unlike most other programming languages, the namespace is automatically applied to all the types that you create. This means that if the default namespace is **CompanyName.ApplicationName**, and you have a namespace for a specific type—for example, **Services.Customers**—then the actual namespace for the type is **CompanyName.ApplicationName.Services.Customers**. If you do not prefix the type with a namespace, then the default namespace is automatically applied. For example, the default type would have the namespace of **CompanyName.ApplicationName**.

Features of Visual C#

Question: How are namespaces used in Visual C#?

Answer: In Visual C#, each project has a default namespace, which is automatically applied to all the types that you create. If the default namespace is **CompanyName.ApplicationName** and you have a namespace for a specific type—for example, **Services.Customers**—the actual namespace for the type is **Services.Customers**, because then the default namespace is ignored. In addition, types in subfolders of the project are automatically created with a namespace that is made up of the default project namespace, followed by the folder name. Therefore, if a type is stored in a subfolder named Test, and the default namespace is **CompanyName.ApplicationName**, the actual namespace for that type is **CompanyName.ApplicationName.Test**. You can override this by specifying the full namespace for the type in code.

Scenarios for Mixed-Language Environments

Question: In what real-world scenarios do you use mixed languages?

Answer: You might have to work with existing projects or solutions that have mixed code. You might need to transition projects and solutions from one programming language to another. Or, you might need to work with other developers who have a preference for a particular programming language.

Considerations for Choosing Between Visual Basic and Visual C# for an Application

Question: What are the possible issues that you might need to handle in a mixed-language environment?

Answer: In a mixed language environment, issues can include:

- Thousands of lines of code that might need to be written.
- Legacy code that might need to get ported, possibly in Visual C++ or classic Visual Basic.

- Java code that might need to be ported.

Additional Reading

Features of Visual Basic

For more information about implicit line continuation, see the Statements in Visual Basic page on the Microsoft Web site.

For more information about Visual Basic, see the Getting Started with Visual Basic page on the Microsoft Web site.

Features of Visual C#

For more information about Visual C# Samples, see the Visual C# Samples and Walkthroughs page on the Microsoft Web site.

Lesson 2

Overview of Visual Studio 2010

Contents:

Question and Answers

Advantages of Using Visual Studio 2010

Question: Which Web server is the default server when creating your Web applications in Visual Studio 2010?

Answer: The ASP.NET Development Server is the default Web server when creating your Web application in Visual Studio 2010.

Available Project Templates

Question: What are some of the additional project templates that are available in Visual Studio 2010?

Answer: The following table lists some of the additional project templates that are available in Visual Studio 2010.

Template group	Description
Other Project Types	Visual Studio 2010 includes templates for setup and deployment projects, database projects, extensibility projects, and blank solutions.
Test Projects	Visual Studio 2010 provides a template for a test project.

Features of the Integrated Development Environment

Question: What are the four views available for editors and designers?

Answer: The four views available for editors and designers are a graphical Design view, the Source (markup) view, the Split view, and the code-behind view.

Visual Studio 2010 Start Page

Question: Which area and tab of the Start Page displays a list of Help topics, Web sites, technical articles, and other resources?

Answer: The **Get Started** tab in the **Content** area displays a list of Help topics, Web sites, technical articles, and other resources.

Additional Reading

Features of the Integrated Development Environment

For more information about Integrated Development Environment, see the Quick Tour of the Integrated Development Environment page on the Microsoft Web site.

Lesson 3

Creating a Simple Web Application

Contents:

Question and Answers

Web Application Development Process

Question: What are the three main phases of the Web application development process?

Answer: The following are the three main phases of the Web application development process:

- Design phase. Create a design specification—or development blueprint—based on customer inputs.
- Develop and debug phase. A Web application or Web site creation is an iterative process that consists of building the UI, writing code that responds to user actions, and testing the application or Web site.
- Deploy phase. In this phase, you deploy your Web site to a production Web server.

Web Application Project Files and Folders

Question: What containers does Visual Studio provide?

Answer: Visual Studio provides two containers to help you efficiently manage the items that are required for your development effort. These containers are called solutions and projects. Visual Studio also provides Solution folders to organize related projects into groups, and then perform actions on those groups. Solution Explorer is an interface for viewing and managing these containers and their associated items, and is part of the IDE.

Web Application Files

Question: What are examples of files that are not based on a programming language, but will have their own extensions?

Answer: Examples of files that are not based on a programming language but have their own extensions are report files, which use the .rdlc extension, and text files, which use the .txt extension.

Demonstration: How to Create a Simple Web Application Project

Question: When you create a Web site, what folders or files does Solution Explorer display?

Answer: Solution Explorer displays a number of folders and files, including the Account folder, the App_Data Web folder, the Scripts folder, the Styles folder, the About.aspx Web Form and its associated code-behind file, the Default.aspx Web Form and its associated code-behind file, the Site.master master page, the Global.asax file, and the Web configuration file. Visual C# projects also contain a Properties and References folder, while the Visual Basic projects contain a My Project folder.

Detailed Demo Steps

Demonstration: How to Create a Simple Web Application Project

Demonstration steps

1. Open Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
2. Create a Web application project.
    a. In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **New Project**.
    b. In the **New Project** dialog box, in the left pane, click either **Visual Basic** or **Visual C#**.
    c. In the middle pane, click **ASP.NET Web Application**, and then click **OK**.
3. Add TextBox and Button controls to the Default Web Form.
    a. In the Default.aspx window, click **Design**.
    b. In Design view, click the text **You can also**, click **p**, press the RIGHT ARROW key, and then click ENTER.
    c. In the Toolbox, expand **Standard**, and then double-click the **Button** control.
    d. In the Toolbox, under **Standard**, double-click the **TextBox** control.
4. Add code for the Click event handler of the **Button** control.
    - In the Default.aspx window, double-click the **Button** control, and then add the following code to the **Click** event handler.

[Visual Basic]

```
TextBox1.Text = "You clicked the button"
```

[Visual C#]

```
TextBox1.Text = "You clicked the button";
```

5. Write code to catch PostBack in the **Page_Load** event handler.
    - Type the following code in the **Page_Load** event handler.

[Visual Basic]

```
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
If Me.IsPostBack Then
Response.Write("Server roundtrip due to postback")
Else
Response.Write("First time page is loaded")
End If
End Sub
```

[Visual C#]

```
protected void Page_Load(object sender, EventArgs e)
```

```
{
if (this.IsPostBack)
{
Response.Write("Server roundtrip due to postback");
}
else
{
Response.Write("First time page is loaded");
}
```

6. Build and debug the solution.

    a. Place the breakpoint by clicking the mouse at the beginning of the code where you check for the PostBack.

    b. In the WebApplication1 – Microsoft Visual Studio (Administrator) window, on the **Debug** menu, click **Start Debugging**.

> **Note:** If you see a **Debugging Not Enabled** message box, click **OK**.

    c. Run the application in the debug mode, and step through the lines of code by pressing the F10 key. When the browser opens after the first debugging session, click **Page** on the toolbar, and then click **View Source** to view the source in Internet Explorer.

    d. Click the **Button** to post back to the server.

    e. In the Default.aspx.vb or Default.aspx.cs code window, step through the lines of code by pressing F10.

    f. In the http://localhost:1186/Default.aspx-Original Source window, click the **Close** button.

    g. In the http://localhost:1186/Default.aspx window, click the **Close** button.

Additional Reading

Web Application Development Process

For more information about the application development process, see the Application Development in Visual Studio page on the Microsoft Web site.

Choosing Between a Web Site and a Web Application Project

For more information about Web application projects, see the Introduction to Web Application Projects page on the Microsoft Web site

Deploying a Web Application

For more information about IIS, see the Official IIS Web page on the Microsoft Web site.

For more information about using the Copy Web Site Tool, see the [How Do I:] Deploy a Web Site Using the Copy Web Site Tool video on the Microsoft Web site.

Module Reviews and Takeaways

Review questions and answers

1. How would you select a .NET-based programming language to create a new Web application project?

   You select a .NET-based language based on the developer's experience with existing programming languages.

2. What role does common language runtime play in running an ASP.NET page?

   The common language runtime component compiles the MSIL to native code, and then runs the native code on the server.

3. What is the role of the JIT compilation?

   The runtime uses a JIT compiler to compile the MSIL to native code.

4. List some of the languages that are currently supported by the .NET Framework.

   Languages that are currently supported by the .NET Framework include Visual Basic, Visual C#, Visual C++, and Visual F#.

5. Why would you create a component for a Web application?

   You can create a component for a Web application to share business logic code with other applications.

Real-world issues and scenarios

1. You want to create a WCF service. What is the easiest way to implement this?

   Use the WCF Service Application project template.

2. You want to create a reusable component that you can share with other projects. What is the easiest way to implement this?

   Use the Class Library project template.

Tools

Tool	Function	Where to find it
Editor window	Displays code for editing and a graphical interface for control placement.	Visual Studio IDE
Solution Explorer	Displays the hierarchy of project files, and enables you to move and modify files.	Visual Studio IDE
Properties window	Enables you to adjust the properties of documents, classes, and controls.	Visual Studio IDE
Task List	Enables you to track the status of tasks as you develop your application.	Visual Studio IDE
Output	Displays status messages for various features in IDE.	Visual Studio IDE

Toolbox	Enables you to use a drag-and-drop operation on the controls in your application.	Visual Studio IDE

Lab Review Questions and Answers

1. How did you create a Web site?

   Answer: In Visual Studio 2010, on the **File** menu, click **New Web Site**.

2. Can you think of a reason why an ASP.NET Web Form (.aspx) has an associated code-behind file?

   Answer: The associated code-behind file separates the programming logic from the Web page.

3. How do you show or hide a server control?

   Answer: You set the **Visible** property to **True** or **False**.

Module 3

Creating a Microsoft® ASP.NET Web Form

Contents:

Lesson 1

Creating Web Forms

Contents:

Question and Answers

What Is a Web Form?

Question: What are the two components that make up Web Forms?

Answer: The two components that make up Web Forms are the visual portion of the .aspx file, and the code behind the form, which resides in a separate class file.

How to Create a Web Form

Question: Which tool can you use to add additional Web Forms if you are expanding an existing project?

Answer: You can use Solution Explorer and the **Add New Item** dialog box to add additional Web Forms.

Lesson 2

Adding and Configuring Server Controls in a Web Form

Contents:

Question and Answers

What Is a Server Control?

Question: Which attribute makes a control a server control?

Answer: A server control has a **Runat** attribute with the value set to **Server**.

Types of Server Controls

Question: Which types of server controls exist for ASP.NET?

Answer: HTML server controls and Web server controls exist for ASP.NET.

Adding and Configuring HTML Server Controls

Question: What are the features offered by HTML server controls?

Answer: HTML server controls offer the following features:

- A server-based object model that you can program against by using familiar object-oriented techniques. Each server control exposes properties that help you programmatically manipulate the control's markup attributes in server code.
- A set of events for which you can write event handlers in the same way you would in a client-based form, except that the event is handled in server code.
- The ability to handle events in client script.
- HTML server controls that are based on HTML elements.
- HTML server controls that are logically grouped in the **System.Web.UI.HtmlControls** namespace.
- Automatic maintenance of the control's state. When the page makes a round trip between the client and the server, the values that the user enters into HTML server controls are automatically maintained and returned to the browser.
- Interaction with ASP.NET validation controls. This helps you verify that a user has entered appropriate information into a control.
- Data binding to one or more properties of the control.
- Support for styles, if the ASP.NET Web page is displayed in a browser that supports CSS.
- Pass-through of custom attributes. You can add any attributes you need to an HTML server control, and the page framework will render them without any change in functionality. This helps you add browser-specific attributes to your controls.

Adding and Configuring Web Server Controls

Question: What are the additional features offered by Web server controls?

Answer: Web server controls offer all the features described for HTML server controls (except one-to-one mapping to elements), and the following additional features:

- A rich object model that provides type-safe programming capabilities.
- Automatic browser detection. The Web server controls can detect browser capabilities and render appropriate markup.
- For some controls, the ability to define your own layout for the control by using templates.
- For some controls, the ability to specify whether a control's event causes immediate posting to the server, or is, instead, cached and raised when the page is submitted.
- Support for themes, which helps you define a consistent look for controls throughout your site.
- Ability to pass events from a nested control—such as a button in a table—to the container control.

Additional Reading

Types of Server Controls

For more information about developing custom controls, see Developing Custom ASP.NET Server Controls.

Module Reviews and Takeaways

Review questions and answers

1. How can you verify that a Web page with an .aspx extension in an ASP.NET Web site is a Web Form?

   In Visual Basic, look for the following code.

   ```
   <%@ Page Language="VB" CodeFile="PageName.aspx.vb" Inherits="PageName"%>
   ```

   In Visual C#, look for the following code.

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

   Also look for a **<form runat="server">** tag.

2. How can you verify that a Web page with an .aspx extension contains intrinsic ASP.NET Web server controls?

   Look for the following code, where ControlType specifies the actual type of control, such as Button or TextBox.

   ```
   <asp:ControlType ...></asp:ControlType>
   ```

   Also look for the **runat="server" attribute.**

3. What type of markup or script does a Web server control generate on the client?

   A Web server control generates **HTML and** JavaScript, if the Web server control is a complex control. The latter is true if the Web server control uses client-side input validation, or updates other controls in response to a user action performed on the Web server control.

Real-world issues and scenarios

1. You want to add a page to your Web application that you can work with by using server-side code. What is the easiest way to implement this?

   You add a new item by using the Web Form item template.

2. You want to add a title to a Web Form. What is the easiest way to implement this?

   You specify a value for the **title** attribute of the **Page** directive.

Best practices

- Web Forms should expose a page title that displays as the browser caption, and on the tab provided your browser session is tabbed. This makes it easier for users to identify a specific page.
- Always use the appropriate Web server control for the task—such as using a **Label** control instead of a **TextBox** control—to display read-only text.
- Follow a naming convention when naming your HTML elements and Web server controls. In this course, a naming convention consisting of what the control or element contains, and it is suffixed by the control type that is used, such as **FirstNameTextBox** for a **TextBox** control that exposes the first name of a person. This way, you can also have a **FirstNameLabel** Label control that indicates to the user what the **TextBox** control contains. In general, all HTML elements and Web

server controls should be given a name or id, because you might need to refer to the controls from other controls, or from either client-side or server-side code.

Lab Review Questions and Answers

1. What are the advantages of Web Form controls?

   Answer: While ASP.NET Web Form controls provides a similar appearance to the conventional HTML controls, ASP.NET Web Form controls provide a consistent and structured interface and more robust features. In addition, you can use client-side scripting to enhance the functionality of these controls.

2. What type of Web server controls are used in the lab?

   Answer: Intrinsic Web server controls are used in the lab.

Module 4

Adding Functionality to a Microsoft® ASP.NET Web Form

Contents:

Lesson 1

Working with Code-Behind Files

Contents:

Question and Answers

Methods for Implementing Code

Question: What are the two main reasons that influenced the decision to make code-behind files the default method for creating Web Forms in Visual Studio 2010?

Answer: Code-behind files provide readability and maintainability, and improved performance when creating Web forms in Visual Studio 2010:

In addition, code should be placed in a class of its own, even if it is in a component. If you need to write code that is indirectly related to your Web Form—for example, information about a customer—you can place this code in a class of its own.

Mixed Code and Inline Code

Question: How can you use the same method in Visual Basic as in Visual C#, though the **OnClick** event is wired up differently for Visual Basic and Visual C#?

Answer: Although the **OnClick** event is wired up differently for Visual Basic and Visual C#, you can use the same method in Visual Basic as in Visual C# if you specify the **OnClick** attribute in the markup. The special **Handles** keyword in Visual Basic gives Visual Basic developers an extra option.

Lesson 2

Handling Server Control Events

Contents:

Question and Answers

What Are Event Handlers?

Question: How are types of events classified for ASP.NET?

Answer: Events are classified for ASP.NET as follows:

- **Application Events**. These are triggered in response to events that occur at the application level, such as application errors, application start, and application end.
- **Page events**. These are triggered in response to events that occur at the page level, such as page loading and unloading.
- **Control events**. These are triggered in response to events that occur at the control level, such as a button click or a link hover.

All event handlers in the Microsoft .NET Framework have a signature that contains two parameters: *sender*, and *e*. The *sender* parameter is always of type **System.Object**, and it generally contains a reference to the control that triggered the event. The *e* parameter is of type **System.EventArgs**, or a type derived from it. The *e* parameter contains the data for the event, and can be used for various purposes, such as canceling an event. The exact functions of *e* depend on the event or how it was triggered.

Detailed Demo Steps

Demonstration: How to Create Server-Side Event Handlers

Demonstration steps

1. Log on to 10267A-GEN-DEV as **Student**, with the password, **Pa$$w0rd**.
    - To log on 10267A-GEN-DEV, use the following user credentials:
        - User name: **Student**
        - Password: **Pa$$w0rd**
2. Create a Web Site by using Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **New Web Site**.
    - In the **New Web Site** dialog box, in the left pane, click either **Visual Basic** or **Visual C#**, in the middle pane, ensure that **ASP.NET Empty Web Site** is selected, and then click **OK**.
3. Add a new Web Form named Default.aspx to the Web site.
    - In Solution Explorer, right-click the Web site, and then click **Add New Item**.
    - In the **Add New Item** dialog box, in the middle pane, ensure that **Web Form** is selected, in the **Name** box, type **Default.aspx**, and then click **Add**.
4. Add the Button and Label controls to the Web Form
    - In the Default.aspx window, click **Design**, and then point to **Toolbox**.
    - In Toolbox, expand **Standard**, and then double-click the **Button** control.
    - In Toolbox, under **Standard**, double-click the **Label** control.
5. View the default **Click** event handler code.
    - In the Default.aspx window, double-click the **Button** control to open the code-behind file.
    - In the Default.aspx.vb or Default.aspx.cs window, view the empty event handler created by Visual Studio 2010.

[Visual Basic]

```
Protected Sub Button1_Click(ByVal sender As Object,
ByVal e As System.EventArgs) Handles Button1.Click
...
End Sub
```

[Visual C#]

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

```
}
```

6. Add the following code in the **Click** event handler of the Button control.

[Visual Basic]

```
Label1.Text = "You clicked the button"
```

[Visual C#]

```
Label1.Text = "You clicked the button";
```

7. View the event handler wiring up.

    - In the Visual Basic programming language, notice the **Handles** statement in the **Button1_Click** event handler.

[Visual Basic]

```
Protected Sub Button1_Click(ByVal sender As Object,
ByVal e As System.EventArgs) Handles Button1.Click
```

    - In the Visual C# programming language, the procedure is bound to the event handler by adding the attribute **onclick** to the markup. You can view the following code from the Source view of the Default.aspx Web Form.

[Visual C#]

```
...
<asp:Button ID="Button1" runat="server" Text="Button" onclick="Button1_Click">
...
```

8. Save the changes, and view the Web Form in the browser.

    - Press the CTRL+SHIFT+S keys.
    - In Solution Explorer, right-click the **Default.aspx** Web Form, and then click **View in Browser**.

> **Note:** Open the rendered source by clicking **View Source** on the **Page** menu, and notice that the event handler code is not rendered to the client.

9. Trigger the event handler by using the **Button** control

    - In the http://localhost:49157/WebSite1/Default.aspx - Internet Explorer window, click the **Button** control on the Web Form. Notice that the text of the label changes.
    - In the http://localhost:49157/WebSite1/Default.aspx - Internet Explorer window, click the **Close** button.
    - In Visual Studio 2010, click the Default.aspx window, and then click **Design**.
    - In the Default.aspx window, click the **Button** control.
    - In the Properties window of the Button control, change the **ID** property of the Button control to **SubmitButton**.

- In the Default.aspx window, double-click the **Button** control to open the code-behind file. Notice that **Button1** has been changed to **SubmitButton** in the code-behind file.

Note: In the Visual Basic programming language, in the code-behind file, notice that the **Handles** keyword was updated to reflect the new object name, **SubmitButton.Click**. This maintains the binding to the correct event handler. The event handler name has not been changed, and remains as **Button1_Click**. In the Visual C# programming language, in the code-behind file, notice that the new button name has been changed in the markup. The **onclick** attribute remains set to **Button1_Click**. Visual Studio does not change the event binding to the procedure, but only the **ID** property of the control.

10. Save and view the Web Form in the browser
    - On the **File** menu, click either **Save Default.aspx.vb** or **Save Default.aspx.cs**.
    - In Solution Explorer, right-click the Default.aspx Web Form, and then click **View in Browser**.
11. Trigger the event handler by using the Button control
    - In the http://localhost:49157/WebSite1/Default.aspx - Internet Explorer window, click the **Button** control on the Web Form.

Note: Notice that the **Click** event handler still triggers when you use the new name of the control.

- In the http://localhost:49157/WebSite1/Default.aspx - Internet Explorer window, click the **Close** button.
- In the WebSite1 - Microsoft Visual Studio window, click the **Close** button.

Additional Reading

What Are Event Handlers?

For more information about the life cycle of a Web Form, including information on scenarios when both page events and control events are triggered, see ASP.NET Page Life Cycle Overview.

Lesson 3

Creating Classes and Components by Using Visual Studio 2010

Contents:

Question and Answers

What Are Types, Components, and Classes?

Question: What are the differences between a structure, class, property, method, object, and component?

Answer: The following are the differences between a structure, class, property, method, object, and component.

- *Structure*. A user-defined value type. Like a class, structures can contain constructors, constants, fields, methods, properties, indexers, operators, and nested types. However, unlike classes, structures do not support inheritance.
- *Class*. A reference type that encapsulates data such as constants and fields, and behavior such as methods, properties, indexers, events, operators, instance constructors, static constructors, and destructors, and can contain nested types. Class types support inheritance, a mechanism whereby a derived class can extend a base class.
- *Property*. A class member in the .NET Framework that is like a public field, but includes features such as versioning, encapsulation, and the ability to run additional logic through **get** and **set** accessor methods.
- *Method*. A function that describes the behavior of a class. Including a method in a class does not guarantee an implementation of the method.
- *Component*. A container for one or more types that can be logically grouped together in a single unit of deployment.
- *Object*. A structure that contains data and methods that manipulate the data. A class is an abstract representation of an object.

Adding Member Variables and Constants to a Class

Question: What are member variables and constants?

Answer: A local variable is one that is declared within a procedure. A member variable is a member of a .NET Framework type; it is declared at module level inside a class, structure, or module, but not within any procedure internal to that class, structure, or module.

Adding Properties and Methods to a Class

Question: Discuss the differences between properties and methods.

Answer: The following are some of the differences between methods and properties.

Methods	Properties
Can take arguments.	Generally do not take any arguments, only the value for setting a property.
Are read-only.	Return a value. However, some properties are

	read-only, while others are read/write.
Can be a function that performs, regardless of whether the object's state is involved.	Usually represent some aspect of an object's state.

Detailed Demo Steps

Demonstration: How to Create a Class in Visual Studio 2010

Demonstration steps

1. Create a Class Library project named **HelloWorld** in the Visual Studio 2010 solution.
    a. On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
    b. On the File menu of Visual Studio 2010, click **New Web Site**.
    c. In the **New Web Site** box, in the left pane click either **Visual Basic** or **Visual C#**, in the middle pane, ensure that **ASP.NET Empty Web Site** is selected, in the **Web location** list, click **File System**, in the text box, type **C:\WebSite1**, and then click **OK**.
    d. In Solution Explorer, right-click **Solution 'WebSite1' (1 Project)**, point to **Add**, and then click **New Project**.
    e. In the **Add New Project** dialog box, in the left pane, click either **Visual Basic** or **Visual C#**, in the middle pane, click **Class Library**, in the **Name** box, type **HelloWorld**, and then click **OK**.

> **Note:** Notice that a default class (either **Class1.vb or Class1.cs)**, is created in Solution Explorer and opens in the Code editor window.

2. Rename the **Class1.vb or Class1.cs** file as **Hello.vb or Hello.cs**.
    a. In Solution Explorer, right-click either **Class1.vb or Class1.cs**, click **Rename**, change the text to **Hello.vb or Hello.cs**, and then press ENTER.
    b. In the Microsoft Visual Studio message box, click **Yes**.
3. Create a method that returns a string on the Class file.
    - Create a **SayHello** method that returns a string by using the following code.

[Visual Basic]

```
Function SayHello() As String
       Return "Hi from Visual Basic component."
End Function
```

[Visual C#]

```
public string SayHello()
{
       return "Hi from C# component.";
}
```

4. Build the Class Library project.
    - On the **Build** menu of Visual Studio 2010, click **Build Solution**.
5. Add a reference to the **HelloWorld** Class Library project.
    a. In Solution Explorer, right-click **C:\...\WebSite1**, add then click **Add Reference**.

b. On the **Projects** tab of the **Add Reference** dialog box, under **ProjectName**, ensure that **HelloWorld** is selected, and then click **OK**.

c. In Solution Explorer, in the **Bin** folder, verify that the **HelloWorld.dll** assembly has been copied.

d. In the WebSite1 - Microsoft Visual Studio window, click the **Close** button.

e. In the Microsoft Visual Studio message box, click **Yes**.

In addition to adding a class manually, Visual Studio 2010 has a Class Diagram that can be used to create classes and class hierarchies. Although the Class Diagram is beyond the scope of this course, you can view it by right-clicking a project in Solution Explorer, and then clicking **View Class Diagram**. After the Class Diagram opens, you can access Help by pressing the F1 key.

Working with layers

Class libraries are often used to create components, which are used to create the functionality that are used in the different layers of a distributed application. Layers are a way of logically grouping functionality, such as grouping all of the business logic into a business logic layer (BLL) and all of the data access functionality into a data access layer (DAL). Grouping functionality makes it easier to manage and distribute the functionality in different tiers, where layers are the logical grouping of functionality, and tiers are the physical separation of functionality onto different computers.

Additional Reading

What Are Types, Components, and Classes?

For more information about structures and classes, see Choosing Between Classes and Structures on the Microsoft Web page.

Adding Class Constructors

For more information about constructors, see Using Constructors (C# Programming Guide).

For more information about Code Snippets, see Creating and Using IntelliSense Code Snippets.

Lesson 4

Handling Page Events

Contents:

Question and Answers

Page Event Life Cycle

Question: What are the page life cycle events that are used most frequently?

Answer: The following table lists the page life cycle events that are used most frequently. There are more events than those listed. However, they are not used for most page processing scenarios. Instead, they are primarily used by server controls on the ASP.NET Web page to initialize and render themselves.

Page event	Typical use
PreInit	Use this event for the following: • Check the **IsPostBack** property to determine whether this is the first time the page is being processed. • Create or re-create dynamic controls. • Read or set profile property values. **Note:** If the request is a postback, the values of the controls have not yet been restored from View state. If you set a control property at this stage, its value might be overwritten in the next event.
Init	This event is raised after all controls have been initialized and any skin settings have been applied. Use this event to read or initialize control properties.
InitComplete	This event is raised by the **Page** object. Use this event for processing tasks that require all initialization be complete.
PreLoad	Use this event if you need to perform processing on your page or control before the Load event. You often check the **IsPostBack** property to determine whether this is the first time the page is being processed. Before the **Page** instance raises this event, it loads the View state for itself and all controls, and then processes any postback data included with the **Request** instance.
Load	The Page calls the **OnLoad** event method on the **Page**, and then recursively does the same for each child control, which does the same for each of its child controls, until the page and all controls are loaded. Use the **OnLoad** event method to set properties in controls and establish database connections.
Control events	Use these events to handle specific control events, such as a Button control's Click event, or a TextBox control's **TextChanged** event. Note: In a postback request, if the page contains validator controls, check the **IsValid** property of both the Page and of individual validation controls before performing any processing.
LoadComplete	Use this event for tasks that require that all other controls on the page be

	loaded.
PreRender	Before this event occurs: • The **Page** object calls **EnsureChildControls** for each control and for the page. • Each data-bound control whose **DataSourceID** property is set, calls its **DataBind** method. The **PreRender** event occurs for each control on the page. Use this event to make final changes to the contents of the page or its controls.
SaveStateComplete	Before this event occurs, **ViewState** has been saved for the page and for all controls. Any changes to the page or controls at this point will be ignored. Use this event to perform tasks that require the View state to be saved, but that do not make any changes to controls.
Render	This is not an event; instead, at this stage of processing, the **Page** object calls this method on each control. All ASP.NET Web server controls have a **Render** method that writes out the control's markup that is sent to the browser. If you create a custom control, you typically override this method to output the control's markup. However, if your custom control incorporates only standard ASP.NET Web server controls and no custom markup, you do not need to override the **Render** method. A user control (an .ascx file) automatically incorporates rendering, so you do not need to explicitly render the control in code.
Unload	This event occurs for each control, and then for the page. In controls, use this event to do a final cleanup for specific controls, such as closing the control-specific database connections. For the page itself, use this event to do final cleanup work, such as closing open files and database connections, or completing the logging or other request-specific tasks. Note: During the unload stage, the page and its controls have been rendered, so you cannot make further changes to the response stream. If you attempt to call a method—such as the **Response.Write** method—the page will throw an exception.

The PostBack Process

Question: Which property can be used in all events, and not just the **Load** event?

Answer: The **IsPostBack** property can be used in all the events, and not just the **Load** event.

Demonstration: How to Handle Page Events

Question: Describe the use of the Events button in the Properties window.

Answer: You can use the **Events** button of the Properties window to write code for any of the events for a form or for the controls. You must first select the object on the form, and then select the event for that object in the Event view of the Properties window.

Detailed Demo Steps

Demonstration: How to Handle Page Events

Demonstration steps

To view the simple code and event handler

1. Open the **EventOrder** solution from the **D:\Demofiles\M4\VB** or **D:\Demofiles\M4\CS** folder.
2. Run the Web application.
3. Switch to Visual Studio 2010, and view the simple code and the **Page_Unload** event handler.
4. Switch to Internet Explorer, and check the Postback event.

In this next demonstration, you will be shown the order of the page events, including the event handlers and the page output. You can perform these tasks by using either Visual Basic or Visual C#.

It is important to understand the order in which events are triggered, and when you need to distinguish between a postback and a non-postback of a page.

> **Note:** The EventOrder.sln solution file in the D:\Demofiles\M4\CS or D:\Demofiles\M4\VB folder contains a complete solution for this demonstration.

To view the order of page events

1. Open the **EventOrder** solution from the D:\Demofiles\M4\VB or D:\Demofiles\M4\CS folder.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
    - On the **File** menu of Visual Studio 2010, click **Open Project**.
    - In the **Open Project** dialog box, in the **File name**, type either **D:\Demofiles\M4\CS\EventOrder.sln** or **D:\Demofiles\M4\VB\EventOrder.sln**, and then click Open.
2. Run the Web application.
    - On the **Debug** menu of Visual Studio 2010, click **Start Without Debugging**.

> **Note:** In the http://localhost:49241/EventOrder/ - Internet Explorer window, notice that all the events have output to the label on the left.

3. Switch to Visual Studio 2010, and view the simple code and the **Page_Unload** event handler.
    - Switch to the Visual Studio 2010 window by clicking the Visual Studio taskbar button in the taskbar.
    - In Solution Explorer, expand **Default.aspx**, right-click either **Default.aspx.vb** or **Default.aspx.cs**, and then click **Open**.
    - In the Default.aspx.vb or Default.aspx.cs window, show the simple code and the **Page_Unload** event handler, which also outputs to the label, but is not shown on the rendered page because the output rendering is completed at this stage.

[Visual Basic]

```
Protected Sub Page_Unload(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Unload
        If Not Me.IsPostBack Then
              NonPostBackEventLabel.Text &= "Unload event<br />"
        Else
              PostBackEventLabel.Text &= "Unload event<br />"
        End If
End Sub
```

[Visual C#]

```
protected void Page_Unload(object sender, EventArgs e)
{
        if (!this.IsPostBack)
              NonPostBackEventLabel.Text += "Unload event";
        else
              PostBackEventLabel.Text += "Unload event";
}
```

4. Switch to Internet Explorer, and check the **Postback** event.
    - Click the Internet Explorer taskbar button in the taskbar.
    - In the http://localhost:49241/EventOrder/ - Internet Explorer window, click the **Postback** button.
    - In the EventOrder - Microsoft Visual Studio window, click the **Close** button.
    - In the http://localhost:49241/EventOrder/default.aspx - Internet Explorer window, click the **Close** button.

Additional Reading

Page Event Life Cycle

For more information about page life cycle and page events, see ASP.NET Page Lifecycle Overview.

Handling Postbacks

For more information about cross-page postbacks, see Cross-Page Posting in ASP.NET Web Pages.

Module Reviews and Takeaways

Review questions and answers

1. What is the advantage of code-behind files when you add functionality to a Web Form?

   Code-behind files allow separation of code from content, and allow the code developer to work on the code-behind file while the UI designer works on the .aspx file.

2. How is an event procedure associated with the event of a server control?

   In Visual Basic, the **Handles** keyword references the **ID** of the control and the name of the event. In Visual C#, an event attribute is added to the markup of a control, and an event procedure template is created in the code-behind file.

3. How can you use a component in your Visual Studio 2010 project?

   To use a component, you must add a reference to it from your Visual Studio 2010 project.

Real-world issues and scenarios

1. You want to be able to have a designer work on a Web Form simultaneously while a developer works on the code for the Web Form. What is the easiest way to implement this?

   Use a Web Form with a code-behind file.

2. You want to add page events to a code-behind file, but you do not want to manually have to wire up the event handlers. What is the easiest way to implement this?

   Specify a value of **true** for the **AutoEventWireUp** attribute of the **Page** directive.

Best practices

- Follow a naming and casing convention when naming your variables. In this course, a naming convention consisting of what the variable contains, and is suffixed by the data type when appropriate, is used. If a variable contains one of the simple data types—such as Integer or String—it is most often not necessary to suffix with the data type. Examples are **FirstName, LastName, Name, Age**, and **Length**.
- If you are naming a variable of a specific object type, always use the object type name as the suffix, such as **CustomerManagementDataSet**, where the data type is DataSet, and the name is CustomerManagement. The casing differs between Pascal Casing and Camel Casing, where the former capitalizes the first character of each word—including acronyms over two letters in length, such as "FirstName". The latter is close to Pascal Casing, but always uses an initial lower-case letter, such as "firstName". Pascal Casing is generally used for public variables, whereas Camel Casing is used for local and private variables, as well as method parameters. See the next best practice regarding public variables.
- Member variables—also known as backing fields—as a general rule should never be made public, because that can corrupt the state of an object. Instead, make a member variable private, and expose the value through a property with which you can check when a user of the object sets the value of the property. There are exceptions to this rule, such as some instances of when an object needs to be serialized, but that is beyond the scope of this Module.
- Follow a naming and casing convention when naming your methods. In this course, the naming convention used describes the data it makes available—such as FirstName—but never with the first word of the name being a verb—such as used for method names (GetFirstName). Casing

uses the same rules as for variables—public methods use Pascal Casing, whereas non-public methods use Camel Casing.

- Follow a naming and casing convention when naming your properties. In this course, a naming convention consisting of what the method does, but always with the first word of the name being a verb, such a GetUserID or SaveID. Casing uses the same rules as for variables and methods—public properties use Pascal Casing, whereas non-public properties use Camel Casing.

Lab Review Questions and Answers

1. How can you run code only, when a Web page loads for the first time?

   Answer: Add a **Page.IsPostback** test to **the Page_Load** page event procedure.

2. What is the default event procedure for common controls?

   Answer: TextBox: **TextChanged**, Button: **Click**, DropDownList: **SelectedIndexChanged**.

3. How can you add items to a list in Design view?

   Answer: You can add and remove items in Design view by setting the **Items** property of a list. You can also set the properties of each list item.

Module 5

Implementing Master Pages and User Controls

Contents:

Lesson 1

Creating Master Pages

Contents:

Question and Answers

What Are Master Pages?

Question: Which features differentiate a master page from a standard Web Form?

Answer: With Web Forms, you can place one or more ContentPlaceHolder controls that define the areas where the replaceable content will be displayed. Web Forms also have a **Master** directive instead of the **Page** directive.

Creating a Master Page

Question: What is the main purpose of creating master pages?

Answer: The main purpose of creating master pages is to simplify the process of creating a Web site that has a consistent layout.

Creating a Content Page

Question: When you design a Web application by using master pages, what are some of the external resources that you might need to reference and add?

Answer: You might need to add image controls that reference image files, or anchors that reference other pages.

What Are Nested Master Pages?

Question: What is the main purpose of using nested master pages?

Answer: Nested master pages create child master pages for partners, departments, or groups on a Web site.

Runtime Behavior of Master Pages

Question: How is the master page merged with the content page at run time?

Answer: The master page content is merged into the control tree of the content page.

Additional Reading

Adding a Master Page to an Existing Web Application Project

For more information about working with ASP.NET Master Pages, see Working with ASP.NET Master Pages Programmatically.

For more information about sharing master pages in Visual Studio, see Sharing Master Pages in Visual Studio.

Lesson 2

Adding User Controls to an ASP.NET Web Form

Contents:

Question and Answers

Adding a User Control to a Web Form

Question: What is the difference between the **TagPrefix**, **TagName**, and **Src** attributes?

Answer: The following are the differences between the attributes:

- **src**. The location (relative or absolute) of the declarative ASP.NET User Controls file to associate with the **tagprefix:tagname** pair.
- **TagName**. An arbitrary alias to associate with a class. This attribute is only used for user controls.
- **TagPrefix**. An arbitrary alias that provides a shorthand reference to the namespace of the markup being used in the file that contains the directive.

Detailed Demo Steps

Demonstration: How to Convert a Web Form into a User Control

Demonstration steps

1. Log on to 10267A-GEN-DEV as **Student**, with the password, **Pa$$w0rd**.
2. Open the **CustomerManagement** solution from either the D:\Demofiles\M5\VB or D:\Demofiles\M5\CS folder.
3. On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
4. On the **File** menu of Visual Studio 2010, click **Open Project**.
5. In the **Open Project** dialog box, in the **File name** box, type either **D:\Demofiles\M5\VB\CustomerManagement.sln** or **D:\Demofiles\M5\CS\CustomerManagement.sln**, and then click **Open**.
6. Open the **InsertCustomer.aspx** Web Form, and change its **Page** directive to a **Control** directive.

[Visual Basic]

```
<%@ Control Language="VB" AutoEventWireup="false"
CodeFile="InsertCustomer.aspx.vb" Inherits="InsertCustomer" %>
```

[Visual C#]

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

   a. In Solution Explorer, right-click **InsertCustomer.aspx**, and then click **Open**.
   b. In the InsertCustomer.aspx window, locate the **Page** directive, and then change it to a **Control** directive.

[Visual Basic]

```
<%@ Control Language="VB" AutoEventWireup="false"
CodeFile="InsertCustomer.aspx.vb" Inherits="InsertCustomer" %>
```

[Visual C#]

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

7. Add a **ClassName** property with a value of **InsertCustomer** to the **Control** directive.

[Visual Basic]

```
<%@ Control Language="VB" AutoEventWireup="false"
CodeFile="InsertCustomer.aspx.vb" Inherits="InsertCustomer"
ClassName="InsertCustomer" %>
```

[Visual C#]

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="InsertCustomer.aspx.cs"
Inherits="InsertCustomer" ClassName="InsertCustomer" %>
```

- In the InsertCustomer.aspx window, add the following code at the end of the **Control** directive.

```
ClassName="InsertCustomer"
```

8. In the InsertCustomer.aspx window, remove all the top-level HTML elements, such as the **DOCTYPE, html, head, body, title, link**, and **form** elements.

Note: You should take care not to delete the **div** element and the content within the **form** element. Both the opening and closing tags for each element—if they exist—must be removed.

Note: After removing all the top-level elements, you will get the following code.

[Visual Basic]

```
<%@ Control Language="VB" AutoEventWireup="false"
CodeFile="InsertCustomer.aspx.vb" Inherits="InsertCustomer"
ClassName="InsertCustomer" %>
       <div class="customertable">
             <div class="customertablerow">
                   <div class="customertableleftcol">
                         <asp:Label ID="CustomerNameLabel" runat="server"
Text="Name:"></asp:Label>
       </div>
       <div class="customertablerightcol">
             <asp:TextBox ID="CustomerNameTextBox" runat="server"
MaxLength="50"></asp:TextBox>
       </div>
       </div>
       <div class="customertablerow">
             <div class="customertableleftcol">
                   <asp:Label ID="CustomerAddressLabel" runat="server"
Text="Address:"></asp:Label>
       </div>
       <div class="customertablerightcol">
             <asp:TextBox ID="CustomerAddressTextBox" runat="server"
MaxLength="50"></asp:TextBox>
       </div>
       </div>
       <div class="customertablerow">
       <div class="customertableleftcol">
             <asp:Label ID="CustomerPhoneLabel" runat="server"
Text="Phone:"></asp:Label>
       </div>
       <div class="customertablerightcol">
<asp:TextBox ID="CustomerPhoneTextBox" runat="server"
MaxLength="30"></asp:TextBox>
</div>
</div>
<div class="customertablerow">
<div class="customertableleftcol">
             <asp:Label ID="CustomerZipCodeLabel" runat="server" Text="Zip
Code:"></asp:Label>
```

```
        </div>
        <div class="customertablerightcol">
                    <asp:TextBox ID="CustomerZipCodeTextBox" runat="server"
MaxLength="10"></asp:TextBox>
              </div>
        </div>
              <div class="customertablerow">
              <div class="customertableleftcol">
                    <asp:Label ID="CustomerCityLabel" runat="server"
Text="City:"></asp:Label>
        </div>
        <div class="customertablerightcol">
                    <asp:TextBox ID="CustomerCityTextBox" runat="server"
MaxLength="30"></asp:TextBox>
              </div>
        </div>
        <div class="customertablerow">
        <div class="customertableleftcol">
                    <asp:Label ID="CustomerStateLabel" runat="server"
Text="State:"></asp:Label>
        </div>
        <div class="customertablerightcol">
                    <asp:TextBox ID="CustomerStateTextBox" runat="server"
MaxLength="30"></asp:TextBox>
              </div>
        </div>
              <div class="customertablerow">
              <div class="customertableleftcol">
                    <asp:Label ID="CustomerEmailAddressLabel" runat="server"
Text="Email Address:"></asp:Label>
              </div>
              <div class="customertablerightcol">
                    <asp:TextBox ID="CustomerEmailAddressTextBox" runat="server"
MaxLength="40"></asp:TextBox>
              </div>
              </div>
              <div class="customertablefooter">
              </div>
              <asp:Button ID="CustomerInsertButton" runat="server" Text="Insert" />
               <asp:Button ID="CustomerCancelButton" runat="server"
Text="Cancel" />
        </div>
```

[Visual C#]

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="InsertCustomer.aspx.cs"
Inherits="InsertCustomer" ClassName="InsertCustomer" %>
        <div class="customertable">
              <div class="customertablerow">
                    <div class="customertableleftcol">
                                <asp:Label ID="CustomerNameLabel" runat="server"
Text="Name:"></asp:Label>
                    </div>
                    <div class="customertablerightcol">
                          <asp:TextBox ID="CustomerNameTextBox" runat="server"
MaxLength="50"></asp:TextBox>
                    </div>
              </div>
              <div class="customertablerow">
                    <div class="customertableleftcol">
                          <asp:Label ID="CustomerAddressLabel" runat="server"
Text="Address:"></asp:Label>
                    </div>
```

```
                  <div class="customertablerightcol">
                        <asp:TextBox ID="CustomerAddressTextBox" runat="server"
MaxLength="50"></asp:TextBox>
                  </div>
            </div>
            <div class="customertablerow">
                  <div class="customertableleftcol">
                        <asp:Label ID="CustomerPhoneLabel" runat="server"
Text="Phone:"></asp:Label>
                  </div>
                  <div class="customertablerightcol">
                        <asp:TextBox ID="CustomerPhoneTextBox" runat="server"
MaxLength="30"></asp:TextBox>
                  </div>
            </div>
            <div class="customertablerow">
                  <div class="customertableleftcol">
                        <asp:Label ID="CustomerZipCodeLabel" runat="server"
Text="Zip Code:"></asp:Label>
                  </div>
                  <div class="customertablerightcol">
                        <asp:TextBox ID="CustomerZipCodeTextBox" runat="server"
MaxLength="10"></asp:TextBox>
                  </div>
            </div>
            <div class="customertablerow">
                  <div class="customertableleftcol">
                        <asp:Label ID="CustomerCityLabel" runat="server"
Text="City:"></asp:Label>
                  </div>
                  <div class="customertablerightcol">
                        <asp:TextBox ID="CustomerCityTextBox" runat="server"
MaxLength="30"></asp:TextBox>
                  </div>
            </div>
            <div class="customertablerow">
                  <div class="customertableleftcol">
                        <asp:Label ID="CustomerStateLabel" runat="server"
Text="State:"></asp:Label>
                  </div>
                  <div class="customertablerightcol">
                        <asp:TextBox ID="CustomerStateTextBox" runat="server"
MaxLength="30"></asp:TextBox>
                  </div>
            </div>
            <div class="customertablerow">
                  <div class="customertableleftcol">
                        <asp:Label ID="CustomerEmailAddressLabel" runat="server"
Text="Email Address:"></asp:Label>
                  </div>
                  <div class="customertablerightcol">
                        <asp:TextBox ID="CustomerEmailAddressTextBox"
runat="server" MaxLength="40"></asp:TextBox>
                  </div>
            </div>
            <div class="customertablefooter">
            </div>
            <asp:Button ID="CustomerInsertButton" runat="server" Text="Insert"
onclick="CustomerInsertButton_Click" />
             <asp:Button ID="customerCancelButton" runat="server"
Text="Cancel"
onclick="CustomerCancelButton_Click" />
       </div>
```

9. Save the **InsertCustomer.aspx** Web Form.
   - In the CustomerManagement – Microsoft Visual Studio (Administrator) window, on the **File** menu, click **Save InsertCustomer.aspx**.
10. Rename the Web Form from **InsertCustomer.aspx** to **InsertCustomer.ascx**.
    a. In Solution Explorer, right-click **InsertCustomer.aspx**, click **Rename**, change the file name to **InsertCustomer.ascx**, and then press ENTER.
    b. In the Microsoft Visual Studio message box, click **Yes**.
11. Open either the **InsertCustomer.ascx.vb** or **InsertCustomer.ascx.cs** user control code-behind file, and change its base class from **System.Web.UI.Page** to **System.Web.UI.UserControl.**

[Visual Basic]

```
Partial Class InsertCustomer
       Inherits System.Web.UI.UserControl
```

[Visual C#]

```
public partial class InsertCustomer : System.Web.UI.UserControl
```

    a. In Solution Explorer, right-click either **InsertCustomer.ascx.vb** or **InsertCustomer.ascx.cs**, and then click **Open**.
    b. In the InsertCustomer.ascx.vb or InsertCustomer.ascx.cs code window, locate the **InsertCustomer** class, and change its property from **System.Web.UI.Page** to **System.Web.UI.UserControl**.

[Visual Basic]

```
Partial Class InsertCustomer
       Inherits System.Web.UI.UserControl
```

[Visual C#]

```
public partial class InsertCustomer : System.Web.UI.UserControl
```

12. Move the content from the **Page_LoadComplete** event method, and append it to the **Page_Load** event method.

[Visual Basic]

```
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
       ' Instantiate Customer
       instantiateCustomerObject()
       ' Populate the UI controls
       populateUI()
End Sub
```

[Visual C#]

```
protected void Page_Load(object sender, EventArgs e)
{
       // Instantiate Customer
       instantiateCustomerObject();
```

```
        // Populate the UI controls
        populateUI();
}
```

- In the InsertCustomer.ascx.vb or InsertCustomer.ascx.cs code window, in the **Page_LoadComplete** event method, select and right-click the following code, and then click **Cut**.

[Visual Basic]

```
' Populate the UI controls
populateUI()
```

[Visual C#]

```
// Populate the UI controls
populateUI();
```

- In the InsertCustomer.ascx.vb or InsertCustomer.ascx.cs code window, append the copied code to the **Page_Load** event method.

[Visual Basic]

```
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
        ' Instantiate Customer
        instantiateCustomerObject()
        ' Populate the UI controls
        populateUI()
End Sub
```

[Visual C#]

```
protected void Page_Load(object sender, EventArgs e)
{
        // Instantiate Customer
        instantiateCustomerObject();
        // Populate the UI controls
        populateUI();
}
```

13. In the InsertCustomer.ascx.vb or InsertCustomer.ascx.cs code window, delete the **Page_LoadComplete** event method.
14. Build the solution.
    - In the CustomerManagement – Microsoft Visual Studio (Administrator) window, on the **Build** menu, click **Build Solution**.
15. Save and close the user control code-behind file.
    a. In the CustomerManagement – Microsoft Visual Studio (Administrator) window, on the **File** menu, click **Save InsertCustomer.ascx.vb** or **Save InsertCustomer.ascx.cs.**
    b. In the InsertCustomer.ascx.vb or InsertCustomer.ascx.cs code window, click the **Close** button.
    c. In the InsertCustomer.ascx window, click the **Close** button.

d. In the CustomerManagement – Microsoft Visual Studio (Administrator) window click **Close** button.

Additional Reading

What Are User Controls?

For more information about the life cycle of a Web Form, including information about what happens to a Web Form when both page events and user control events are triggered, see ASP.NET Page Life Cycle Overview.

Advantages and Disadvantages of Using User Controls

For more information about Web custom controls, see Walkthrough: Developing and Using a Custom Server Control.

Module Reviews and Takeaways

Review questions and answers

1. What is the file extension of a master page?

   The file extension for a master page is .master, instead of .aspx—for example, **MasterPage.master**.

2. Which attribute overrides any master page setting specified in the web.config file?

   The **MasterPageFile** attribute of the **Page** directive overrides any master page setting specified in the web.config file.

3. What is the file extension of a user control?

   User controls are ASP.NET pages with an .ascx file extension.

4. What are the disadvantages of user controls?

   User interface and code can get duplicated, cannot be easily shared between Web applications, and code is visible to users of your user control.

Real-world issues and scenarios

1. You want to create a control that other developers can make changes to without recompiling the code. What is the easiest way to implement this?

   You should implement the control as a user control.

2. You want to create an overall master page, but you want to dictate two different types of layouts, based on the overall master page. What is the easiest way to implement this?

   You could create three master pages, with the first being the overall master page, and the other two being nested master pages that are based on the overall master page.

Best practices

Mention some best practices in the context of your own business situations.

- Use master pages whenever you have a layout that will be used for two or more pages.
- Apply master pages in the web.config file, if it is to be used with all or nearly all the pages on the Web site, or in specific areas of the Web site. This makes it easier to replace the master pages at a later stage.

Lab Review Questions and Answers

1. How will you programmatically attach master pages to an ASP.NET application?

   Answer: You can attach a master page to a content page, or assign a master page dynamically during the **PreInit** stage.

2. Why did you convert a Web Form into an ASP.NET user control in the lab?

   Answer: To access the functionality of the Web Form throughout the application, you need to convert the Web Form into a user control.

Module 6

Validating User Input

Contents:

Lesson 1

Overview of User Input Validation

Contents:

Question and Answers

What Is Input Validation?

Question: How do ASP.NET validation controls protect against spoofing and malicious code attacks?

Answer: ASP.NET runs all the validation controls on the server side, regardless of client-side validation. This helps protect against spoofing and malicious code.

Client-Side and Server-Side Validation

Question: When will you post a page to the server, even with errors?

Answer: By default, when client-side validation is being performed, the user cannot post the page to the server if there are errors on the page. However, you might find it necessary to enable the user to post a page, even with errors, if client-side validation might not be possible due to the validation requiring information or resources that are available only on the server, such as access to a database. To do this, you might have a **Cancel** button or a navigation button on a page, which will enable the user to submit the page even if some controls fail validation.

Additional Reading

What Is Input Validation?

For more information about how ASP.NET helps protect against script exploits, see Script Exploits Overview.

Lesson 2

ASP.NET Validation Controls

Contents:

Question and Answers

Overview of ASP.NET Validation Controls

Question: Why would you attach more than one validation control to an input control?

Answer: You would attach more than one validation control to an input control to specify that a control is required, and that it must contain a specific range of values.

Basic ASP.NET Validation Controls

Question*:* Which server control will you use to perform data-type validation?

Answer: You can use the **CompareValidator** or the **RangeValidator** server control to perform data-type validation.

RegularExpressionValidator Control

Question: For which validation purpose can you use the **RegularExpressionValidator** control?

Answer: You can use **RegularExpressionValidator** control to validate a value entered in a text box for a specific pattern. This type of validation allows you to check for predictable sequences of characters, such as those in social security numbers, e-mail addresses, telephone numbers, and postal codes. For example, for a US ZIP Code, the pattern of characters is 5 or 9 digits.

CustomValidator Control

Question: When could you use the **CustomValidator** control?

Answer: You could use the **CustomValidator** control when you can create a validation control that checks whether the value entered into a text box is an even number.

Combining Validation Controls

Question: When will you use the **RequiredFieldValidator** control?

Answer: You can use the **RequiredFieldValidator** control to ensure that a specific control differs from the default value, or is not left empty.

Demonstration: Adding Validation Controls to a Web Form

Question: Why do validation controls share the same validation properties?

Answer: The process of adding validation controls to a Web Form is the same for all the validation controls, because all the input validation controls share a common object model.

Positioning and Configuring Validation Controls on a Web Form

Question: Which property will you set to display text or a single character at the location of the validation control when both the **ErrorMessage** property and **Text** property are used?

Answer: To display text or a single character at the location of the validation control when both the **ErrorMessage** property and **Text** property are set, use the **Text** property.

Detailed Demo Steps

Demonstration: Adding Validation Controls to a Web Form

Demonstration Steps

1. Log on to 10267A-GEN-DEV as **Student**, with the password, **Pa$$w0rd**.
2. Open Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, click **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
3. Open an existing Web site from either D:\Demofiles\M6\VB\Validation or D:\Demofiles\M6\CS\Validation.
    - On the **File** menu of the Start Page - Microsoft Visual Studio window, click **Open Web Site.**
    - In the **Open Web Site** dialog box, in the **Folder** box, type either **D:\Demofiles\M6\VB\Validation** or **D:\Demofiles\M6\CS\Validation**, and then click **Open.**
4. Open the **Order** Web Form in Design view.
    - In Solution Explorer, right-click **Order.aspx**, and then click **View Designer**.
5. Add a **RequiredFieldValidator** control named **DueDateRequiredFieldValidator**, for the **DueDateTextBox** control, with an asterisk (*****) as the text, and **The Due Date must be filled in.** as the error message. The display must be dynamic.

```
<asp:RequiredFieldValidator ID="DueDateRequiredFieldValidator"
ControlToValidate="DueDateTextBox" runat="server" ErrorMessage="The Due Date must
be filled in." Text="*"></asp:RequiredFieldValidator>
```

    - In the Order.aspx window, place the cursor next to the **Due Date** TextBox control.
    - In the Toolbox, expand **Validation**, and then double-click **RequiredFieldValidator**.
    - In Properties window, in the **(ID)** box, type **DueDateRequiredFieldValidator,** in the **ControlToValidate** list, click **DueDateTextBox**, and in the **Display** list, click **Dynamic**. In the **Text** box, type *****, in the **ErrorMessage** box, type **The Due Date must be filled in.**, and then press ENTER.
6. Add a **RangeValidator** control named **DueDateRangeValidator** for the **DueDateTextBox** control, with an asterisk (*) as the text, and **The Due Date must be valid.** as the error message. The validator should accept only dates. Today's date should be set as the minimum value, and today's date plus 30 days should be set as the maximum value, both programmatically.

```
<asp:RangeValidator ID="DueDateRangeValidator" ControlToValidate="DueDateTextBox"
runat="server" ErrorMessage="The Due Date must be valid."
Text="*"></asp:RangeValidator >
```

    - In the Order.aspx window, place the cursor next to the **DueDateRequiredFieldValidator** control.
    - In the Toolbox, expand **Validation**, and then double-click **RangeValidator**.

- In Properties window, in the **(ID)** box, type **DueDateRangeValidator**, in the **ControlToValidate** list, click **DueDateTextBox,** and in the **Type** list, click **Date**. In the **Text** box, type *****, in the **ErrorMessage** box, type **The Due Date must be valid.,** and then press ENTER.
- In Solution Explorer, right-click **Order.aspx**, and then click **View Code.**
- In the Order.aspx.vb or Order.aspx.cs window, in the **Page_Load** event handler, append the following code.

[Visual Basic]

```
DueDateRangeValidator.MinimumValue = DateTime.Now.ToShortDateString()
DueDateRangeValidator.MaximumValue = (DateTime.Now +
        New TimeSpan(30, 0, 0, 0)).ToShortDateString()
```

[Visual C#]

```
DueDateRangeValidator.MinimumValue = DateTime.Now.ToShortDateString();
DueDateRangeValidator.MaximumValue = (DateTime.Now +
    new TimeSpan(30, 0, 0, 0)).ToShortDateString();
```

- In the Order.aspx.vb or Order.aspx.cs window, click the **Close** button.
- In the Microsoft Visual Studio message box, click **Yes.**

7. Add a **RegularExpressionValidator** control named **CustomerEmailAddressRegularExpressionValidator** for the **CustomerEmailAddressTextBox** control, with an asterisk (*) as the error message, and **\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*** as the validation expression.

```
<asp:RegularExpressionValidator
ID="CustomerEmailAddressRegularExpressionValidator"
ControlToValidate="CustomerEmailAddressTextBox" runat="server" ErrorMessage="*"
ValidationExpression="\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-
.]\w+)*"></asp:RegularExpressionValidator>
```

- In the Order.aspx window, place the cursor next to the **Email Address** TextBox control.
- In the Toolbox, expand **Validation,** and then double-click **RegularExpressionValidator.**
- In Properties window, in the **(ID)** box, type **CustomerEmailAddressRegularExpressionValidator,** in the **ControlToValidate** list, click **CustomerEmailAddressTextBox,** and in the **ErrorMessage** box, type **The Email Address must be valid..** In the **Text** box, type *****, in the **ValidationExpression** box, type **\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*,** and then press ENTER.

8. Add a **ValidationSummary** control named **OrderValidationSummary,** below the two **Button** controls.

```
<asp:ValidationSummary ID="OrderValidationSummary"
runat="server"></asp:ValidationSummary>
```

- In the Order.aspx window, place the cursor next to the **OrderCancelButton** TextBox control, and then press ENTER.
- In the Toolbox, expand **Validation**, and then double-click **ValidationSummary.**

- In Properties window, in the **(ID)** box, type **OrderValidationSummary.**

9. Save the **Order** Web Form, and view the changes in the browser.
   - In the Validation– Microsoft Visual Studio window, on the **File** menu, click **Save Order.aspx.**
   - In Solution Explorer, right-click **Order.aspx**, and then click **View in Browser**.

Note: Notice that today's date has been added to the **Order Date** box.

10. Save the order.
    - In the Order Entry – Windows Internet Explorer window, click the **Save** button.

Note: Notice that there is an asterisk next to the **Due Date** box, and an error message in the validation summary at the bottom.

11. Specify an invalid due date, and save the order.
    - In the Order Entry – Windows Internet Explorer window, in the **Due Date** box, type a date that is 31 days from today's date, in the format **m/d/yyyy**, and then click the **Save** button.

Note: Notice that there is an asterisk next to the **Due Date** box, at the same location as the one shown on the previous try to save the order, and there is a different error message in the validation summary at the bottom.

12. Specify a valid due date, and save the order.
    - In the Order Entry – Windows Internet Explorer window, in the **Due Date** box, type a date that is 30 days or less from today's date, in the format **m/d/yyyy**, and then click the **Save** button.

Note: Notice that there are no validation errors.

13. Specify an invalid email address and save the order.
    - In the Order Entry – Windows Internet Explorer window, in the **Email Address** box, type **claus@cohowinery,** and then click the **Save** button.

Note: Notice that there is an asterisk next to the **Email Address** box, and an error message in the validation summary at the bottom.

14. Specify a valid email address, and save the order.
    - In the Order Entry – Windows Internet Explorer window, in the **Email Address** box, type **claus@cohowinery.com**, and then click the **Save** button.

Note: Notice that there are no validation errors.

15. Close Windows® Internet Explorer®.
    - In the Order Entry – Windows Internet Explorer window, click the **Close** button.

Additional Reading

Overview of ASP.NET Validation Controls

For more information about Web custom controls, see Types of Validation for ASP.NET Server Controls.

RegularExpressionValidator Control

For more information about the .NET Framework regular expressions, see the **.NET Framework Regular Expressions** at http://go.microsoft.com/fwlink/?LinkID=180061&clcid=0x409.

For more information about the JScript 10.0 regular expressions, see the **Introduction to Regular Expressions** at http://go.microsoft.com/fwlink/?LinkID=180062&clcid=0x409.

Combining Validation Controls

For more information about the **RequiredFieldValidator** control, see the RequiredFieldValidator Class.

Demonstration: Adding Validation Controls to a Web Form

For more information about how to set the **ControlToValidate** property, see the BaseValidator.ControlToValidate Property.

For more information about how to set the **ValidationGroup** property, see the BaseValidator.ValidationGroup Property.

Lesson 3

Validating Web Forms

Contents:

Question and Answers

Adding the ValidationSummary Control

Question: How will you summarize error messages from a group of validation controls on a Web Form?

Answer: You can summarize the error messages from a group of validation controls on a Web Form by assigning the **ValidationSummary** control to a validation group. To assign the control, set the **ValidationGroup** property of the **ValidationSummary** control and of the other validation controls that comprise up the validation group.

Programmatically Validating Web Forms

Question: How do you verify if the content of a control or a page is valid?

Answer: To check if the content of a page or control is valid, you verify the **IsValid** property.

Additional Reading

Adding the ValidationSummary Control

For more information about adding and configuring a **ValidationSummary** control, see How to: Add and Configure a ValidationSummary Control.

For more information about **ValidationSummary** class, see ValidationSummary Class.

Programmatically Validating Web Forms

For more information about ensuring an ASP.NET Web Form is valid, see the Page.Validate Method.

Module Reviews and Takeaways

Review questions and answers

1. Which control will you use to perform the following validation tasks?

   You should use the following controls to perform the following validation tasks:

   - Verify that the user has entered the correct age in the **Age** field. **RequiredFieldValidator, RangeValidator**
   - Verify that the user has entered the correct telephone number format in the **Telephone Number** field. **RequiredFieldValidator, RegularExpressionValidator**
   - Verify that the user has entered the correct password twice in the **Password** and **Re-enter Password** fields. **RequiredFieldValidator, CompareValidator** (comparing two input controls)

Verify that all the fields in a form are correctly filled in. **ValidationSummary**

Real-world issues and scenarios

1. You want to make sure that an input control is always filled in by the user, but you also want it to be in a specific format. What is the easiest way to add this?

   Add a **RequiredFieldValidator** and a **RegularExpressionValidator** control.

2. None of the intrinsic ASP.NET validation controls fit your validation needs. What would you do to solve the issue?

   Create a new **CustomValidator** control, and add the validation logic as code.

Best practices

Mention some best practices in the context of your own business situations.

- Always add server-side validation code when using validation controls, because client-side scripting might be disabled, or a malicious user could compromise the user input, which could then be sent to the server.
- Always validate user input that is used for performing actions server-side, such as searching a database or accessing server-side resources.

Lab Review Questions and Answers

1. Why have you added the **RegularExpressionValidator** control for the e-mail address field?

   Answer: The **RegularExpressionValidator** control is added to the e-mail address field to ensure that the basic format of someone@microsoft.com is followed.

2. Why do you use a **ValidationSummary** control?

   Answer: The **ValidationSummary** control is used to display a summary of all validation errors that occurred on a Web page. If the **ErrorMessage** property of the validation control is not set, the error message will not be displayed for the validation control.

Module 7

Troubleshooting Microsoft® ASP.NET Web Applications

Contents:

Lesson 1

Debugging in ASP.NET

Contents:

Question and Answers

What Is Debugging?

Question: How will you ensure that your program or code is without any errors?

Answer: When you write a computer program, errors can and will occur. You might make a typographical error, your program might not perform as expected, or it might not run at all. When there is an error in your program, you need to find it and fix it. Finding and fixing errors is called debugging.

Gathering Debug Information at Run Time

Question: What are the two default configurations in Visual Studio 2010?

Answer: Visual Studio 2010 includes two configurations by default: debug, and release. The debug configuration automatically defines the debug and trace constants, enabling your application to provide the context to the developer who is troubleshooting. The option to generate debugging information is turned on by default, causing a program database (PDB) or debug file to be generated for each assembly in your solution. They appear in the same \bin folder as your assemblies.

Methods for Printing Debug Information

Question: Which method will you use to conditionally write a string of text of your choice?

Answer: To conditionally write a string of text of your choice, or to write conditionally the output from an object's **ToString** method, use the **Debug.WriteIf** method.

Debugging a Web Application

Question: How will you examine a variable or a value of the variable during a debugging session?

Answer: During the debugging session, you can assign values to the variable, and then examine the variable by using the Watch window.

Remote Debugging

Question: In what scenarios will you perform remote debugging?

Answer: You perform remote debugging when the program you are debugging is on the Web server, or if the program is performing differently on your computer.

Additional Reading

Discussion: Types of Errors

For more information about exception handling, see Exception Handling.

Debug Class

For more information about the Debug.Write method and its overloads, see Debug.Write Method.

For more information about the Debug.WriteLine method and its overloads, see Debug.WriteLine Method.

For more information about the Debug.WriteIf method and its overloads, see Debug.WriteIf Method.

For more information about the Debug.Print method and its overloads, see Debug.Print Method.

For more information about how to use the Call Stack window, see How to: Use the Call Stack Window.

For more information about the Debug.Assert method and its overloads, see Debug.Assert Method.

Gathering Debug Information at Run Time

For more information about how to create and initialize the TraceListeners class, see How to: Create and Initialize Trace Listeners.

For more information about the TraceListenerCollection class and its syntax, see TraceListenerCollection Class.

Debugging a Web Application

For more information about editing a value in a variable window, see How to: Edit a Value in a Variable Window.

For more information about working with the Watch window, see How to: Watch an Expression in the Debugger.

For more information about general debugging in Visual Studio 2010, see Debugging in Visual Studio.

For more information about ASP.NET debugging, see ASP.NET Debugging.

For more information about debugging script and code in Web applications, see Debugging Web Applications and Script.

Remote Debugging

For more information about the remote debugging setup, see Remote Debugging Setup.

For more information about debugging security, see Debugger Security.

Lesson 2

Tracing in ASP.NET

Contents:

Question and Answers

What Is Tracing?

Question: What is the purpose of using tracing functionality?

Answer: You can use tracing to view diagnostic information about a single request for an ASP.NET page simply by enabling it for your page or application. Tracing also allows you to write debug statements directly in your code without having to remove them from your application when you deploy it to production servers.

TraceContext Class

Question: When will you use the **TraceContext** class?

Answer: You can use the **TraceContext** class to append messages to specific trace categories, and to obtain a set of trace records at the end of request execution.

Tracing a Web Application

Question: How will you view trace information?

Answer: You can view trace information at the bottom of each page. Alternatively, you can also use the trace viewer or **Trace.axd** to view trace information that ASP.NET collects and caches, when tracing is enabled.

Additional Reading

TraceContext Class

For more information about the TraceContext class and its syntax, see TraceContext Class.

For more information about how to use System.Diagnostics.Trace class with the TraceContext class, see Walkthrough: Integrating ASP.NET Tracing with System.Diagnostics Tracing.

For more information about the TraceRecords messages collection, see TraceContextEventArgs.TraceRecords Property.

Tracing a Web Application

For more information about ASP.NET tracing and its features, see ASP.NET Tracing Overview.

For more information about how to interpret the tracing information, see Reading ASP.NET Trace Information.

Module Reviews and Takeaways

Review questions and answers

1. What is the difference between the System.Diagnostics.Trace object and the System.Diagnostics.Debug object?

   Messages written to the **Debug** object will only display when you run the Web application in the debugger. Messages written to the **Trace** object will be displayed in to the Web page as long as trace is enabled.

2. What is the difference between page-level tracing and application-level tracing?

   When you turn on page-level tracing for one page, trace information will only display for that page. When you enable application-level tracing, trace information is available in trace viewer, trace.axd, and trace information can be displayed for pages that have been requested.

3. How do you enable application-level tracing?

   To enable application-level tracing, you need to configure the **trace** element of the web.config file. The following code shows how you can enable application-level tracing.

```
<configuration>
...
<system.web>
<trace enabled="true" />
...
</system.web>
</configuration>
```

4. What are the types of errors that you need to handle when you develop applications?

   There are three types of errors that you need to handle when you develop applications: syntax errors, runtime errors, and semantic errors.

Real-world issues and scenarios

1. In a method, you want to write the value of a variable to the trace listeners, but only if a specific condition is true. What is the easiest way to implement this?

   Call **the Debug.WriteIf** method.

2. On a production server, you need to start tracing, but you do not want the users to see the trace output on each page they request. How will you implement this?

   You enable application-level tracing, and use the trace viewer trace.axd to view the trace output.

Best practices

Mention some best practices in the context of your own business situations.

- Messages written to the trace listeners that should not be included in a release version of your application should always be written by using the Debug object.
- Always disable tracing in a Web application before deployment to a production server.

Lab Review Questions and Answers

1. What are the steps required to enable page-level tracing?

   Answer: In the Web Form markup, add a **Trace** attribute to the **Page** directive, and then set the attribute value to **true**.

2. How will you explicitly turn on tracing in a component by using code?

[Visual Basic]

```
System.Web.HttpContext.Current.Trace.IsEnabled = True
```

[Visual C#]

```
System.Web.HttpContext.Current.Trace.IsEnabled = true;
```

Module 8

Managing Data in a Microsoft® ASP.NET 4.0 Web Application

Contents:

Lesson 1

Overview of ADO.NET

Contents:

Questions and Answers

What Is ADO.NET?

Question: What is the primary use of ADO.NET?

Answer: The primary use of ADO.NET is to access and manage data stored in relational database systems, such as SQL Server 2008.

ADO.NET Object Model

Question: How will you work with a **DataSet** to populate data?

Answer: You can work with a **DataSet** in the following combinations:

- Create a **DataTable**, **DataRelation** and **Constraint** within a DataSet programmatically, and populate the tables with data.
- Populate the **DataSet** with tables of data from an existing relational data source, by using a **DataAdapter**.
- Load and persist the **DataSet** contents by using XML.

Overview of ADO.NET Entity Framework

Question: What makes the Entity Framework different from most other data access frameworks?

Answer: The Entity Framework enables developers to work with data in the form of domain-specific objects and properties—such as customers and customer addresses—without having to concern themselves with the underlying database tables and columns where this data is stored.

Additional Reading

What Is ADO.NET?

For more information about ADO.NET, see ADO.NET (core data access).

For more information about the architecture of ADO.NET, see ADO.NET Architecture.

ADO.NET separates data access from data manipulation, and into discrete components that can be used separately, or in tandem. ADO.NET includes .NET Framework data providers for connecting to a database, executing commands, and retrieving results. Those results are either processed directly, placed in an ADO.NET **DataSet** object in order to be exposed to the user in an ad hoc manner, combined with data from multiple sources, or passed between tiers. The **DataSet** object can also be used independently of a .NET Framework data provider to manage data that is local to the application, or sourced from XML.

ADO.NET Object Model

For more information about how the DataSet, DataTable and DataView classes relate to each other, see DataSets, DataTables, and DataViews (ADO.NET) page on the Microsoft Web site.

For more information about the classes in connected layers, see the following links:

- DbCommand Class
- DbConnection Class
- DbDataAdapter Class
- DbDataReader Class

For more information about the classes in disconnected layers, see the following links:

- Constraint Class
- DataColumn Class
- DataRelation Class
- DataRow Class
- DataSet Class
- DataTable Class
- DataTableReader Class
- DataView Class

For more information about the **Odbc** classes, see the following links:

- OdbcCommand Class
- OdbcConnection Class
- OdbcDataAdapter Class
- OdbcDataReader Class

For more information about the **OleDb** classes, see the following links:

- OleDbCommand Class
- OleDbConnection Class
- OleDbDataAdapter Class
- OleDbDataReader Class

For more information about the **Sql** classes, see the following links:

- SqlCommand Class
- SqlConnection Class
- SqlDataAdapter Class
- SqlDataReader Class

Overview of ADO.NET Entity Framework

For more information on Data Development, see Data Developer Center.

For more information on LINQ to Entities, see LINQ to Entities Overview.

For more information on Entity SQL, see Entity SQL Overview.

For more information on EntityClient Provider, see EntityClient Provider for the Entity Framework.

The Entity Framework generates a class derived from the ObjectContext class that represents the entity container in the conceptual model. This object context provides the facilities for tracking changes, and managing identities, concurrency, and relationships. This class also exposes a **SaveChanges** method that writes inserts, updates, and deletes to the data source. Like queries, these changes are either made by commands that are automatically generated by the system, or by stored procedures that are specified by the developer.

For more information on managing Entity Framework, see Creating, Adding, Modifying, and Deleting Objects (Entity Framework).

For more information on Entity Designer, see ADO.NET Entity Data Model Designer Overview.

For more information about the EDM Generator, see EDM Generator (EdmGen.exe).

For more information on Entity Framework Data Modeling, see Data Modeling in the Entity Framework.

For more information about Object Services, see Object Services Overview (Entity Framework).

For more information about the Entity Data Model, see Entity Data Model.

For more information about EDM Tools, see the ADO.NET Entity Data Model Tools.

Lesson 2

Connecting to a Database

Contents:

Questions and Answers

Creating a Connection

Question: How can you change the data source type for a connection?

Answer: In the **Add Connection** dialog box, when you click **Change**, a **Change Data Source** dialog box appears that enables you to choose a different data source type.

Facilitating Data Transport Between Clients and Servers

Question: What are the four command-type properties of the **DataAdapter** object?

Answer: The **DataAdapter** object four command-type properties are: **SelectCommand**, **UpdateCommand**, **InsertCommand**, and **DeleteCommand**.

Additional Reading

Creating a Connection

For more information about how to work with the Server Explorer window to connect to data, see Connecting to Data with Server Explorer/Database Explorer.

For more information about how to create a connection string and how to keep a connection string safe in a configuration file to allow changing it after deploying the application, see Connection Strings (ADO.NET).

Facilitating Data Transport Between Clients and Servers

For more information about generating commands with the CommandBuilder class, see Generating Commands with CommandBuilders (ADO.NET).

For more information about Transact-SQL (T-SQL), see Transact-SQL Reference (Database Engine).

For more information about the SqlParameter class and SqlDbType enumeration, see the following pages on the Microsoft Web site:

- SqlParameter Class
- SqlDbType Enumeration

Lesson 3

Managing Data

Contents:

Questions and Answers

Retrieving Simple Data

Question: When should you use a **DataReader** object instead of a **DataSet** object?

Answer: You must choose between **DataReader** or **DataSet** objects based on your intended use for the data. Generally, you use **DataReader** objects to read data in one-time, read-only situations, such as when you access a stored password, or fill in a list-bound control. You use **DataSet** objects for more complicated data access, such as accessing a customer's entire order history.

Some of the data access factors to consider when you select between **DataSet** and **DataReader** objects are:

- Access to data. If you intend to both read from and write to your data source, you must use a **DataSet** object. **DataReader** objects are read-only connections, and you must use them only when the data will be used in a read-only situation.
- Access to multiple databases. If you intend to combine tables from one or more databases, you must use a **DataSet** object. **DataReader** objects are based on single or multiple SQL statements from a single database.
- Binding to controls. If you intend to bind the data to more than one control, you must use a **DataSet** object. **DataReader** objects can be bound only to a single control.
- Connection mode. If you intend to run in a disconnected mode, you must use a **DataSet** object. **DataReader** objects must run in a connected mode. You can also use multiple **DataReader** objects over a single connection if your data provider supports it.
- Data scanning. If you intend to scan both backward and forward through the data, you must use a **DataSet** object. **DataReader** objects only scan forward as the data is streamed from the database.
- Access speed. If you require high-speed access to your data source, use a **DataReader** object. **DataSet** objects access data from a database more slowly than **DataReader** objects, because **DataSet** objects store the data in an object on the Web server. There is also more overhead when you create a **DataSet** object because of its ability to read and write data, to and scan forward and backward. **DataReader** objects are faster because they are lightweight objects, and there is very little associated overhead, because it is both forward-only and read-only.

Retrieving Non-Simple Data

Question: How does the **Fill** method return the column names?

Answer: The **Fill** method uses the **DataReader** object implicitly to return the column names and types that are used to create the tables in the **DataSet**, and the data to populate the rows of the tables in the **DataSet**.

Manipulating Data

Question: What is the main purpose of using the **DataAdapter** object?

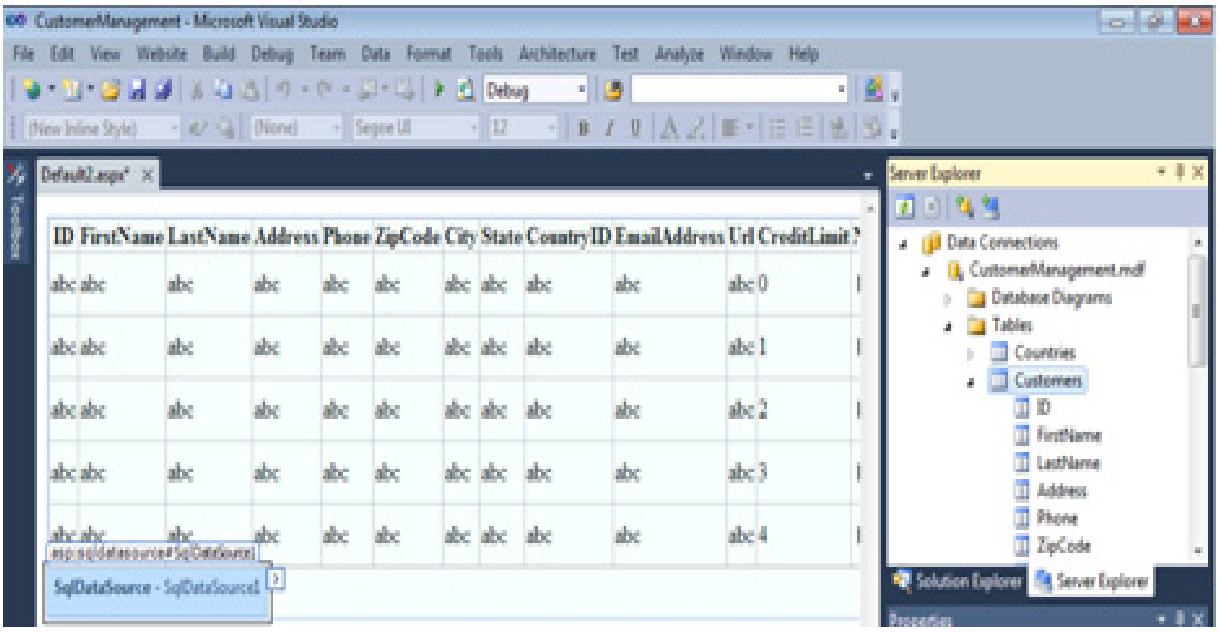
Answer: The **DataAdapter** object serves as a bridge between a **DataSet** and a data source, for retrieving and saving data.

Binding Data to Server Controls by Using the IDE

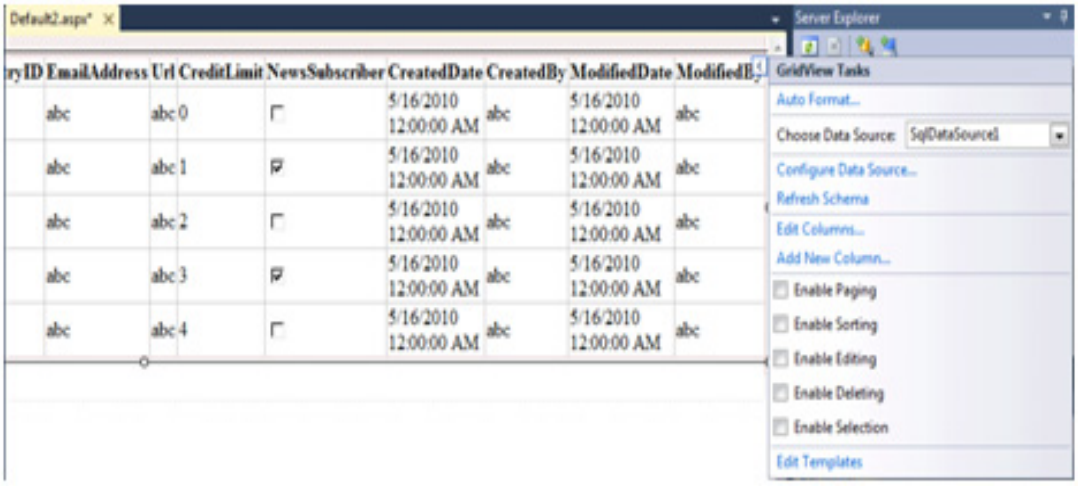
Question: What is the purpose of using the **SqlDataSource** control?

Answer: The **SqlDataSource** control helps you access data in databases by providing code-free retrieving and updating operations.

The following screen shot shows a **GridView** control and a **SqlDataSource** control in the Editor window that displays data from the **Customers** table.



The following screen shot shows how these properties are set in Design view.



Additional Reading

Retrieving Simple Data

For more information about retrieving data by using a DataReader, see Retrieving Data Using a DataReader (ADO.NET).

For more information about accessing the values in the DataReader, view the individual Getxxx methods for getting the values in the correct data type. See SqlDataReader Methods.

For more information about the properties and methods of the SqlCommand.ExecuteReader method, see SqlCommand.ExecuteReader Method.

For more information about the properties and methods of the SqlCommand.ExecuteScalar method, see SqlCommand.ExecuteScalar Method.

For more information about the SqlDataReader.NextResult method, see SqlDataReader.NextResult Method.

For more information about the SqlDataReader.Read method, see SqlDataReader.Read Method.

For more information about the SqlDataReader.HasRows property, see SqlDataReader.HasRows Property.

Retrieving Non-Simple Data

For more information about how to populate a DataSet by using the DataAdapter object, see Populating a DataSet from a DataAdapter (ADO.NET).

For more information about typed datasets, see Typed DataSets (ADO.NET).

For more information about working with DataSet objects, see DataSets, DataTables, and DataViews (ADO.NET).

For more information about applying the full table schema to a DataTable object,

see SqlDataAdapter.FillSchema Method.

For more information about locating specific rows in a DataTable object, see DataRowCollection.Find Method.

For more information about the DataTable.Select method, see DataTable.Select Method.

For more information about filtering, sorting, and having multiple views of the same data, see DataView Class.

Manipulating Data

For more information about how to modify data by using the **DataAdapter** class, see DbDataAdapter Class.

Binding Data to Server Controls by Using the IDE

For more information about how to use the SqlDataSource server control, see SqlDataSource Web Server Control Overview.

For more information about binding data by using the GridView control, see GridView Class.

For more information about how to use the LinqDataSource server control or LINQ, see LinqDataSource Web Server Control Overview.

For more information about LINQ, see Language-Integrated Query (LINQ).

Module Reviews and Takeaways

Review Questions

1. How do you create a connection to a database in Visual Studio 2010?

   In Server Explorer, right-click **Data Connections**, and then click **Add Connection**. In the **Add Connection** dialog box, configure the connection, and then click **OK**.

2. Which object is used to facilitate transport from the data source to a **DataSet** object and back again??

   Use the **DataAdapter** object.

3. What is the main difference between a **DataSet** and a **DataReader** object?

   The **DataSet** is a disconnected object that can be used after the connection to the data source has been closed. The **DataReader** class is designed to produce a read-only, forward-only stream that is returned from the data source. However, to retrieve values from a **DataReader**, the connection to the data source must remain open.

Real-world issues and scenarios

1. You need to connect to an Oracle database. What is the easiest way to connect to the Oracle database?

   Use the OLE DB .NET Framework Data Provider.

2. On a production server—where many connections are opened to the database—it appears that after a few days of uptime, the Web server and the database server slow down due to increased amount of memory and connection resources being used. What is the first thing you should check?

   You should check if you have closed your connection objects correctly.

Best practices

Mention some best practices in the context of your own business situations.

- Connection objects should always be explicitly disposed of.
- Use a **DataTable** object whenever you only need to work locally with the content of a single database entity, instead of a **DataSet** object. It saves resources.
- Whenever possible, place the connection string in the web.config file, making it accessible to all of the Web application, and making it easy for an administrator to change the file without recompiling code.

Lab Review Questions and Answers

1. How can you enable paging for a **GridView** control?

   Answer: When the control is selected, in the Properties window, set the **AllowPaging** property to **True**.

2. How do you connect to a SQL Server database?

   Answer: To connect to a SQL Server database by using the **SqlDataSource** control, you need to use a connection string, and you need access rights to a SQL Server database. You can use the **SqlDataSource** control to provide data to any data-bound control that supports the **DataSourceID** property, such as the **GridView** control.

Module 9

Managing Data Access Tasks by Using LINQ

Contents:

Lesson 1

Overview of LINQ

Contents:

Questions and Answers

What Is LINQ?

Question: What are the different LINQ implementations?

Answer: The different LINQ implementations are:

- LINQ to DataSet
- LINQ to Entities
- LINQ to Objects
- LINQ to SQL
- LINQ to XML

LINQ Query Operators

Question: What are the types of objects, on which the two sets of LINQ standard query operators operate?

Answer: The types of objects on which the two sets of LINQ standard query operators operate are **IEnumerable(T)** and **IQueryable(T)**.

What Is LINQ to XML?

Question: What are the various uses of LINQ to XML?

Answer: You can use LINQ to XML to:

- Load XML from files or streams.
- Serialize XML to files or streams.
- Create XML from scratch by using functional constructions.
- Query XML using XPath-like axes.
- Manipulate the in-memory XML tree by using methods such as **Add**, **Remove**, **ReplaceWith**, and **SetValue**.
- Validate XML trees by using XSD.
- Use a combination of these features to transform XML trees from one shape into another.

What Is LINQ to SQL?

Question: In which scenario will you use LINQ to SQL?

Answer: You can use LINQ to SQL to access SQL databases just as you would access an in-memory collection.

What Is LINQ to Entities?

Question: In which scenario will you use LINQ to Entities?

Answer: You can use LINQ to Entities to query an Entity Framework conceptual model.

Additional Reading

What Is LINQ?

For more information about LINQ to DataSet, see LINQ to DataSet Overview.

For more information about LINQ to Objects, see LINQ to Objects.

For more information about LINQ, see LINQ Portal.

LINQ Query Operators

For more information about the syntax for the standard query operators, see Query Expression Syntax for Standard Query Operators.

What Is LINQ to XML?

For more information about LINQ to XML, see LINQ to XML Overview.

What Is LINQ to SQL?

For more information about the typical steps for using LINQ to SQL, see Typical Steps for Using LINQ to SQL.

For more information about LINQ to SQL, see LINQ to SQL.

What Is LINQ to Entities?

For more information about LINQ to Entities, see LINQ to Entities.

Lesson 2

Managing XML Data by Using LINQ to XML

Contents:

Questions and Answers

Querying XML Data by Using LINQ to XML

Question: What is the purpose of using the **XElement** class?

Answer: The **XElement** class is one of the fundamental classes in LINQ to XML. It represents an XML element. You can use this class to: create elements; change the content of the element; add, change, or delete child elements; add, modify, and delete attributes of an element; or serialize the contents of an element.

Working with XML Data by Using LINQ to XML

Question: How will you use LINQ queries to process an XML tree?

Answer: LINQ queries operate on axis methods, and provide several flexible ways to navigate through and process an XML tree.

Displaying LINQ to XML Data

Question: What are the advantages of using LINQ to XML?

Answer: The most important advantage of LINQ to XML is its integration with LINQ. This integration enables you to write queries on the in-memory XML document, to retrieve collections of elements and attributes.

The query capability of LINQ to XML is comparable in functionality—although not in syntax—to XPath and XQuery. The integration of LINQ in the programming languages included with Visual Studio 2010 provides stronger typing, compile-time checking, and improved debugger support.

Another advantage of LINQ to XML is the ability to use query results as parameters to **XElement** and **XAttribute** object constructors, and this enables a powerful approach to creating XML trees. This approach—known as functional construction—enables developers to easily transform XML trees from one shape to another.

Additional Reading

Querying XML Data by Using LINQ to XML

For more information about the other LINQ to XML classes, their properties, and their methods, see LINQ to XML Classes Overview.

For more information about an how LINQ to XML compares to other Microsoft XML technologies, see LINQ to XML vs. Other XML Technologies.

For more information about querying XML trees, see Querying XML Trees.

For more information about an overview of the XmlDocument class, see XmlDocument Class.

For more information about an overview of the XPathDocument class, see XPathDocument Class.

For more information about how to process XML by using the XPath data model, see Process XML Data Using the XPath Data Model.

For more information about the XmlReader class, which provides fast, non-cached, forward-only access to XML data, see XmlReader Class.

For more information about how to read from a file by using the XmlReader class, see Reading XML with the XmlReader.

Working with XML Data by Using LINQ to XML

For more information about how to process XML by using the XPath data model, see Process XML Data Using the XPath Data Model.

For more information about mixed declarative code and imperative code bugs, see Mixed Declarative Code/Imperative Code Bugs (C#) (LINQ to XML).

Displaying LINQ to XML Data

For more information about Lambdas in Visual C#, see Lambda Expressions (C# Programming Guide).

For more information about Lambdas in Visual Basic, see Lambda Expressions.

Lesson 3

Managing SQL Data by Using LINQ to SQL and LINQ to Entities

Contents:

Questions and Answers

Querying SQL Data by Using LINQ

Question: What is the purpose of using LINQ to SQL?

Answer: LINQ to SQL is a .NET Framework component that provides a run-time infrastructure for managing relational data as objects. In LINQ to SQL, the data model of a relational database is mapped to an object model designed in the programming language. When the application runs, LINQ to SQL translates the language-integrated queries in the object model into SQL, and then sends them to the database for execution. When the database returns the results, LINQ to SQL translates them back to objects that you can work with, in your own programming language.

Working with SQL Data by Using LINQ

Question: Which methods will submit and save changes made to the local LINQ to SQL or LINQ to Entities context, to the database?

Answer: You can use the SubmitChanges method in LINQ to SQL, or the SaveChanges method in LINQ to Entities, to submit and save changes to the database.

Additional Reading

Querying SQL Data by Using LINQ

For more information about the SqlMetal.exe code generation tool, see SqlMetal.exe (Code Generation Tool).

For more information about the Object Relation Designer (O/R Designer), see O/R Designer Overview.

For more information about the ADO.NET Entity Data Model Designer (Entity Designer), see ADO.NET Entity Data Model Designer.

For more information about the Entity Data Model Wizard, see Entity Data Model Wizard.

For more information about the Update Data Model Wizard, see Update Model Wizard (Entity Data Model Tools).

For more information about the Generate Database Wizard, see Generate Database Wizard (Entity Data Model Tools).

For more information about the LINQ to Entities Queries, see Queries in LINQ to Entities.

Working with SQL Data by Using LINQ

For more information about making and submitting data changes, see Making and Submitting Data Changes (LINQ to SQL).

For more information about making and saving data changes, see Saving Changes and Managing Concurrency (Entity Framework).

Displaying SQL Data by Using LINQ

For more information about configuring the Where expressions for a LinqDataSource control, see Configure Where Expression – LinqDataSource.

For more information about configuring the Order By expressions for a LinqDataSource control, see Configure Order By Expression – LinqDataSource.

For more information about the overall configuration of the LinqDataSource control, see Configure Data Source Dialog Box - LinqDataSource.

For more information about configuring the Where expressions for an EntityDataSource control, see Filtering Data (EntityDataSource).

For more information about configuring the Order By expressions for an EntityDataSource control, see Ordering Results (EntityDataSource).

For more information about the overall configuration of the EntityDataSource control, see Configuring the EntityDataSource Control or Configure Data Source Wizard (EntityDataSource Control).

Module Reviews and Takeaways

Review questions and answers

1. What are the tools and features provided by Visual Studio 2010 to support application development using LINQ?

   The following are the tools and features provided by Visual Studio 2010 to support application development using LINQ:

   - **LINQ-Aware Code Editors**. LINQ-aware code editors for Visual Basic and Visual C# provide IntelliSense and LINQ-specific formatting capabilities.
   - **Visual Studio Debugger Support**. The Visual Studio debugger supports debugging of query expressions.

2. What are the three models provided by the .NET Framework for processing XML data?

   The .NET Framework three models for processing XML data are:

   - The **System.Xml.XmlDocument** class, which implements the W3C DOM Level 1 Core, and the Core DOM Level 2.
   - The **System.Xml.XPathDocument** class, which provides a fast, read-only in-memory representation of an XML document by using the XPath data model.
   - LINQ to XML.

3. What are the panes available in Object Relational Designer design surface?

   The Object Relational Designer design surface has two panes:

   - Entities pane. The entities pane is the main design surface that displays the entity classes, associations, and inheritance hierarchies.
   - Methods pane. The methods pane is the design surface that displays the **DataContext** methods that are mapped to stored procedures and functions.

Real-world issues and scenarios

1. You need to display data from an XML file that is queried by using LINQ to XML. What is the easiest way to display the data returned from the query?

   You can use the **XmlDataSource** control.

2. Your Web application has been deployed to a production server, but one of the tables in the SQL Server database has a new column added. You need to update the **DataContext** so that that new column can be returned in the LINQ to SQL queries when all the columns are returned. How will you implement this without using Visual Studio 2010?

   You should use the SqlMetal.exe command-line tool.

Best practices

Mention some best practices in the context of your own business situations.

- **DataContext** and **ObjectContext** objects should always be disposed of.

Lab Review Questions and Answers

1. What types of data can you query by using LINQ?

   Answer: You can query the DataSet, in-memory data such as arrays and lists, data in relational databases, and data stored in XML documents by using LINQ.

2. Why do you use stored procedures instead of accessing the database directly?

   Answer: Direct access and manipulation of data can be a very inefficient use of resources, and can create security risks. One way to improve the efficiency and security of database access is to create stored procedures on the database server, and then call these stored procedures from your application.

Module 10

Managing Data by Using Microsoft® ASP.NET Dynamic Data

Contents:

Lesson 1

Overview of ASP.NET Dynamic Data

Contents:

Questions and Answers

What Is ASP.NET Dynamic Data?

Question: What are the some of the capabilities provided by ASP.NET Dynamic Data?

Answer: Using ASP.NET Dynamic Data, you can:

- Build a Web site using scaffolding.
- Add dynamic data to an existing Web site.
- Add data field validation business logic.
- Customize the UI that is rendered to display and edit specific data fields or a specific table.

Demonstration: How to Implement ASP.NET Dynamic Data Scaffolding

Question: What is the use of ScaffoldTable attribute?

Answer: The **ScaffoldTable** attribute shows or hides a table in the UI.

Detailed Demo Steps

Demonstration: How to Implement ASP.NET Dynamic Data Scaffolding

Demonstration steps

To implement ASP.NET Dynamic Data scaffolding, you need to:

1. Log on to 10267A-GEN-DEV as Student, with the password, Pa$$w0rd.
2. Open Visual Studio 2010.
    - On the Start menu of 10267A-GEN-DEV, point to All Programs, click Microsoft Visual Studio 2010, and then click Microsoft Visual Studio 2010.
3. Open an existing Web site from either D:\Demofiles\M10\VB\Scaffolding or D:\Demofiles\M10\CS\Scaffolding.
    - On the File menu of the Start Page - Microsoft Visual Studio window, click Open Web Site.
    - In the Open Web Site dialog box, in the Folder box, type either D:\Demofiles\M10\VB\Scaffolding or D:\Demofiles\M10\CS\Scaffolding, and then click Open.
4. Build and run the Web site.
    - On the Build menu of the Scaffolding - Microsoft Visual Studio window, click Build Web Site.
    - In the Save File As dialog box, click Cancel.
    - On the Debug menu of the Scaffolding - Microsoft Visual Studio window, click Start Without Debugging.
5. Close the Windows® Internet Explorer® browser.
    - In the There are no accessible tables. Make sure that at least one data model is registered in Global. - Windows Internet Explorer window, click the Close button.
6. Open Global.asax.
    - In Solution Explorer, right-click Global.asax, and then click Open.
7. Uncomment the following lines of code at the top of the Shared or static RegisterRoutes method, and modify the uncommented line of code to register the AdventureWorksLT2008_DataModel.AdventureWorksLT2008_DataEntities ObjectContext with the model.

[Visual Basic]

```
DefaultModel.RegisterContext(GetType(AdventureWorksLT2008_DataModel.AdventureWorks
LT2008_DataEntities), New ContextConfiguration() With {.ScaffoldAllTables =
False})
```

[Visual C#]

```
DefaultModel.RegisterContext(typeof(AdventureWorksLT2008_DataModel.AdventureWorksL
T2008_DataEntities), new ContextConfiguration()
{
```

```
        ScaffoldAllTables = false
});
```

- In the Global.asax code window, change the following code,

[Visual Basic]

```
' DefaultModel.RegisterContext(GetType(YourDataContextType), New
ContextConfiguration() With {.ScaffoldAllTables = False})
```

[Visual C#]

```
//DefaultModel.RegisterContext(typeof(YourDataContextType), new
ContextConfiguration() { ScaffoldAllTables = false });
```

to:

[Visual Basic]

```
DefaultModel.RegisterContext(GetType(AdventureWorksLT2008_DataModel.AdventureWorksL
T2008_DataEntities), New ContextConfiguration() With {.ScaffoldAllTables = False})
```

[Visual C#]

```
DefaultModel.RegisterContext(typeof(AdventureWorksLT2008_DataModel.AdventureWorksL
T2008_DataEntities), new ContextConfiguration()
{
        ScaffoldAllTables = false
});
```

8. Build and run the Web site.
    - On the Build menu of the Scaffolding - Microsoft Visual Studio window, click Rebuild Web Site.
    - In the Save File As dialog box, click Cancel.
    - On the Debug menu, click Start Without Debugging.

Note: Explain that the reason the exception is thrown, is that the tables that scaffolding should use have not been specified, even if you have registered the data context.

9. Close Internet Explorer.
    - In the There are no accessible tables. Make sure that at least one data model is registered in Global. - Windows Internet Explorer window, click the Close button.
10. Modify the uncommented line of code to scaffold all tables.

[Visual Basic]

```
DefaultModel.RegisterContext(GetType(AdventureWorksLT2008_DataModel.AdventureWorks
LT2008_DataEntities), New ContextConfiguration() With {.ScaffoldAllTables = True})
```

[Visual C#]

```
DefaultModel.RegisterContext(typeof(AdventureWorksLT2008_DataModel.AdventureWorksL
T2008_DataEntities), new ContextConfiguration()
```

```
{
        ScaffoldAllTables = true
});
```

- In the Global.asax code window, change the following code,

[Visual Basic]

```
' DefaultModel.RegisterContext(GetType(YourDataContextType), New
ContextConfiguration() With {.ScaffoldAllTables = False})
```

[Visual C#]

```
//DefaultModel.RegisterContext(typeof(YourDataContextType), new
ContextConfiguration() { ScaffoldAllTables = false });
```

to:

[Visual Basic]

```
DefaultModel.RegisterContext(GetType(AdventureWorksLT2008_DataModel.AdventureWorks
LT2008_DataEntities), New ContextConfiguration() With {.ScaffoldAllTables = True})
```

[Visual C#]

```
DefaultModel.RegisterContext(typeof(AdventureWorksLT2008_DataModel.AdventureWorksL
T2008_DataEntities), new ContextConfiguration()
{
        ScaffoldAllTables = true
});
```

11. Build and run the Web site.
    - On the Build menu of the Scaffolding - Microsoft Visual Studio window, click Rebuild Web Site.
    - In the Save File As dialog box, click Cancel.
    - On the Debug menu, click Start Without Debugging.
12. View the Customers table, and then return to the home page.
    - In the Dynamic Data Site - Windows Internet Explorer window, under My tables, click Customers.
    - In the Customers - Windows Internet Explorer window, under DYNAMIC DATA SITE, click Back to home page.
13. View the Products table, and then return to the home page.
    - In the Dynamic Data Site - Windows Internet Explorer window, under My tables, click Products.
    - In the Products - Windows Internet Explorer window, under DYNAMIC DATA SITE, click Back to home page.

Note: Notice that now all the tables in the data context are now shown on the Web Site home page.

14. Close Internet Explorer.
    - In the Dynamic Data Site - Windows Internet Explorer window, click the Close button.
15. Modify the uncommented line of code to not scaffold all tables.

[Visual Basic]

```
DefaultModel.RegisterContext(GetType(AdventureWorksLT2008_DataModel.AdventureWorks
LT2008_DataEntities), New ContextConfiguration() With {.ScaffoldAllTables =
False})
```

[Visual C#]

```
DefaultModel.RegisterContext(typeof(AdventureWorksLT2008_DataModel.AdventureWorksL
T2008_DataEntities), new ContextConfiguration()
{
        ScaffoldAllTables = false
});
```

- In the Global.asax code window, change the following code,

[Visual Basic]

```
' DefaultModel.RegisterContext(GetType(YourDataContextType), New
ContextConfiguration() With {.ScaffoldAllTables = True})
```

[Visual C#]

```
//DefaultModel.RegisterContext(typeof(YourDataContextType), new
ContextConfiguration() { ScaffoldAllTables = true });

        to:
```

[Visual Basic]

```
DefaultModel.RegisterContext(GetType(AdventureWorksLT2008_DataModel.AdventureWorks
LT2008_DataEntities), New ContextConfiguration() With {.ScaffoldAllTables =
False})
```

[Visual C#]

```
DefaultModel.RegisterContext(typeof(AdventureWorksLT2008_DataModel.AdventureWorksL
T2008_DataEntities), new ContextConfiguration()
{
        ScaffoldAllTables = false
});
```

16. Close the Global.asax file.
    - In the Global.asax code window, click the Close button.
    - In the Microsoft Visual Studio message box, click Yes.
17. Add a new class file named Customer to the App_Code folder.
    - In Solution Explorer, right-click App_Code, and then click Add New Item.

- In the middle pane of the Add New Item - D:\Demofiles\M10\VB\Scaffolding\ or Add New Item - D:\Demofiles\M10\CS\Scaffolding\ dialog box, click Class, in the Name box, type Customer, and then click Add.

18. Import the System.ComponentModel.DataAnnotations namespace in the Customer class file.

[Visual Basic]

```
Imports System.ComponentModel.DataAnnotations
```

[Visual C#]

```
using System.ComponentModel.DataAnnotations;
```

- In either the App_Code/Customer.vb or App_Code/Customer.cs code window, type the following code.

[Visual Basic]

```
Imports System.ComponentModel.DataAnnotations
```

[Visual C#]

```
using System.ComponentModel.DataAnnotations;
```

19. Wrap the Customer class in the AdventureWorksLT2008_DataModel namespace.

[Visual Basic]

```
Namespace AdventureWorksLT2008_DataModel
        Public Class Customer
        ...
        End Class
End Namespace
```

[Visual C#]

```
namespace AdventureWorksLT2008_DataModel
{
        /// <summary>
        /// Summary description for Customer
        /// </summary>
        public class Customer
        {
              ...
        }
}
```

- In the App_Code/Customer.vb or App_Code/Customer.cs code window, add a namespace directive at the beginning and end of the Customer class.

[Visual Basic]

```
Namespace AdventureWorksLT2008_DataModel
        Public Class Customer
        ...
        End Class
End Namespace
```

[Visual C#]

```
namespace AdventureWorksLT2008_DataModel
{
        /// <summary>
        /// Summary description for Customer
        /// </summary>
        public class Customer
        {
              ...
        }
}
```

20. Apply the Partial or partial keyword to the Customer class.

[Visual Basic]

```
Partial Public Class Customer
```

[Visual C#]

```
public partial class Customer
```

- In the App_Code/Customer.vb or App_Code/Customer.cs code window, apply the Partial or partial keyword to the Customer class.

[Visual Basic]

```
Partial Public Class Customer
```

[Visual C#]

```
public partial class Customer
```

1. Remove the default constructor from the Customer class.

- In the App_Code/Customer.cs code window, remove the following code.

```
public partial class Customer
Remove the default constructor from the Customer class.
In the App_Code/Customer.cs code window, remove the following code.
```

[Visual C#]

```
public Customer()
(
        //
        // TODO: Add Constructor logic here
        //
}
```

Note: You only need to remove the default constructor from the **Customer** class if you are working in Visual C#.

21. Apply the ScaffoldTable attribute to the Customer class with a value of True/true.

- In the App_Code/Customer.vb or App_Code/Customer.cs code window, type the following code before the Customer class declaration.

[Visual Basic]

```
<ScaffoldTable(True)>
```

[Visual C#]

```
[ScaffoldTable(true)]
```

22. Build and run the Web site.
    - On the Build menu of the Scaffolding - Microsoft Visual Studio window, click Rebuild Web Site.
    - In the Save File As dialog box, click Cancel.
    - On the Debug menu, click Start Without Debugging.

    Note: Explain that now only the Customers table is exposed, because of the scaffolding attribute in the **Customer** class.
23. View the Customers table, and then return to the home page.
    - In the Dynamic Data Site - Windows Internet Explorer window, under My tables, click Customers.

    Note: Notice that all table columns are shown by default.
    - In the Customers - Windows Internet Explorer window, under DYNAMIC DATA SITE, click Back to home page.
24. Close all open windows.
    - In the Dynamic Data Site - Windows Internet Explorer window, click the Close button.
    - In the Scaffolding - Microsoft Visual Studio window, click the Close button.

    Note: If the Microsoft Visual Studio message box displays, click **No**.

Additional Reading

What Is ASP.NET Dynamic Data?

For more information about how the DynamicControl control displays the content that is defined for the field in template-based, data-bound controls, and uses ASP.NET Dynamic Data features, see DynamicControl Class.

For more information about how the DynamicField control represents a data field that is displayed in a data-bound control that uses ASP.NET Dynamic Data features, see the DynamicField Class.

Dynamic Data Project Infrastructure

For more information about how the MetaModel class represents one or multiple databases that are used by ASP.NET Dynamic Data, see MetaModel Class.

For more information about how the data model represents the information in a database and how the objects in the database are related to each other, see ASP.NET Dynamic Data Model Overview.

For more information about how the RouteCollection class provides a collection of routes for ASP.NET routing, see RouteCollection Class.

For more information about how the ScriptManager class manages ASP.NET Ajax script libraries and script files, partial-page rendering, and client proxy class generation for Web and application services, see ScriptManager Class.

ASP.NET Dynamic Data Scaffolding

For more information about how the data model represents the information in a database and how the objects in the database are related to each other, see ASP.NET Dynamic Data Model Overview.

For more information about how the MetaModel.RegisterContext method registers the data context with the meta model, see MetaModel.RegisterContext Method.

For more information about how the ContextConfiguration class provides information for a data-context instance to allow customization, see ContextConfiguration Class.

For more information about how the MetadataTypeAttribute class specifies the metadata class to associate with a data model class, see MetadataTypeAttribute Class.

ASP.NET Dynamic Data Templates

For more information about field templates, see ASP.NET Dynamic Data Field Templates Overview.

ASP.NET Dynamic Data Routing

For more information about ASP.NET Routing, see ASP.NET Routing.

Lesson 2

Applying ASP.NET Dynamic Data

Contents:

Questions and Answers

Demonstration: How to Create an ASP.NET Dynamic Data Web Site

Question: What is the use of the AdventureWorks.edmx file?

Answer: The AdventureWorks.edmx file extracts the schema information from the database, and is used for mapping entities to objects.

Detailed Demo Steps

Demonstration: How to Create an ASP.NET Dynamic Data Web Site

Demonstration steps

1. Open Visual Studio 2010.
    - On the Start menu of 10267A-GEN-DEV, point to All Programs, click Microsoft Visual Studio 2010, and then click Microsoft Visual Studio 2010.
    - In the User Account Control dialog box, in the text box, type Pa$$w0rd, and then click Yes.
2. Create a Web site by using the New Web Site dialog box.
    - On the File menu, click New Web Site.
    - In the New Web Site dialog box, in the left pane, click either Visual Basic or Visual C#.
    - In the middle pane, click ASP.NET Dynamic Data Entities Web Site.
    - In the Web location list, click File System, and in the text box, type either D:\Demofiles\M10\VB\DDWebSite or D:\Demofiles\M10\CS\DDWebSite, and then click OK.
3. Add an existing SQL Server 2008 Express Edition database to the Web site.
    - In Solution Explorer, right-click App_Data, and then click Add Existing Item.
    - In the middle pane of the Add Existing Item – D:\Demofiles\M10\VB\DDWebSite\ or Add Existing Item – D:\Demofiles\M10\CS\DDWebSite\ dialog box, in the File name box, type D:\Demofiles\M10\AdventureWorksLT2008_Data.mdf, and then click Add.
4. Add an ADO.NET Entity Data Model named AdventureWorks.edmx to the Web site.
    - In Solution Explorer, right-click D:\...\DDWebSite\, and then click Add New Item.
    - In the middle pane of the Add New Item – D:\Demofiles\M10\VB\DDWebSite\ or Add New Item – D:\Demofiles\M10\CS\DDWebSite\dialog box, in the middle pane, click ADO.NET Entity Data Model.
    - In the Name box, type AdventureWorks.edmx, and then click Add.
    - In the Microsoft Visual Studio message box, click Yes.

    Note: Point out that in a Web site, code files that are not directly related to a Web Form or user control must be placed in the App_Code folder.

    - In the Entity data Model Wizard, click Generate from database, and then click Next.
    - In the Entity data Model Wizard, on the Choose Your Data Connection page, in the Which data connection should your application use to connect to the database? list, click AdventureWorksLT2008_Data.mdf, and then click Next.
    - In the Entity data Model Wizard, on the Choose Your Database Objects page, in the Which database objects do you want to include in your model? pane, expand Tables, select Tables, remove the selection from BuildVersion, click Pluralize or singularize generated object names, and then click Finish.

Note: Point out that each table is represented as an entity that is named as the corresponding database table, but in singular.

5. Save and close AdventureWorks.edmx.
   - On the File menu of Visual Studio 2010, click Save App_Code/AdventureWorks.edmx.
   - In the App_Code/AdventureWorks.edmx window, click the Close button.
6. Open the Global.asax file, and add context registration to the RegisterRoutes procedure.

[Visual Basic]

```
DefaultModel.RegisterContext(GetType(AdventureWorksLT2008_DataModel.AdventureWorks
LT2008_DataEntities),
        New ContextConfiguration() With {.ScaffoldAllTables = True})
```

[Visual C#]

```
DefaultModel.RegisterContext(typeof(AdventureWorksLT2008_DataModel.AdventureWorksL
T2008_DataEntities), new ContextConfiguration()
{
        ScaffoldAllTables = true
});
```

   - In Solution Explorer, right-click Global.asax, and then click Open.
   - In the Global.asax window, add the following code at the top of to the RegisterRoutes procedure.

[Visual Basic]

```
DefaultModel.RegisterContext(GetType(AdventureWorksLT2008_DataModel.AdventureWorks
LT2008_DataEntities),
        New ContextConfiguration() With {.ScaffoldAllTables = True})
```

[Visual C#]

```
DefaultModel.RegisterContext(typeof(AdventureWorksLT2008_DataModel.AdventureWorksL
T2008_DataEntities), new ContextConfiguration()
{
        ScaffoldAllTables = true
});
```

Note: Point out that this will register the Entity Data model object context for use by the Dynamic Data field, and enable the automatic scaffolding of the data model.

7. Save and close the Global.asax file.
   - On the File menu of Visual Studio 2010, click Save Global.asax.
   - In the Global.asax window, click the Close button.
8. Build and run the Web site.
   - On the Debug menu of Visual Studio 2010, click Start Without Debugging.

Note: The page that displays in the browser displays a list of the tables that you added to the

data model.

9. Test the Web site for the Dynamic Data functionality, and view some of the pages by using the pager control.

- In the Dynamic Data Site – Windows Internet Explorer window, click Customers.

Note: The page displays the List view that contains the data from the Customers table.

- In the Page box, type 20, and then press ENTER.
- In the Customers - Windows Internet Explorer window, click Delete for a specific customer.
- In the Message from webpage message box, click Cancel.
- In the Customers - Windows Internet Explorer window, to the left of a customer, click Details.

Note: The page displays the Details view, which contains the data for the row selected from the Customers table.

- In the Customers - Windows Internet Explorer window, click Show all items.
- In the Customers - Windows Internet Explorer window, click FirstName.
- In the Customers - Windows Internet Explorer window, in the second row from the top, under CustomerAddresses, click View CustomerAddresses.

Note: The page displays the List view, which contains the customer address data.

- In the Customer list, click All.
- In the CustomerAddresses - Windows Internet Explorer window, at the end of the page, click Insert new item to create a new customer.
- In the CustomerAddresses - Windows Internet Explorer window, at the end of the page, click Insert, view the smart validation that has been added to the required fields, and then click Cancel.

Note: The page displays the List view, which contains the data from the CustomerAddresses table. This is the default view, and the one you are returned to, after cancelling.

- In the CustomerAddresses - Windows Internet Explorer window, at the end of the page, click Edit to modify a customer address.

Note: The page displays the Edit view, which contains the data for the CustomerAddresses table's selected row.

- In the CustomerAddresses - Windows Internet Explorer window, at the end of the page, click Cancel to cancel the edit operation.
- In the CustomerAddresses - Windows Internet Explorer window, click the Close button.
- In the DDWebSite - Microsoft Visual Studio window, click the Close button.

Additional Reading

Demonstration: How to Create an ASP.NET Dynamic Data Web Site

For more information about how to create a data-driven Web site with minimal or no additional code, Walkthrough: Creating a New Dynamic Data Web Site Using Scaffolding.

Adding Dynamic Behavior to ASP.NET Data-Bound Controls

For more information about how to enable dynamic behavior for ASP.NET Web controls that support ASP.NET Dynamic Data, see DynamicDataManager Class.

For more information about how to display the content that is defined for the field in template-based data-bound controls, using ASP.NET Dynamic Data features, see DynamicControl Class.

For more information about how to represent a data field that is displayed in a data-bound control that uses ASP.NET Dynamic Data features, see DynamicField Class.

For more information about how to represent a field that displays custom content in a data-bound control, see TemplateField Class.

For more information about how to get or set the name of the field template that is used to render the data field, see DynamicControl.UIHint Property.

Lesson 3

Customizing ASP.NET Dynamic Data Applications

Contents:

Questions and Answers

Demonstration: How to Create a Dynamic Data Field Template

Question: Why do you add the GetDateString method?

Answer: The **GetDateString** method returns a string representation of the specified date, and uses the specified format.

Detailed Demo Steps

Demonstration: How to Create a Dynamic Data Field Template

Demonstration steps

To create field templates to render the UI for displaying and editing **DateTime** data fields, you need to perform the following steps:

1. Open Visual Studio 2010.
    - On the Start menu of 10267A-GEN-DEV, point to All Programs, click Microsoft Visual Studio 2010, and then click Microsoft Visual Studio 2010.
2. Open the Scaffolding folder from either D:\Demofiles\M10\VB or D:\Demofiles\M10\CS.
    - On the File menu of Visual Studio 2010, click Open Web Site.
    - In the Open Web Site dialog box, in the Folder name, type either D:\Demofiles\M10\VB\Scaffolding or D:\Demofiles\M10\CS\Scaffolding, and then click Open.
3. Add a new Dynamic Data field named DateCalendar.ascx to the DynamicData\FieldTemplates folder.
    - In Solution Explorer, expand DynamicData, right-click FieldTemplates, and then click Add New Item.
    - In the left pane of the Add New Item – D:\Demofiles\M10\VB\Scaffolding\ or Add New Item – D:\Demofiles\M10\CS\Scaffolding\ dialog box, click either Visual Basic or Visual C#.
    - In the middle pane, click Dynamic Data Field, in the Name box, type DateCalendar.ascx, and then click Add.

    Note: The code must be placed in a code-behind file.

    Note: Notice that the two dynamic data field templates named DateCalendar.ascx and DateCalendar_Edit.ascx are created. The **DateCalendar.ascx** user control renders the UI for displaying the **DateTime** data fields, while the **DateCalendar_Edit.ascx** user control renders the UI for editing the **DateTime** data fields.
4. In the DateCalendar.ascx user control, remove the Text attribute of the Literal control with the value FieldValueString.
    - In the DynamicData/FieldTemplates/DateCalendar.ascx window, delete the Text attribute value FieldValueString.
5. Save and close the DateCalendar.ascx user control file.
    - In the Scaffolding – Microsoft Visual Studio window, on the File menu, click Save DynamicData/FieldTemplates/DateCalendar.ascx.
    - In the DynamicData/FieldTemplates/DateCalendar.ascx window, click the Close button.
6. Open either the DateCalendar.ascx.vb or DateCalendar.ascx.cs code-behind file.

- In Solution Explorer, expand DateCalendar.ascx, right-click either DateCalendar.ascx.vb or DateCalendar.ascx.cs, and then click Open.

7. Use the following code to override the user control OnDataBinding method to format the display of the DateTime value by excluding the time.

> **Note:** The value of the field is available in the **FieldValue** property.

[Visual Basic]

```
Protected Overloads Overrides Sub OnDataBinding(ByVal e As EventArgs)
       MyBase.OnDataBinding(e)
       Dim shortDate As String = String.Empty
       If FieldValue IsNot Nothing Then
             Dim dt As DateTime = Format(CType(FieldValue, DateTime), "d")
             shortDate = dt.ToShortDateString()
       End If
       Literal1.Text = shortDate
End Sub
```

[Visual C#]

```
protected override void OnDataBinding(EventArgs e)
{
       base.OnDataBinding(e);
       string shortDate = string.Empty;
       if (FieldValue != null)
       {
             DateTime dt = (DateTime) FieldValue;
             shortDate = dt.ToShortDateString();
       }
       Literal1.Text = shortDate;
}
```

- In either the DateCalendar.ascx.vb or DateCalendar.ascx.cs window, add the following code in the DynamicData_FieldTemplates_DateCalendar class.

[Visual Basic]

```
Protected Overloads Overrides Sub OnDataBinding(ByVal e As EventArgs)
       MyBase.OnDataBinding(e)
       Dim shortDate As String = String.Empty
       If FieldValue IsNot Nothing Then
             Dim dt As DateTime = Format(CType(FieldValue, DateTime), "d")
             shortDate = dt.ToShortDateString()
       End If
       Literal1.Text = shortDate
End Sub
```

[Visual C#]

```
protected override void OnDataBinding(EventArgs e)
{
       base.OnDataBinding(e);
       string shortDate = string.Empty;
       if (FieldValue != null)
       {
             DateTime dt = (DateTime) FieldValue;
             shortDate = dt.ToShortDateString();
```

```
        }

        Literal1.Text = shortDate;
}
```

8. Save and close the DateCalendar.ascx.vb or DateCalendar.ascx.cs code-behind file.
    - In the Scaffolding – Microsoft Visual Studio window, on the File menu, click either Save DynamicData/FieldTemplates/DateCalendar.ascx.vb or Save DynamicData/FieldTemplates/DateCalendar.ascx.cs.
    - In the DynamicData/FieldTemplates/DateCalendar.ascx.vb or DynamicData/FieldTemplates/DateCalendar.ascx.cs window, click the Close button.
9. Open the DateCalendar_Edit.ascx user control, and ensure that the AutoEventWireup is set to true.
    - In Solution Explorer, right-click DateCalendar_Edit.ascx, and then click Open.
    - In the DynamicData/Field...Calendar_Edit.ascx window, in the Control directive, ensure that the AutoEventWireup attribute is set to true, by placing the cursor inside the Control directive, and viewing the Properties Window in Visual Studio 2010.
10. Replace the FieldValueEditString expression of the Text attribute with the custom GetDateString method.
    - In the DynamicData/Field...Calendar_Edit.ascx window, change the FieldValueEditString value of the Text attribute in the Textbox control, with the value GetDateString.

```
<asp:TextBox ID="TextBox1" runat="server" Text='<%# GetDateString() %>' >
</asp:TextBox>
```

11. Append the following markup to the DateCalendar_Edit.ascx user control, to define the Calendar control that renders the UI to enable another way of editing dates.

[Visual Basic]

```
<asp:Calendar ID="Calendar1" runat="server"
       VisibleDate='<%# If(FieldValue, DateTime.Now) %>'
       SelectedDate='<%# If(FieldValue, DateTime.Now) %>'
       OnSelectionChanged="Calendar1_SelectionChanged" />
```

[Visual C#]

```
<asp:Calendar ID="Calendar1" runat="server"
       VisibleDate='<%# (FieldValue != null) ? FieldValue : DateTime.Now %>'
       SelectedDate='<%# (FieldValue != null) ? FieldValue : DateTime.Now %>'
       OnSelectionChanged="Calendar1_SelectionChanged" />
```

- In either the DynamicData/Field...Calendar_Edit.ascx window, append the following markup.

[Visual Basic]

```
<asp:Calendar ID="Calendar1" runat="server"
       VisibleDate='<%# If(FieldValue, DateTime.Now) %>'
       SelectedDate='<%# If(FieldValue, DateTime.Now) %>'
       OnSelectionChanged="Calendar1_SelectionChanged" />
```

[Visual C#]

```
<asp:Calendar ID="Calendar1" runat="server"
       VisibleDate='<%# (FieldValue != null) ? FieldValue : DateTime.Now %>'
       SelectedDate='<%# (FieldValue != null) ? FieldValue : DateTime.Now %>'
       OnSelectionChanged="Calendar1_SelectionChanged" />
```

12. Save and close DateCalendar_Edit.ascx user control.
    - In the Scaffolding – Microsoft Visual Studio window, on the File menu, click Save DynamicData/FieldTemplates/DateCalendar_Edit.ascx.
    - In the DynamicData/FieldTemplates/DateCalendar_Edit.ascx window, click the Close button.
13. Open either the DateCalendar_Edit.ascx.vb or DateCalendar_Edit.ascx.cs code-behind file. Add a GetDateString method. In the method, process the user's input as it is entered, by using the TextBox control. The method formats the date by using a short date format that excludes the time by using the following code.

[Visual Basic]

```
Protected Function GetDateString() As String
       If FieldValue <> Nothing Then
             Dim dt As DateTime = Format(CType(FieldValue, DateTime), "d")
             Return dt.ToShortDateString()
       Else
             Return String.Empty
       End If
End Function
```

[Visual C#]

```
protected string GetDateString()
{
       if (FieldValue != null)
       {
             DateTime dt = (DateTime)FieldValue;
             return dt.ToShortDateString();
       }
       else
             return string.Empty;
}
```

- In either the DynamicData/Field...Calendar_Edit.ascx.vb or DynamicData/Field...Calendar_Edit.ascx.cs window, append the following code in the DynamicData_FieldTemplates_DateCalendar class.

[Visual Basic]

```
Protected Function GetDateString() As String
       If FieldValue <> Nothing Then
             Dim dt As DateTime = Format(CType(FieldValue, DateTime), "d")
             Return dt.ToShortDateString()
       Else
             Return String.Empty
       End If
End Function
```

[Visual C#]

```
protected string GetDateString()
{
        if (FieldValue != null)
        {
              DateTime dt = (DateTime)FieldValue;
              return dt.ToShortDateString();
        }
        else
              return string.Empty;
}
```

14. Add a handler for the SelectionChanged event of the Calendar control. In the handler, set the Text property of the TextBox control to a formatted version of the selected date from the calendar.

Note: This displays the current selection of the user in the text box by using the following code:

[Visual Basic]

```
Protected Sub Calendar1_SelectionChanged(ByVal sender As Object, ByVal e As
EventArgs)
        ' Display value using the short date format.
        TextBox1.Text = Calendar1.SelectedDate.ToString("d")
End Sub
```

[Visual C#]

```
protected void Calendar1_SelectionChanged(object sender, EventArgs e)
{
        // Display value using the short date format.
        TextBox1.Text = Calendar1.SelectedDate.ToString("d");
}
```

- In either the DynamicData/Field...Calendar_Edit.ascx.vb or DynamicData/Field...Calendar_Edit.ascx.cs window, append the following code in the DynamicData_FieldTemplates_DateCalendar class.

[Visual Basic]

```
Protected Sub Calendar1_SelectionChanged(ByVal sender As Object, ByVal e As
EventArgs)
        ' Display value using the short date format.
        TextBox1.Text = Calendar1.SelectedDate.ToString("d")
End Sub
```

[Visual C#]

```
protected void Calendar1_SelectionChanged(object sender, EventArgs e)
{
        // Display value using the short date format.
        TextBox1.Text = Calendar1.SelectedDate.ToString("d");
}
```

15. Save and close the DateCalendar_Edit.ascx.vb or DateCalendar_Edit.ascx.cs code-behind file.

- In the Scaffolding – Microsoft Visual Studio window, on the File menu, click either Save DynamicData/FieldTemplates/DateCalendar_Edit.ascx.vb or Save DynamicData/FieldTemplates/DateCalendar_Edit.ascx.cs.
- In the Scaffolding – Microsoft Visual Studio window, on the File menu, click the Close button.

You have now created the field templates that render the UI for displaying and editing **DateTime** data field types. The field templates instruct Dynamic Data to apply the appropriate formatting and validation. If validation fails, the field templates generate appropriate error messages.

Additional Reading

Creating a Dynamic Data Page Template

For more information about how to create a custom page template, see How to: Customize the Layout of an Individual Table By Using a Custom Page Template.

Demonstration: How to Create a Dynamic Data Field Template

For more information about how the **FieldTemplateUserControl** class represents the base class for all field template controls, see FieldTemplateUserControl Class.

For more information about the data types that Dynamic Data displays by default, see ASP.NET Dynamic Data Default Field Templates.

For information about how to display custom data types, see MetaColumn.DataTypeAttribute Property.

Customizing Dynamic Data Scaffolding

For more information about how to make a site read-only by changing page templates, see How To: Create a Read-Only Dynamic Data Web Site.

Module Reviews and Takeaways

Review questions and answers

1. What are the capabilities provided by Scaffolding?

   Scaffolding provides the following capabilities:

   - Allows you to create a data-driven Web application with little or no code.
   - Provides built-in data validation based on the database schema, which is part of the data model.
   - Creates automatic filters for each foreign key or Boolean field, which is derived from the database schema.

2. What are the features of Dynamic Data?

   Dynamic Data offers the following features:

   - Supports Web scaffolding that can run a Web application based on reading the underlying database schema. Scaffolding is a mechanism that enhances an existing ASP.NET page framework by dynamically displaying pages based on the data model, without a physical page behind the scenes. Dynamic Data scaffolding can generate a standard UI from the data model.
   - Allows you to perform full data access operations (such as create, update, and remove; display operations), relational operations, and data validation operations.
   - Provides automatic support for foreign-key relationships. Dynamic Data detects relationships between tables and creates a UI that makes it easy for users to view data from related tables.
   - Ability to customize the UI to display and edit specific data fields.
   - Ability to customize the UI to display and edit data fields for a specific table.
   - Ability to customize data field validation. This allows you to keep the business logic in the data layer separate from the presentation layer.

3. List the built-in page templates.

   The built-in page templates are:

   - List view, or List.aspx
   - Details view, or Details.aspx
   - Edit view, or Edit.aspx
   - List Details, view or ListDetails.aspx
   - Insert view, or Insert.aspx

Real-world issues and scenarios

1. You need to modify the page template for displaying all rows in any registered database entity. What is the easiest way to do this?

   You should modify the List.aspx page template**.**

2. You need to create a specific view for a single entity in your data model. How do you do this?

   Create a subfolder in the **DynamicData\CustomPages** folder, and name the subfolder the same as the entity in your data context, but use a plural ending. Create a new content page, and then add it to the subfolder.

Best practices

- Do not modify the generated DataContext or ObjectContext classes. Instead, create partial classes and place the custom code in partial classes. When the DataContext or ObjectContext classes are regenerated, any changes are overridden.

Lab Review Questions and Answers

1. In which file should you add the registration of the context?

   Answer: The context registration should be added to the Global Application file.

2. What is the use of the ScaffoldColumnAttribute attribute?

   Answer: The **ScaffoldColumnAttribute** attribute specifies whether a column is exposed by the scaffolding mechanism.

Module 11

Creating a Microsoft® ASP.NET Ajax-enabled Web Forms Application

Contents:

Lesson 1

Introduction to Ajax

Contents:

Questions and Answers

What Is Asynchronous JavaScript and XML?

Question: What are some of the benefits that Ajax provides?

Answer: The following are some of the benefits that Ajax provides:

- In many situations, related pages on a Web site consist of common content. In the traditional method of performing a postback to the server, the content would have to be reloaded with every request. However, by using Ajax, a Web application can request only the content that needs to be updated. This reduces load time and bandwidth usage considerably.
- The use of asynchronous requests allows the client's Web browser UI to be more interactive and respond more quickly to user input. In addition, sections of pages can be reloaded individually.
- Users will perceive the application to be faster or more responsive, even if the application has not changed on the server side.
- Ajax minimizes the number of connections to the server, because style sheets and scripts are requested only once.
- State can be maintained throughout a Web site, because JavaScript variables will persist, and the main container page need not be reloaded.

Additional Reading

Architecture of ASP.NET Ajax

For more information about ASP.NET Ajax, see ASP.NET AJAX Roadmap.

Lesson 2

Creating an ASP.NET Ajax Application by Using the Ajax Features for ASP.NET

Contents:

Questions and Answers

What Are the Ajax Features for ASP.NET?

Question: What are the ASP.NET Ajax server-side controls that help you add Ajax functionality to an ASP.NET Web Forms application, without the need for any scripting?

Answer: ASP.NET Ajax server-side controls such as the **ScriptManager, UpdatePanel**, and the **UpdateProgress** controls, help you add Ajax functionality to an ASP.NET Web Forms application without writing any JavaScript.

ASP.NET Ajax Server Controls

Question: What are some of the scenarios in which you can use the **ScriptManager** control, the **Timer** control, the **UpdateProgress** control, and the **UpdatePanel** control?

Answer: The following are some of the scenarios for using the ASP.NET Ajax server controls:

Server control	Scenarios
ScriptManager control	You must use a **ScriptManager** control on a page to enable the following Ajax Features for ASP.NET: • Client-script functionality of the Microsoft Ajax Library, and any custom script that you want to send to the browser. • Partial-page rendering, which enables regions on the page to be independently refreshed without a postback. The ASP.NET **UpdatePanel**, **UpdateProgress**, and **Timer** controls require a **ScriptManager** control to support partial-page rendering. • JavaScript proxy classes for Web services, which enable you to use client script to access Web services and specially marked methods in ASP.NET Web Forms. It does this by exposing the Web services and page methods as strongly-typed objects. • JavaScript classes to access ASP.NET authentication, profile, and roles application services.
Timer control	You use the **Timer** control when you want to: • Periodically update the contents of one or more **UpdatePanel** controls without refreshing the whole Web page. • Run code on the server every time a **Timer** control causes a postback. • Synchronously post the whole Web page to the Web server at defined intervals.
UpdatePanel control	The **UpdatePanel** control is a server control that helps you develop Web pages with complex client behavior that makes a Web page appear more interactive to users. Writing code that coordinates between server and client to update only specified parts of a Web page usually requires in-depth knowledge of ECMAScript or JavaScript. However, by using the **UpdatePanel** control, you can enable a Web page to participate in partial-page updates without writing any client script. If you want, you can add a custom client script to enhance the client user experience. When you use an **UpdatePanel** control, the page behavior is browser-

	independent, and can potentially reduce the amount of data that is transferred between the client and the server.
UpdateProgress control	The **UpdateProgress** control helps you design a more intuitive UI when a Web page contains one or more **UpdatePanel** controls for partial-page rendering. If a partial-page update is slow, you can use the **UpdateProgress** control to provide visual feedback about the status of the update. You can put multiple **UpdateProgress** controls on a page, each associated with a different **UpdatePanel** control. Alternatively, you can use one **UpdateProgress** control, and associate it with all **UpdatePanel** controls on the page.

Uses of the ScriptManager Control

Question: How many **ScriptManager** controls can you add to a Web page?

Answer: You can add only one instance of the **ScriptManager** control to a Web page. If a page already contains a **ScriptManager** control, but a nested or parent component needs additional features of the **ScriptManager** control, the component can include a **ScriptManagerProxy** control.

Demonstration: How to Add the UpdatePanel Control

Question: What should you define in a trigger in the **UpdatePanel** control?

Answer: The trigger specifies the postback control and the event that causes a panel to update. When the specified event of the trigger control is raised, the update panel is refreshed. If a control event is not specified for a control, the trigger will use the default event for the specified control. For example, for the **Button** control, the default event is the **Click** event.

Partial-Page Updates

Question: What are some scenarios when you may want to handle the events of the **PageRequestManager** class?

Answer: You can enable partial-page updates by using the **ScriptManager** and **UpdatePanel** Web server controls. Partial-page updates require no client scripting. However, you can use the **PageRequestManager** class and client script when you want to do the following:

- Control how multiple asynchronous postbacks are processed. The default behavior is that the last postback takes precedence. The **PageRequestManager** class allows you to give priority to a specific postback, and cancel other postbacks that are being processed.
- Provide visual cues or other notification to mark regions on the page that are updated or created as a result of the last asynchronous postback. This improves the user experience, especially in scenarios where multiple **UpdatePanel** controls are used.
- Display status messages during asynchronous postback. If the postback take a long time to process, you might want to show a progress indicator, such as an animated image. You can also give the user the option to cancel the postback.
- Provide custom error-message handling for partial-page updates. If an unexpected error occurs during an asynchronous postback, you can handle the error in client script.

- Access the underlying request and response objects that are used for the asynchronous postback.

Detailed Demo Steps

Demonstration: How to Add the ScriptManager Control

Demonstration steps

1. Open Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
2. Create a new ASP.NET Web site.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **New Web Site**.
    - In the **New Web Site** dialog box, in the left pane, click either **Visual Basic** or **Visual C#**.
    - In the middle pane, ensure that **ASP.NET Web Site** is selected.
    - In the **Web location** box of the **New Web Site** dialog box, ensure **File System** is selected, and then click **OK**.

    Note: You can add the **ScriptManager** control after you create an ASP.NET Web site.
3. Add a Web Form named **Employees.aspx**, to the Web site.
    - In Solution Explorer, right-click **C:\...\WebSite1**, and then click **Add New Item**.

    Note: In this demonstration, the name of the Web site is **WebSite1**.
    - In the **Add New Item – C:\Users\student\Documents\Visual Studio 2010\WebSites\WebSite1** dialog box, in the left pane, click either **Visual Basic** or **Visual C#**.
    - In the middle pane, click **Web Form**.
    - In the **Name** box, type **Employees.aspx**, clear the **Place code in separate file** check box, and then click **Add**.
4. Open the Web Form in the Design view.
    - In the Employees.aspx window, click **Design**.
5. Add a **ScriptManager** control to the Web Form.
    - In the **AJAX Extensions** tab of the toolbox, double-click the **ScriptManager** control to add it to the page.
6. Set the **ID** attribute of the **ScriptManager** control to a value of **EmployeesScriptManager**.
    - In the Properties window, in the **ID** attribute, change the **ScriptManager1** to **EmployeesScriptManager**.
7. Add an **UpdatePanel** control under the **ScriptManager** control.
    - In the **AJAX Extensions** tab of the toolbox, drag the **UpdatePanel** control under the **ScriptManager** control.
    - In the Properties window, set the **ID** property of the **UpdatePanel** control to a value of **EmployeesUpdatePanel**.

- In the WebSite1 – Microsoft Visual Studio window, click **Close** button.

Note: After adding the **ScriptManager** control, you can add controls that you want asynchronously refreshed to **EmployeesUpdatePanel**, and set their properties by using the Properties window. You can also add any code to the events of the page and the controls to the code-behind file.

Demonstration: How to Add the UpdatePanel Control

Demonstration steps

To run the demonstration, you need to perform the following steps:

1. Open Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then **Microsoft Visual Studio 2010**.
2. Create a new ASP.NET Web site.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **New Web Site**.
    - In the **New Web Site** dialog box, in the left pane, click either **Visual Basic** or **Visual C#**.
    - In the middle pane, ensure that **ASP.NET Web Site** is selected.
    - In the **Web location** box of the **New Web Site** dialog box, ensure **File System** is selected, and then click **OK**.
3. Add a Web Form named **UpdatePanel.aspx**, to the Web site.
    - In Solution Explorer, right-click **C:\Users\student\Documents\Visual Studio 2010\WebSites\WebSite1**, and then click **Add New Item**.
    - In the **Add New Item – C:\Users\student\Documents\Visual Studio 2010\WebSites\WebSite1** dialog box, in the left pane, click either **Visual Basic** or **Visual C#**.
    - In the middle pane, click **Web Form**.
    - In the **Name** box, type **UpdatePanel.aspx**, and then click **Add**.
4. Open the Web Form in the Design view.
    - In the UpdatePanel.aspx window, click **Design**.
5. Add **ScriptManager** and **UpdatePanel** controls in the Design view of the **UpdatePanel.aspx** Web Form.
    - In the UpdatePanel.aspx window, point to **Toolbox**.
    - In Toolbox, under **AJAX Extensions**, double-click **ScriptManager**.
    - In the UpdatePanel.aspx window, point to **Toolbox**.
    - In Toolbox, under **AJAX Extensions**, double-click **UpdatePanel**.
6. Add a **ContentTemplate** element to the **UpdatePanel**.
    - In the UpdatePanel.aspx window, click **Source**.

- In the UpdatePanel.aspx window, place the cursor after the opening tag of the **UpdatePanel** control, and then type the following code.

```
<ContentTemplate></ContentTemplate>
```

7. On the form, inside the **UpdatePanel** control, add a **Button** control to the **ContentTemplate**.
   - In the UpdatePanel.aspx window, click **Design**.
   - In **Toolbox**, expand **Standard**, and then drag the **Button** control inside the **UpdatePanel** control.
8. Set the **Text** property of the **Button** control to **InsidePanel**.
   - In the Properties window, change the value of the **Text** property from **Button** to **InsidePanel**.
9. On the form, outside the **UpdatePanel** control, add another **Button** control.
   - In **Toolbox**, double-click **Button**.
10. Set the **Text** property of the **Button** control to the value **OutsidePanel**.
    - In the Properties window, change the value of the **Text** property from **Button** to **OutsidePanel**.
11. Save the **UpdatePanel.aspx** Web Form, and then view the page in the browser.
    - On the **File** menu of Visual Studio 2010, click **Save UpdatePanel.aspx**.
    - In the UpdatePanel.aspx window, right-click anywhere, and then click **View in Browser**.
12. Test both **InsidePanel** and **OutsidePanel** buttons and verify that the button inside the **UpdatePanel** control does not cause a page postback and the button outside **UpdatePanel** control causes a postback.
    - In the http://localhost:49158/WebSite1/UpdatePanel.aspx - Windows Internet Explorer window, click **InsidePanel**, and then click **OutsidePanel**.

    Note: Notice that the button inside the **UpdatePanel** control does not cause a page postback and the button outside **UpdatePanel** control causes a page postback.
13. Close Internet Explorer.
    - In the http://localhost:49158/WebSite1/UpdatePanel.aspx - Windows Internet Explorer window, click the **Close** button.
14. Delete the **InsidePanel** button.
    - In the UpdatePanel.aspx window, delete the **InsidePanel** button.
15. Add a **Label** control inside the **UpdatePanel** control.
    - In Toolbox, under **Standard**, double-click **Label**.
16. Add code to the **Click** event handler to set the **Text** of the **Label** to the string **Current Time:**, followed by the current date and time.
    - In the UpdatePanel.aspx window, double-click the **OutsidePanel** control.

- In the UpdatePanel.aspx.vb or UpdatePanel.aspx.cs window, type the following code in the **Click** event handler.

[Visual Basic]

```
Label1.Text = "Current Time: " & DateTime.Now.ToString()
```

[Visual C#]

```
Label1.Text = "Current Time: " + DateTime.Now.ToString();
```

17. Switch to the Design view of the **UpdatePanel.aspx** Web Form.
    - In Visual Studio 2010, click **UpdatePanel.aspx**.
18. View the properties of the **UpdatePanel** control.
    - In the UpdatePanel.aspx window, click the **UpdatePanel** control.
    - In the Properties window, click the ellipsis of the **Triggers** property.
19. Add a **ControlID** property of the new trigger **to Button2**.
    - In the **UpdatePanelTrigger Collection Editor** dialog box, click **Add**.
    - In the **ControlID** property, type **Button2**, and then click **OK**.
20. Save the Web site, and then view the **UpdatePanel.aspx** Web Form in the browser.
    - On the **File** menu of Visual Studio 2010, click **Save All**.
    - In the UpdatePanel.aspx window, right-click anywhere, and then click **View in Browser**.
21. Test the **OutsidePanel** button.
    - In the http://localhost:49158/WebSite1/UpdatePanel.aspx - Windows Internet Explorer window, click **OutsidePanel** two times, and verify that the content in the label changes, but a page refresh does not occur.
22. Close Internet Explorer.
    - In the http://localhost:49158/WebSite1/UpdatePanel.aspx - Windows Internet Explorer window, click the **Close** button.
    - In the WebSite1 – Microsoft Visual Studio window, click **Close** button.

Additional Reading

ASP.NET Ajax Server Controls

For more information about the ScriptManager control, see ScriptManager Control Overview.

For more information about the UpdatePanel control, see UpdatePanel Control Overview.

For more information about the UpdateProgress control, see UpdateProgress Control Overview.

For more information about the Timer control, see Timer Control Overview.

Uses of the ScriptManager Control

For more information about the ScriptManagerProxy control, see ScriptManagerProxy Web Server Control Declarative Syntax.

Demonstration: How to Add the UpdatePanel Control

For more information about how to add dynamic partial-page updates to your ASP.NET application using ASP.NET Ajax, watch the video on the [How Do I:] Implement Dynamic Partial-Page Updates with ASP.NET AJAX?.

For more information about how the **UpdatePanel** control enables sections of a page to be partially rendered without a postback, see UpdatePanel Class.

Partial-Page Updates

For more information about how the **PageRequestManager** manages partial-page updates of server **UpdatePanel** controls in the browser, see Sys.WebForms.PageRequestManager Class.

Lesson 3

Extending an ASP.NET Web Forms Application by Using the Ajax Control Toolkit

Contents:

Questions and Answers

Overview of the Ajax Control Toolkit

Question: What advantages does the Ajax Control Toolkit (part of the Microsoft Ajax Library) provide?

Answer: The Ajax Control Toolkit (part of the November 2009 Microsoft Ajax Library release) contains more than 40 controls that enable you to easily create rich, interactive Web pages. For example, you can use the Toolkit controls to display HTML editors, auto-complete textboxes, cascading drop-down lists, picture slide shows, and modal popup dialog boxes.

You do not need to know JavaScript to use the Ajax Control Toolkit. For example, if you want to add a popup calendar to a **TextBox** control, you can simply select the **Add Extender** task option, and then select the **Toolkit Calendar** from a dialog box.

The Microsoft Ajax Library—which contains the Ajax Control Toolkit—is a community-supported library, and is not supported by Microsoft. You can download the Microsoft Ajax Library from the CodePlex.com Web site.

Demonstration: How to Add an Ajax Extender Control

Question: What is the purpose of using the **ConfirmButtonExtender** control with a **Button** control?

Answer: The **ConfirmButtonExtender** control is used with a **Button** control to confirm the click action of the **Button** control.

The **ConfirmButton** is an extender that catches click events on a button, or any instance of a type derived from Button, and displays a message to the user. If the **OK** button is clicked in the displayed message, the button or link functions normally. If the **OK** button is not clicked, then the click is trapped, and the button will not perform its default submit behavior.

> For more information and tutorials on the Ajax Control Toolkit, see ASP.NET AJAX Videos and AJAX Control Toolkit Tutorials.

Detailed Demo Steps

Demonstration: How to Add an Ajax Extender Control

Demonstration steps

1. Open Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
2. Create a new ASP.NET Web site named **AjaxExtenderControl**.
    - On the **File** menu of Visual Studio 2010, click **New Web Site**.
    - In the **New Web Site** dialog box, in the left pane, click either **Visual Basic** or **Visual C#**, in the middle pane, ensure that **ASP.NET Web Site** is selected, in the **Web location** box, type **C:\Users\student\Documents\Visual Studio 2010\WebSites\AjaxExtenderControl\AjaxExtenderControl**, and then click **OK**.
3. Add a Web Form named **AjaxExtenderControl.aspx** to the Web site.
    - In Solution Explorer, right-click **C:\...\AjaxExtenderControl**, and then click **Add New Item**.
    - In the **Add New Item – C:\Users\student\Documents\Visual Studio 2010\WebSites\AjaxExtenderControl** dialog box, in the left pane, click either **Visual Basic** or **Visual C#**.
    - In the middle pane, click **Web Form**.
    - In the **Name** box, type **AjaxExtenderControl.aspx**, clear the **Place code in separate file** check box, and then click **Add**.
4. Open the Web Form in Design view.
    - In the AjaxExtenderControl.aspx window, click **Design**.
5. Create a tab named **Ajax Control Toolkit**, in the Toolbox.
    - In the AjaxExtenderControl.aspx window, point to **Toolbox**, right-click anywhere in Toolbox, click **Add Tab**, type **Ajax Control Toolkit**, and then press ENTER.
6. Navigate to **C:\Program Files\ASP.NET Ajax Library\WebForms\Debug**, and add the items from the **AjaxControlToolkit.dll** file to the **Ajax Control Toolkit** tab.
    - On the **Start** menu, click **Computer**.
    - In the Computer window, navigate to **C:\Program Files\ASP.NET Ajax Library\WebForms\Debug**, and then drag the **AjaxControlToolkit.dll** file to the **Ajax Control Toolkit** tab of the Toolbox.

    Note: You can view the added Toolbox items in the **Ajax Control Toolkit** tab. If you cannot view the items, right-click in **the Ajax Control Toolkit** tab, click **Reset Toolbox**, and then drag the **AjaxControlToolkit.dll** file to the **Ajax Control Toolkit** tab of the Toolbox.
7. Add a **Label** and a **Button** control to the **AjaxExtenderControl.aspx** Web form.
    - In Toolbox, under **Standard**, double-click the **Label** control.
    - In Toolbox, under **Standard**, double-click the **Button** control.

- In the AjaxExtenderControl.aspx window, double-click the **Button** control.

8. Create a **Click** event handler for the **Button** control, and set the **Text** property of the label to the string, **Button click occurred**.

    - In the AjaxExtenderControl.aspx.vb or AjaxExtenderControl.aspx.cs code-behind window, in the event handler, add the code to set the **Text** property of the **Label1.Text** to the string **Button click occurred**.

```
[Visual Basic]
Label1.Text = "Button click occurred"
[Visual C#]
```

```
Label1.Text = "Button click occurred";
```

1. Save the changes, and view the **AjaxExtenderControl.aspx** Web Form in the browser.

    - On the **File** menu of Visual Studio 2010, click **Save All**.
    - In AjaxExtenderControl – Microsoft Visual Studio window, click **AjaxExtenderControl.aspx**, right-click anywhere, and then click **View in Browser**.

2. Test the **Button** control, and verify that the text of the label changes.

    - In the http://localhost:49158/AjaxExtenderControl/AjaxExtenderControl.aspx - Windows Internet Explorer window, click **Button**.

    Note: Verify that the text of the label changes to **Button click occurred**.

3. Close Internet Explorer.

    - In the http://localhost:49158/AjaxExtenderControl/AjaxExtenderControl.aspx - Windows Internet Explorer window, click the **Close** button.

4. Add a **ScriptManager** control at the top of the form.

    - In Toolbox, on the **AJAX Extensions** tab, drag a **ScriptManager** onto the top of the form.

5. Add a **ConfirmButtonExtender** control onto the **Button1** control.

    - In Toolbox, on the **Ajax Control Toolkit** tab, drag the **ConfirmButtonExtender** control onto the **Button1** control.

6. Set the **ConfirmText** property of the **ConfirmButtonExtender** control to **Continue?**.

    - In the Properties window, click **Button1_ConfirmButtonExtender** from the drop-down list.
    - In the Properties window, set the **ConfirmText** property of the **ConfirmButtonExtender** control to **Continue?**.

7. Save the changes, and view the **AjaxExtenderControl.aspx** Web Form in the browser.

    - On the **File** menu of Visual Studio 2010, click **Save AjaxExtenderControl.aspx**.
    - In the AjaxExtenderControl.aspx window, right-click anywhere, and then click **View in Browser**.

8. Test the **Button** control, and verify that the confirmation message box appears.

- In the http://localhost:49158/AjaxExtenderControl/AjaxExtenderControl.aspx - Windows Internet Explorer window, click **Button**.
- In the **Message from webpage** message box, click **OK**.
- In the AjaxExtenderControl – Microsoft Visual Studio window, click the **Close** button.
- In the http://localhost:49158/AjaxExtenderControl/AjaxExtenderControl.aspx - Windows Internet Explorer window, click the **Close** button.
- In the Explorer window, click **Close** button.

Note: Verify that the text of the label changes to **Button click occurred**.

Additional Reading

Overview of the Ajax Control Toolkit

For more information about the Ajax Control Toolkit, you can view and experiment with the AJAX Control Toolkit. Or, learn more about the Toolkit by watching video tutorials.

For more information about the Ajax Control Toolkit, see Ajax Control Toolkit or Welcome to the ASP.NET Ajax Wiki.

Note: You can download the Ajax Control Toolkit in ASP.NET Ajax Library from the Microsoft CodePlex Web site.

Module Reviews and Takeaways

Review questions and answers

1. What is the design goal of ASP.NET Ajax?

   The design goals of ASP.NET Ajax is to increase the responsiveness of Web pages.

2. What features does the **ScriptManager** control provide?

   The **ScriptManager** control facilitates the management of script resources and features such as partial-page rendering, localization, and globalization.

3. How can you use the Ajax Control Toolkit to enhance your applications?

   The Ajax Control Toolkit provides many prebuilt controls that you can quickly add to your applications. You can also use the templates provided by the Toolkit to build your own controls and extenders.

Real-world issues and scenarios

1. You need to implement client-side functionality that is not provided by the standard ASP.NET Web server controls. What is the easiest way to do this?

   Download and use the Ajax Control Toolkit.

2. You do not have the experience of a client-side developer, but you have been asked to implement asynchronous updates to a Web Form. How do you do this?

   You add an **UpdatePanel** to the Web Form, and place the controls and elements that should be updated asynchronously in the **UpdatePanel** control.

Best practices

Mention some best practices in the context of your own business situations.

- Keep in mind that even if you implement asynchronous event processing, and thus create a responsive user interface for your users, the load on the Web server is the same. This means that each time an asynchronous callback is made, the entire page from which the callback is initiated must go through the full page life cycle on the server. This can quickly overburden the server if you are not careful with the amount of asynchronous callbacks the application uses.
- The **UpdatePanel** control should generally only be used with parts of a page, not the entire page.

Lab Review Questions and Answers

1. To implement partial-page rendering, what controls should you use?

   Answer: You should use **ScriptManager** and **UpdatePanel** controls to implement partial-page rendering.

2. What is the prerequisite step to use a control from the Ajax Control Toolkit?

   Answer: Before using a control from the Ajax Control Toolkit, you need to download the Microsoft Ajax Library, and add the controls from the Toolkit to a new tab in the Toolbox.

Module 12

Consuming Microsoft® Windows Communication Foundation Services

Contents:

Lesson 1

Overview of Windows Communication Foundation Services

Contents:

Questions and Answers

What Is Windows Communication Foundation?

Question: Which three main technologies does WCF replace?

Answer: WCF replaces XML Web services, .NET Remoting Services, and .NET Enterprise Services.

What Is a WCF Service?

Question: How do WCF services help you to make the application platform and technology independent?

Answer: WCF services allow different applications to communicate with each other, and share data and services among themselves. Other applications can also use the functionality of the WCF services.

WCF Service Communication

Question: How does communication with a WCF service occur?

Answer: All communication with a WCF service occurs through the endpoints of the service.

Uses of WCF Services

Question: What is the primary reason for using WCF services?

Answer: The primary reason for using WCF services is that you can reuse any existing functionality.

Additional Reading

What Is a WCF Service?

For more information about interoperability and integration with Windows Communication Foundation (WCF), see Interoperability and Integration.

For more information about service metadata, see Metadata.

For more information about data contracts, see Using Data Contracts.

WCF Service Communication

For more information about endpoint binding, see Windows Communication Foundation Bindings Overview.

For more information about endpoint contracts, see Designing Service Contracts.

Lesson 2

Calling Windows Communication Foundation Services

Contents:

Questions and Answers

Consuming a Windows Communication Foundation Service

Question: What is the name of the command-line tool that is used for generating WCF service proxy classes?

Answer: The command-line tool used for generating WCF service proxy classes is the ServiceModel Metadata Utility Tool, or SvcUtil.exe.

Adding a Service Reference by Using Visual Studio 2010

Question: How do you open the **Add Service Reference** dialog box?

Answer: To open the **Add Service Reference** dialog box, in Solution Explorer, right-click the name of your application, and then click **Add Service Reference**. The **Add Service Reference** dialog box displays.

Overview of Service Reference Metadata Files

Question: Which metadata file is the main reference file?

Answer: The Reference.svcmap file is the main reference file.

Demonstration: How to Programmatically Call a WCF Service

Question: What is the default name for the proxy created by using the **Add Service Reference** dialog box?

Answer: The default name for the proxy created by using the **Add Service Reference** dialog box is ServiceReference1.

Detailed Demo Steps

Demonstration: How to Programmatically Call a WCF Service

Demonstration steps

To create a proxy to call a WCF service from a Web Form, you need to perform the following steps:

1. Log on to 10267A-GEN-DEV as Student, with the password, Pa$$w0rd.
2. Open Microsoft Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
3. Open the Web application and the code-behind file for the Web Form from which you will be calling the WCF service, or create a new Web site and use the **Default** Web form of the Web site.
    - On the **File** menu of Visual Studio 2010, click **New Web Site**.
    - In the **New Web Site** dialog box, in the left pane, click either **Visual Basic** or **Visual C#**.
    - In the middle pane, click **ASP.NET Web Site**.
    - In the **Web location** list, ensure **File System** is selected, and then click **OK**.
    - In Solution Explorer, right-click **C:\Users\Admin\Documents\Visual Studio 2010\WebSites\WebSite1**, and then click **Add Service Reference**.
4. Enter the URL of the WCF service that you want to access, and create a service reference for the WCF service.
    - In the **Add Service Reference** dialog box, in the **Address** box, type **http://localhost:1112/Services/Customers.svc**, and then click **Go**.
    - In the **Services** list, expand **Customers**, and then select the **ICustomers** contract.
    - Notice the service method that displays in the Operations list, and then click **OK**.
5. Open the **Default.aspx** Web Form in Source view, remove the existing markup from the **BodyContent Content** control, and add a **Button** and **GridView** control.
    - In Solution Explorer, right-click Default.aspx, and then click **View Markup**.
    - Select the markup between the opening and closing Content tags for the **BodyContent** control, and then press the DELETE key.
    - In the Default.aspx window, point to **Toolbox**, expand **Standard**, and then double-click **Button**.
    - In Toolbox, expand **Data**, and then double-click **GridView**.
6. Create an instance of the proxy of the WCF service in the **Click** event procedure of the **Button** control.

[Visual Basic]

```
Dim customersWcfClient As New ServiceReference1.CustomersClient
```

[Visual C#]

```
ServiceReference1.CustomersClient customersWcfClient =
    new ServiceReference1.CustomersClient();
```

- In the Default.aspx window, click **Design**.
- In the Default.aspx window, double-click **Button**.
- In the Default.aspx.vb or Default.aspx.cs code window, type the following code in the **Click** event procedure of the **Button** control.

[Visual Basic]

```
Dim customersWcfClient As New ServiceReference1.CustomersClient
```

[Visual C#]

```
ServiceReference1.CustomersClient customersWcfClient =
       new ServiceReference1.CustomersClient();
```

Note: In the code example, **ServiceReference1** is the name of the WCF reference, and **CustomersClient** is the name of the WCF service.

7. Import the **System.Security.Principal** namespace.

[Visual Basic]

```
Imports System.Security.Principal
```

[Visual C#]

```
using System.Security.Principal;
```

- In the Default.aspx.vb or Default.aspx.cs code window, add the following at the top of the code file.

[Visual Basic]

```
Imports System.Security.Principal
```

[Visual C#]

```
using System.Security.Principal;
```

8. Impersonate a user account that can be impersonated to allow access to secure resources from the WCF service by using Windows credentials.

[Visual Basic]

```
customersWcfClient.ClientCredentials.Windows.ClientCredential.UserName = "10267A-
GEN-DEV\Student"
customersWcfClient.ClientCredentials.Windows.ClientCredential.Password =
"Pa$$w0rd"
```

```
customersWcfClient.ClientCredentials.Windows.AllowedImpersonationLevel =
TokenImpersonationLevel.Impersonation
```

[Visual C#]

```
customersWcfClient.ClientCredentials.Windows.ClientCredential.UserName = "10267A-
GEN-DEV\\Student";
customersWcfClient.ClientCredentials.Windows.ClientCredential.Password =
"Pa$$w0rd";
customersWcfClient.ClientCredentials.Windows.AllowedImpersonationLevel =
TokenImpersonationLevel.Impersonation;
```

- In the Default.aspx.vb or Default.aspx.cs code window, append the following code to the **Click** event procedure of the **Button** control.

[Visual Basic]

```
customersWcfClient.ClientCredentials.Windows.ClientCredential.UserName = "10267A-
GEN-DEV\Student"
customersWcfClient.ClientCredentials.Windows.ClientCredential.Password =
"Pa$$w0rd"
customersWcfClient.ClientCredentials.Windows.AllowedImpersonationLevel =
TokenImpersonationLevel.Impersonation
```

[Visual C#]

```
customersWcfClient.ClientCredentials.Windows.ClientCredential.UserName = "10267A-
GEN-DEV\\Student";
customersWcfClient.ClientCredentials.Windows.ClientCredential.Password =
"Pa$$w0rd";
customersWcfClient.ClientCredentials.Windows.AllowedImpersonationLevel =
TokenImpersonationLevel.Impersonation;
```

9. Call the **GetCountries** method of the WCF service, requesting countries starting with the letter D.

[Visual Basic]

```
GridView1.DataSource = customersWcfClient.GetCountries("D")
GridView1.DataBind()
```

[Visual C#]

```
GridView1.DataSource = customersWcfClient.GetCountries("D");
GridView1.DataBind();
```

- In the Default.aspx.vb or Default.aspx.cs code window, append the following code to the **Click** event procedure of the **Button** control.

[Visual Basic]

```
GridView1.DataSource = customersWcfClient.GetCountries("D")
GridView1.DataBind()
```

[Visual C#]

```
GridView1.DataSource = customersWcfClient.GetCountries("D");
GridView1.DataBind();
```

10. Close the WCF service client.

[Visual Basic]

```
customersWcfClient.Close()
```

[Visual C#]

```
customersWcfClient.Close();
```

- In the Default.aspx.vb or Default.aspx.cs code window, append the following code to the **Click** event procedure of the **Button** control.

[Visual Basic]

```
customersWcfClient.Close()
```

[Visual C#]

```
customersWcfClient.Close();
```

11. Build and run the Microsoft ASP.NET Web application.
    - In Visual Studio 2010, on the **Debug** menu, click **Start Without Debugging**.
    - In the Windows Internet Explorer® window, click **Button**.
    - In the http://localhost:49157/WebSite1/Default.aspx - Windows Internet Explorer window and WebSite1 - Microsoft Visual Studio window, click the **Close** button.

Note: Verify that the customers' details display on the Web Form by using the WCF service.

Additional Reading

Consuming a Windows Communication Foundation Service

For more information about the ServiceModel Metadata utility tool, see ServiceModel Metadata Utility Tool (Svcutil.exe).

Lesson 3

Working with WCF Data Services

Contents:

Questions and Answers

Overview of WCF Data Services

Question: What is the goal of the WCF Data Services framework?

Answer: The goal of the WCF Data Services framework is to facilitate the creation of flexible data services that are naturally integrated with the Web.

Creating WCF Data Services

Question: What is the file extension for a WCF Data Service?

Answer: The file extension for a WCF Data Service is .svc.

Demonstration: How to Add WCF Data Services to an Existing Web Site

Question: Which template will you use to add a WCF Data Service to a project in Visual Studio 2010?

Answer: You can use the WCF Data Service item template to add a WCF Data Service item to a project in Visual Studio 2010.

Examining a WCF Data Service

Question: Which attribute in the markup or host file refers to the code-behind file or the class that defines the data service?

Answer: The **Service** attribute in the markup or host file refers to the code-behind file, or the class that defines the data service.

Detailed Demo Steps

Demonstration: How to Add WCF Data Services to an Existing Web Site

Demonstration steps

To add WCF Data Services to an existing Web site, perform the following steps:

1. Log on to 10267A-GEN-DEV as **Student**, with the password, **Pa$$w0rd**.
2. Open Microsoft Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
3. Open the **CustomerManagement** solution from the D:\Demofiles\M12\CS or D:\Demofiles\M12\VB folder.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **Open Project**.
    - In the **Open Project** dialog box, in the **File name** box, type either **D:\Demofiles\M12\VB\CustomerManagement.sln** or **D:\Demofiles\M12\CS\CustomerManagement.sln**, and then click **Open**.
4. Add a new WCF Data Service named **Countries**.
    - In Solution Explorer, right-click either the **D:\Demofiles\M12\VB\CustomerManagement\ or D:\Demofiles\M12\CS\CustomerManagement** Web site, and then click **Add New Item**.
    - In the **Add New Item – D:\Demofiles\M12\VB\CustomerManagement\ or Add New Item – D:\Demofiles\M12\CS\CustomerManagement** dialog box, in the middle pane, click WCF Data Service.
    - In the **Name** box, type **Countries.svc**, and then click **Add**.
5. Add the existing EDM namespace and class name to the declaration of the generic **Countries** class, **CustomerManagementModel.Entities**.

[Visual Basic]

```
Public Class Countries
       Inherits DataService(Of CustomerManagementModel.Entities)
```

[Visual C#]

```
public class Countries : DataService<CustomerManagementModel.Entities>
```

- In the App_Code/Countries.vb or App_Code/Countries.cs window, modify the following class definition code from,

[Visual Basic]

```
Public Class Countries
       ' TODO: replace [[class name]] with your data class name
       Inherits DataService(Of [[class name]])
```

[Visual C#]

```
public class Countries: DataService< /* TODO: put your data source class name here
*/ >
```

to:

[Visual Basic]

```
Public Class Countries
       Inherits DataService(Of CustomerManagementModel.Entities)
```

[Visual C#]

```
public class Countries : DataService<CustomerManagementModel.Entities>
```

6. Allow read access to the **Countries** and **Customers** entities of the EDM by using the **IDataServiceConfiguration.SetEntitySetAccessRule** method in the **Shared** or **static InitializeService** method of the **DataService** class.

[Visual Basic]

```
config.SetEntitySetAccessRule("Countries", EntitySetRights.AllRead)
config.SetEntitySetAccessRule("Customers", EntitySetRights.AllRead)
```

[Visual C#]

```
config.SetEntitySetAccessRule("Countries", EntitySetRights.AllRead);
config.SetEntitySetAccessRule("Customers", EntitySetRights.AllRead);
```

- In the App_Code/Countries.vb or App_Code/Countries.cs window, replace the following code of the class definition,

[Visual Basic]

```
' TODO: set rules to indicate which entity sets and service operations are
visible, updatable, etc.
' Examples:
' config.SetEntitySetAccessRule("MyEntityset", EntitySetRights.AllRead)
' config.SetServiceOperationAccessRule("MyServiceOperation",
ServiceOperationRights.All)
```

[Visual C#]

```
// TODO: set rules to indicate which entity sets and service operations are
visible, updatable, etc.
// Examples:
// config.SetEntitySetAccessRule("MyEntityset", EntitySetRights.AllRead);
// config.SetServiceOperationAccessRule("MyServiceOperation",
ServiceOperationRights.All);
```

with:

[Visual Basic]

```
config.SetEntitySetAccessRule("Countries", EntitySetRights.AllRead)
config.SetEntitySetAccessRule("Customers", EntitySetRights.AllRead)
```

[Visual C#]

```
config.SetEntitySetAccessRule("Countries", EntitySetRights.AllRead);
config.SetEntitySetAccessRule("Customers", EntitySetRights.AllRead);
```

7. Save and close the Countries code-behind file.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click either **Save App_Code/Countries.vb** or **Save App_Code/Countries.cs**.
    - In the App_Code/Countries.vb or App_Code/Countries.cs window, click the **Close** button.
8. Build the solution.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **Build** menu, click **Build Solution**.
9. View the service in the browser.
    - In Solution Explorer, right-click **Countries.svc**, and then click **View in Browser**.

> **Note:** Notice that the XML returned from the service contains both **Countries** and **Customers** entities.

10. View the Countries entity by browsing to the URL **http://localhost:1111/CustomerManagement/Countries.svc/Countries** or **http://localhost:1110/CustomerManagement/Countries.svc/Countries**.
    - In the Address bar of Internet Explorer, type **http://localhost:1111/CustomerManagement/Countries.svc /Countries** or **http://localhost:1110/CustomerManagement/Countries.svc /Countries**, and then press ENTER.

> **Note:** Notice that the XML returned from the service contains all the Countries from the data model. Notice how it looks similar to the way Dynamic Data applies routing, by appending the name of the entity to the URL.

11. View the Customers entity by browsing to the URL **http://localhost:1111/CustomerManagement/Countries.svc/Customers** or **http://localhost:1110/CustomerManagement/Countries.svc/Customers**.
    - In the Address bar of Internet Explorer, type **http://localhost:1111/CustomerManagement/Countries.svc /Customers** or **http://localhost:1110/CustomerManagement/Countries.svc /Customers**, and then press ENTER.

> **Note:** Notice that the XML returned from the service contains all the Customers from the data model.

12. View the information for the last name, **Abercrombie**, by browsing to the URL, **http://localhost:1111/CustomerManagement/Countries.svc/Customers?$filter=LastName eq 'Abercrombie'** or **http://localhost:1110/CustomerManagement/Countries.svc/Customers?$filter=LastName eq 'Abercrombie'**.
    - In the Address bar of Internet Explorer, type **http://localhost:1111/CustomerManagement/Countries.svc**

/Customers?$filter=LastName eq 'Abercrombie' or http://localhost:1110/CustomerManagement/Countries.svc /Customers?$filter=LastName eq 'Abercrombie', and then press ENTER.

> **Note:** Notice that the XML returned from the service contains a single customer, **Kim Abercrombie**. Mention that the **$filter** operator is used with the **LastName** column. That column cannot be queried directly, because it is not the primary key of the entity.

13. View the countries for which the **InternetTLD** starts with less than B by using the URL, **http://localhost:1111/CustomerManagement/Countries.svc /Countries?$filter=InternetTLD lt 'B'** or **http://localhost:1110/CustomerManagement/Countries.svc /Countries?$filter=InternetTLD lt 'B'**.

    - In the Address bar of Internet Explorer, type **http://localhost:1111/CustomerManagement/Countries.svc /Countries?$filter=InternetTLD lt 'B'** or **http://localhost:1110/CustomerManagement/Countries.svc /Countries?$filter=InternetTLD lt 'B'**, and then press ENTER.

> **Note:** Notice that the XML returned from the service contains all countries for which the **InternetTLD** starts with less than B. Mention that the **$filter** operator is used with the Name column. That column cannot be queried directly, because it is not the primary key of the entity.

14. Close Internet Explorer.

    - In the http://localhost:1111/CustomerManagement/Countries.svc /Countries?$filter=InternetTLD lt 'B' – Windows Internet Explorer or http://localhost:1110/CustomerManagement/Countries.svc /Countries?$filter=InternetTLD lt 'B' – Windows Internet Explorer window, click the **Close** button.

15. Close Visual Studio 2010.

    - In the CustomerManagement – Microsoft Visual Studio window, click the **Close** button.

Additional Reading

Creating WCF Data Services

To watch a video on building a simple WCF Data Service over a Relational Database, see How Do I: Getting Started with WCF Data Services over a Relational Database.

To watch a video on integrating WCF Data Services with ASP.NET Ajax support, see ADO.NET Data Services with ASP.NET AJAX Support.

For more information about WCF Data Services, see WCF Data Services.

Demonstration: How to Add WCF Data Services to an Existing Web Site

For more information about the URI format for addressing WCF Data Services Resources, see Addressing Resources (WCF Data Services).

For more information about WCF Data Services query operators, see Query Operators (WCF Data Services).

Examining a WCF Data Service

For more information about hosting a WCF Data Service, see Hosting the Data Service (WCF Data Services).

Module Reviews and Takeaways

Review questions and answers

1. Does a WCF service have a user interface?

   No, WCF services do not have user interfaces. Users can interact with WCF services directly by using the description page, if they know the URL and have the authorization, but this is normally used for testing only.

2. How do you access a WCF service from a Web Form?

   You can create a service reference to the WCF service, which then creates a proxy. On the Web Form, in an event procedure, instantiate the proxy, and call the methods of the WCF service.

3. What is the name of the dialog box that you can use to add a reference to a WCF service in Visual Studio 2010?

   You can use the **Add Service Reference** dialog box in Visual Studio 2010 to add a reference to a WCF service.

Real-world issues and scenarios

1. You need to expose data to a client that does not have access to your intranet. What is the easiest way to implement this?

   You should create and publish a WCF service.

2. You need to expose data internally to a variety of clients, including Web applications, Windows Forms, and console applications. What is the best way to implement this?

   Create and publish a WCF service, because it will allow different types of hosting and different types of connections, all depending on the actual connectivity requirements of the consuming application.

Best practices

Mention some best practices in the context of your own business situations.

- When you expose data externally through a service, or you do not know who will be accessing your service, you should always return any data using data types that map to simple data types, such as strings and integers, which can be used directly with XML.

- When you add a reference to a WCF service, always give the service reference a name that spells out what the service reference refers to. Do not use the default names, such as **ServiceReference1**.

Lab Review Questions and Answers

1. What are the elements in the Customers.discomap file?

   Answer: The Customers.discomap file contains the contractRef and soap elements that link to other documents, including the WSDL and documentation on the server.

2. How do you define data source to the **CustomersGridView** control?

   Answer: You need to call the **GetCountries** Web Service method, and assign the return value to a **CustomersGridView** control as the data source.

Module 13

Managing State in Web Applications

Contents:

Lesson 1

State Management

Contents:

Questions and Answers

What Is State Management?

Question: If you do not use state management, what happens to user details after they are provided by a user on an initial login page?

Answer: They are discarded as if they were never entered in the first place.

Types of State Management

Question: What are the differences between server-side and client-side state?

Answer: The differences between server-side and client-side state are performance and security. Server-side state is more secure, but client-side state offers better performance.

Server-Side State Management

Question: How does ASP.NET store Application state and Session state?

Answer: ASP.NET stores Application state in memory. ASP.NET by default stores Session state in memory, but can optionally store Session state in a State Server or a database.

Demonstration: How to Use Session State

Question: Why it is good practice to cast session variables?

Answer: Session variables are of type **System.Object**, and therefore should be cast to the original type. This means that the session variables should be cast to the variables that were originally saved to the Session state.

Client-Side State Management

Question: Why are cookies less secure than server-side state management options?

Answer: Cookies are less secure than server-side state management options because they are stored as plain text on the client computer, and can be tampered with.

Detailed Demo Steps

Demonstration: How to Use Session State

Demonstration steps

1. Log on to 10267A-GEN-DEV as **Student**, with the password, **Pa$$w0rd**.
2. Open Microsoft Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
3. Open the **SessionState** solution from either the D:\Demofiles\M13\VB or D:\Demofiles\M13\CS folder.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **Open Project**.
    - In the **Open Project** dialog box, in the **File name** box, type either **D:\Demofiles\M13\VB\SessionState.sln** or **D:\Demofiles\M13\CS\SessionState.sln**, and then click **Open**.
4. Open the Global.asax file, and add a variable named **PageHits**, with a value of **0**, to Session state upon session start.
    - In Solution Explorer, right-click **Global.asax**, and then click **Open**.
    - In the Global.asax window, append the following code to the **Session_Start** method.

[Visual Basic]

```
' Save session variable
Session("PageHits") = 0
```

[Visual C#]

```
// Save session variable
Session["PageHits"] = 0;
```

5. Save and close the Global.asax file.
    - In the SessionState – Microsoft Visual Studio window, on the **File** menu, click **Save Global.asax**.
    - In the Global.asax window, click the **Close** button.
6. In the **Page_Load** event handler of the master page Site.master, create a local variable named **numPageHits**, of type **long**, to hold the number of page hits, and initialize the variable with a value of **0**.
    - In Solution Explorer, right-click **Site.master**, and then click **View Code**.
    - In the Site.master.vb window, select **(Page Events)** from the **Class Name** list, and on the **Method Name** list, click **Load**.
    - In the Site.master.vb or Site.master.cs window, type the following code in the **Page_Load** event handler.

[Visual Basic]

```
Dim numPageHits As Long = 0
```

[Visual C#]

```
long numPageHits = 0;
```

7. In the **Page_Load** event handler, check if the **PageHits** Session variable exists, and if so, assign the value to **numPageHits** local variable, casting it to data type **Long** or **long**.

    - In the Site.master.vb or Site.master.cs window, append the following code in the **Page_Load** event handler, after the declaration and initialization of the **numPageHits** variable.

[Visual Basic]

```
' Check if Session variable exists
If Not Session("PageHits") Is Nothing Then
       numPageHits = Long.Parse(Session("PageHits").ToString())
End If
```

[Visual C#]

```
// Check if Session variable exists
if (Session["PageHits"] != null)
{
       numPageHits = long.Parse(Session["PageHits"].ToString());
}
```

> **Note:** The Session variable must be converted, such as by using the **Parse** method of the integer data type to which you are casting, or by doing a direct cast (**long**) or (**int**).

8. In the **Page_Load** event handler, update the value of the **numPageHits** variable, and save the value to the **PageHits** Session variable.

    - In the Site.master.vb or Site.master.cs window, append the following code in the **Page_Load** event handler after the code that checks for the session variable existence.

[Visual Basic]

```
numPageHits += 1
Session("PageHits") = numPageHits
```

[Visual C#]

```
numPageHits += 1;
Session["PageHits"] = numPageHits;
```

9. In the **Page_Load** event handler, output the value of the **numPageHits** variable to the **Debug** pane of the Output window by using the **WriteLine** method.

    - In the Site.master.vb or Site.master.cs window, append the following code in the **Page_Load** event handler, after the code that saves the session variable.

[Visual Basic]

```
System.Diagnostics.Debug.WriteLine("Number of page hits = " &
numPageHits.ToString())
```

[Visual C#]

```
System.Diagnostics.Debug.WriteLine("Number of page hits = " +
numPageHits.ToString());
```

10. Save and close the Site.master.vb or Site.master.cs file.
    - In the SessionState – Microsoft Visual Studio window, on the **File** menu, click either **Save Site.master.vb** or **Save Site.master.cs**.
    - In the Site.master.vb or Site.master.cs window, click the **Close** button.
11. Run the Web site in the Debug mode.
    - In the SessionState – Microsoft Visual Studio window, on the **Debug** menu, click **Start Debugging**.
    - In the Home Page - Windows Internet Explorer window, click **About**.
    - In the About Us - Windows Internet Explorer window, click the **Close** button.
12. Verify the output in the **Debug** pane of the Output window.
    - In the SessionState – Microsoft Visual Studio window, click **Output** at the bottom of the window.
    - In the **Show output from** list, click **Debug**.
    - In the SessionState – Microsoft Visual Studio window, click the **Close** button.

Note: If necessary, scroll up to see the output from the **Page_Load** event handler.

Demonstration: How to Use Application State to Display the Visitors Counter

Demonstration steps

1. Log on to 10267A-GEN-DEV as **Student**, with the password, **Pa$$w0rd**.
2. Open Microsoft Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
3. Open an existing Web site.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **Open Project**.
    - In the **Open Project** dialog box, in the **File name** box, type either **D:\Demofiles\M13\VB\CustomerManagement.sln** or **D:\Demofiles\M13\CS\CustomerManagement.sln**, and then click **Open**.
4. Add a variable named **Visitors**, with a value of **0**, to the Application state.
    - In Solution Explorer, right-click **Global.asax**, and then click **Open**.

- In the **Global.asax** window, append the following code to the **Application_Start** method, after the code that calls the **Register** method.

[Visual Basic]

```
' Save application variable
Application("Visitors") = 0
```

[Visual C#]

```
// Save application variable
Application["Visitors"] = 0;
```

5. Save the **Global.asax** file.
   - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click **Save Global.asax**.
6. Add a **div** element with a **class** attribute set to the value of **footer** to the **MainContent** control of the **default** Web Form.
   - In Solution Explorer, right-click **Default.aspx**, and then click **View Markup**.
   - In the Default.aspx window, type the following markup within the **MainContent** control.

```
<div class="footer">
</div>
```

7. Add a **Literal** control named **VisitorLiteral**, to the **div** element.
   - In the Default.aspx window, type the following markup within the **div** element.

```
<asp:Literal ID="VisitorLiteral" runat="server" />
```

8. Save and close the Default Web Form.
   - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click **Save Default.aspx**.
   - In the Default.aspx window, click the **Close** button.
9. In the **Page_Load** event handler of the **Default** Web Form, create a local variable named **numVisitors**, of type **long**, to hold the number of visitors and initialize the variable with a value of **0**.
   - In Solution Explorer, right-click **Default.aspx**, and then click **View Code**.
   - In the Default.aspx.vb window, in the **Class Name** list, click (**Page Events**), and in the **Method Name** list, click **Load**.

> **Note:** In Microsoft Visual C#®, the **Page** event handlers are created automatically.

   - In the Default.aspx.vb or Default.aspx.cs window, type the following code in the **Page_Load** event handler.

[Visual Basic]

```
Dim numVisitors As Long = 0
```

[Visual C#]

```
long numVisitors = 0;
```

10. In the **Page_Load** event handler, check if the Visitors Application variable exists, and if so, assign the value to the **numVisitors** local variable, casting it to data type **long**.

    - In the Default.aspx.vb or Default.aspx.cs window, type the following code in the **Page_Load** event handler, after the declaration and initialization of the **numVisitors** variable.

> **Note:** The Visitors Application variable must be converted by using the Parse method of the integer data type.

[Visual Basic]

```
' Check if Application variable exists
If Application("Visitors") Is Nothing Then
       numVisitors = Long.Parse(Application("Visitors").ToString())
End If
```

[Visual C#]

```
// Check if Application variable exists
if (Application["Visitors"] != null)
{
       numVisitors = long.Parse(Application["Visitors"].ToString());
}
```

11. In the **Page_Load** event handler, assign the text **"Number of visitors: "**, and the value of the **numVisitors** variable to the Text property of the VisitorLiteral control.

    - In the Default.aspx.vb or Default.aspx.cs window, type the following code in the **Page_Load** event handler, after the code that checks for the application variable existence.

[Visual Basic]

```
VisitorLiteral.Text = "Number of visitors: " & numVisitors.ToString()
```

[Visual C#]

```
VisitorLiteral.Text = "Number of visitors: " + numVisitors.ToString();
```

12. Save and close the Default.aspx.vb or Default.aspx.cs file.

    - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click either **Save Default.aspx.vb** or Save **Default.aspx.cs**.
    - In the Default.aspx.vb or Default.aspx.cs window, click the **Close** button.

13. Each time a new user session is started, increment the **Visitors** application variable by 1.

    - In the Global.asax window, type the following code in the **Session_Start** event handler, after the code that checks for the application variable existence.

[Visual Basic]

```
' Increment Visitors counter
Application("Visitors") =
       Long.Parse(Application("Visitors").ToString()) + 1
```

[Visual C#]

```
// Increment Visitors counter
Application["Visitors"] =
       long.Parse(Application["Visitors"].ToString()) + 1;
```

14. Run the application.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.

    Note: Notice that the Number of visitors is set to **1**.
15. Close Internet Explorer.
    - In the Contoso Customer Management – Windows Internet Explorer window, click the **Close** button.
16. Run the application.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.

    Note: Notice that the Number of visitors is set to **2**.
17. Close Internet Explorer.
    - In the Contoso Customer Management – Windows Internet Explorer window, click the **Close** button.
18. Close Visual Studio 2010.
    - In the CustomerManagement – Microsoft Visual Studio window, click the **Close** button.

Additional Reading

What Is State Management?

For more information about the overview of ASP.NET state management, see ASP.NET State Management Overview.

Server-Side State Management

For more information about saving application variables, see How to: Save Values in Application State.

For more information about reading application variables, see How to: Read Values from Application State.

For more information about the Aspnet_regsql command-line tool, see ASP.NET SQL Server Registration Tool (Aspnet_regsql.exe).

For more information about the HttpApplicationState class, see HttpApplicationState Class.

For more information about the ASP.NET state management recommendations, see ASP.NET State Management Recommendations.

For more information about the HttpSessionState class, see HttpSessionState Class.

For more information about the session-state modes, see Session-State Modes.

For more information about how to cache application data, see Caching Application Data.

For more information about how to use output caching, see Caching ASP.NET Pages.

Client-Side State Management

For more information about cookies, see ASP.NET Cookies Overview.

For more information about View state, see ASP.NET View State Overview.

For more information about the comparison of View state and control state, see Control State vs. View State Example.

For more information about the HiddenField control, see HiddenField Web Server Control Overview.

Lesson 2

ASP.NET Profiles

Contents:

Questions and Answers

What Are ASP.NET Profiles?

Question: Which property of the **HttpContext** class makes ASP.NET Profiles available anywhere in your Web application?

Answer: The Profile property of the HttpContext class makes ASP.NET Profiles available anywhere in your Web application.

Using ASP.NET Profiles

Question: What is the name of the default ASP.NET profile provider?

Answer: The default ASP.NET profile provider name is System.Web.Profile.SqlProfileProvider.

Additional Reading

What Are ASP.NET Profiles?

For more information about the properties of ASP.NET, see ASP.NET Profile Properties Overview.

Using ASP.NET Profiles

For more information about how to implement a profile provider, see Implementing a Profile Provider.

For more information about the SqlProfileProvider class, see SqlProfileProvider Class.

For more information about the profile element, see profile Element (ASP.NET Settings Schema).

Lesson 3

ASP.NET Caching

Contents:

Questions and Answers

What Is ASP.NET Caching?

Question: Which types of caching does ASP.NET provide?

Answer: ASP.NET provides application caching and page output caching.

Using ASP.NET Caching

Question: What are the two method names for adding items to the application cache?

Answer: The two method names for adding items to the application cache are Add and Insert.

Demonstration: How to Implement Output Caching

Question: Why would you use ASP.NET page output caching?

Answer: ASP.NET page output caching reduces the load on the Web server, and helps improve performance.

Detailed Demo Steps

Demonstration: How to Implement Caching

Demonstration steps

1. Log on to 10267A-GEN-DEV as **Student**, with the password, **Pa$$w0rd**.
2. Open Microsoft Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
3. Open an existing Web site.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **Open Project**.
    - In the **Open Project** dialog box, in the **File name** box, type either **D:\Demofiles\M13\VB\CustomerManagement.sln** or **D:\Demofiles\M13\CS\CustomerManagement.sln**, and then click **Open**.
4. In the **GetCountriesButton_Click** event handler of the **Services/Customers** Web Form, create a local variable named **countryDataTable**, of type **System.Data.DataTable**, to hold the countries retrieved from the database, and initialize the variable to **Nothing** or **null**.
    - In Solution Explorer, expand **Services**, right-click **Customers.aspx**, and then click **View Code**.
    - In the Services/Customers.aspx.vb or Services/Customers.aspx.cs window, add the following code to the **GetCountriesButton_Click** event handler, after the method signature.

[Visual Basic]

```
Dim countryDataTable As System.Data.DataTable = Nothing
```

[Visual C#]

```
System.Data.DataTable countryDataTable = null;
```

5. Assign the **Countries** cache item value to the **countryDataTable** local variable, casting it to the data type **System.Data.DataTable**.
    - Type the following code in the **GetCountriesButton_Click** event handler, after the declaration of the **countryDataTable** variable.

[Visual Basic]

```
' Retrieve DataTable from cache
countryDataTable = CType(Cache("Countries"), System.Data.DataTable)
```

[Visual C#]

```
// Retrieve DataTable from cache
countryDataTable =
       (System.Data.DataTable) Cache["Countries"];
```

6. Check whether an item has been retrieved from the cache by comparing the **countryDataTable** variable with a **null** value, in an **If** or **if** construct.

    - Type the following code after the code that retrieves the **Countries** cache item.

[Visual Basic]

```
' Does cached item exist?
If countryDataTable Is Nothing Then
End If
```

[Visual C#]

```
// Does cached item exist?
if (countryDataTable == null)
{
}
```

7. If the item is retrieved from the cache, assign the value of the **countryDataTable** local variable to the **DataSource** property of the **CustomersGridView** control.

    - After the **If** or **if** construct, type the following code.

[Visual Basic]

```
' Set GridView DataSource
CustomersGridView.DataSource = countryDataTable
```

[Visual C#]

```
// Set GridView DataSource
CustomersGridView.DataSource = countryDataTable;
```

8. If an item is not retrieved from the cache, move the existing line of code that declares and instantiates the local **customersService** variable, from after the **If** or **if** construct.

    - Locate and cut the following code.

[Visual Basic]

```
Dim customersService As New CustomersClient()
```

[Visual C#]

```
CustomersClient customersService = new CustomersClient();
```

    - In the **If** or **if** construct, paste the copied code.

9. If an item is not retrieved from the cache, assign the result of the call to the **GetCountries** Windows Communication Foundation (WCF) service method, to the local **countryDataTable** variable.

    - In the **If** or **if** construct, append the following code after the line that declares and instantiates the local **customersService** variable.

[Visual Basic]

```
' Retrieve DataTable from WCF Service
```

```
countryDataTable = customersService.GetCountries(StartingLettersTextBox.Text)
```

[Visual C#]

```
// Retrieve DataTable from WCF Service
countryDataTable = customersService.GetCountries(StartingLettersTextBox.Text);
```

10. Add a new variable named **Countries** to the cache object, and assign the value of the local **countryDataTable** variable.
    - In the **If** or **if** construct, append the following code after the line that assigns the result of the call to the **GetCountries** WCF service method.

[Visual Basic]

```
' Save DataTable to cache
Cache("Countries") = countryDataTable
```

[Visual C#]

```
// Save DataTable to cache
Cache["Countries"] = countryDataTable;
```

11. Delete the existing line of code that assigns the result of the call to the **GetCountries** Web service method, to the **DataSource** property of the **CustomersGridView** control.
    - Delete the following line of code.

[Visual Basic]

```
CustomersGridView.DataSource =
CustomersService.GetCountries(StartingLettersTextBox.Text)
```

[Visual C#]

```
CustomersGridView.DataSource =
CustomersService.GetCountries(StartingLettersTextBox.Text);
```

12. Save the changes to the Customers.aspx.vb or Customers.aspx.cs file.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click either **Save Services/Customers.aspx.vb** or **Save Services/Customers.aspx.cs**.

Demonstration: How to Implement Output Caching

Demonstration steps

1. Log on to 10267A-GEN-DEV as **Student**, with the password, **Pa$$w0rd**.
2. Open Microsoft Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
3. Open the **OutputCaching** solution from either the D:\Demofiles\M13\VB or D:\Demofiles\M13\CS folder.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **Open Project**.

- In the **Open Project** dialog box, in the **File name** box, type either **D:\Demofiles\M13\VB\OutputCaching.sln** or **D:\Demofiles\M13\CS\OutputCaching.sln**, and then click **Open**.

Note: Notice the Global.asax file, in which an application variable is stored upon application start up, and is incremented upon session start.

- In Solution Explorer, right-click **Global.asax**, and then click **Open**.

Note: View the code for the Application_Start and Session_Start procedures.

- In the Global.asax window, click the **Close** button.

Note: Notice the Default.aspx.vb or Default.aspx.cs file in the **Page_Load** event handler, in which an application variable was retrieved and displayed on the page.

- In Solution Explorer, right-click **Default.aspx**, and then click **View Code**.

Note: View the code for the **Page_Load** event handler.

- In the Default.aspx.vb or Default.aspx.cs window, click the **Close** button.

4. Run the Web site, and copy the URL from Internet Explorer.
   - In the OutputCaching - Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.

Note: Notice that the number of visitors is 1.

   - Copy the URL from the Address bar.
5. Open another instance of Internet Explorer, and browse to the copied URL.
   - On the **Start** menu of 10267A-GEN-DEV, click **Internet Explorer**.
   - In the Address bar of Internet Explorer, paste the copied URL, and then press ENTER.

Note: Notice that the number of visitors is 2.

6. Close the two instances of Internet Explorer.
   - In the http://localhost:49116/OutputCaching/Default.aspx - Windows Internet Explorer window, click the **Close** button.
   - In the http://localhost:49116/OutputCaching/Default.aspx - Windows Internet Explorer window, click the **Close** button.
7. Stop the ASP.NET Development Server.
   - In the notification area of Windows Task Bar, right-click **ASP.NET Development Server - Port 49116 or ASP.NET Development Server - Port 49117**, and then click **Stop**.
8. Set the cacheability of the Default Web Form by adding the OutputCache directive with a duration of 15 seconds, without being dependent on query string or form parameters.
   - In Solution Explorer, right-click **Default.aspx**, and then click **View Markup**.
   - In the Default.aspx window, add the following markup below the **Page** directive.

```
<%@ OutputCache Duration="15" VaryByParam="None" %>
```

- In the OutputCaching – Microsoft Visual Studio window, on the **File** menu, click **Save Default.aspx**.

9. Run the Web site.
    - In the OutputCaching - Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.

> **Note:** Notice that the number of visitors is 1.

    - Copy the URL from the Address bar.
10. Open another instance of Internet Explorer, and browse to the copied URL.
    - On the **Start** menu, click **Internet Explorer**.
    - In the Address bar of Internet Explorer, paste the copied URL, and then press ENTER.

> **Note:** Notice the number of visitors is still 1, because the page is cached.

11. Close the two instances of Internet Explorer.
    - In the http://localhost:49116/OutputCaching/Default.aspx - Windows Internet Explorer window, click the **Close** button.
    - In the http://localhost:49116/OutputCaching/Default.aspx - Windows Internet Explorer window, click the **Close** button.
12. Open a new instance of Internet Explorer.
    - On the **Start** menu, click **Internet Explorer**.
13. Wait for 15 seconds, and then browse to the copied URL.
    - Wait for 15 seconds, and then in the Address bar of Internet Explorer, paste the copied URL, and then press ENTER.
    - Notice the number of visitors is now 2, because the page had expired in the cache.
14. Close all open windows.

Additional Reading

What Is ASP.NET Caching?

For more information about ASP.NET Caching, see ASP.NET Caching Overview.

Using ASP.NET Caching

For more information about saving application cache items, see How to: Add Items to the Cache.

For more information about how to read application cache items, see How to: Retrieve Values of Cached Items.

For more information about caching pages for which different versions are created based on the request, see Caching Multiple Versions of a Page.

For more information about implementing the application cache for a Web application, see Cache Class.

For more information about the relative priority of items stored in the cache object, see CacheItemPriority Enumeration.

For more information about notification when an item is removed from the application cache, see How to: Notify an Application When an Item Is Removed from the Cache.

For more information about the cacheability setting you need for a page during design, see How to: Set the Cacheability of an ASP.NET Page Declaratively.

For more information about how to write custom logic to check whether the page is valid, see How to: Check the Validity of a Cached Page.

For more information about caching portions of the page, see Caching Portions of an ASP.NET Page.

For more information about removing a page from the output cache when a file

changes, see How to: Cache Page Output with File Dependencies.

For more information about how to remove a page from the output cache when another item in the cache is removed, see How to: Cache Page Output with Cache Key Dependencies.

Module Reviews and Takeaways

Review questions and answers

1. What are the types of state management?

   The types of state management are server-side state management, and client-side state management. Server-side state management uses server resources to store state information, and client-side state management stores page information without using server resources.

2. Which command will you use for storing profiles in a SQL Server database?

   ```
   aspnet_regsql -S 10267A-GEN-DEV -E –A p
   ```

3. Which class is used for customization of caching items?

   The Cache class allows customization of how items are cached, and for how long they are cached.

Real-world issues and scenarios

1. You need to make a value available to all sessions of your Web application. What is the easiest way to implement this?

   You should create and store an Application variable.

2. You need to make a value available throughout a user session of your Web application. What is the best way to implement this?

   You should create and store a Session variable.

3. You need to save values on a Web Form between round trips. What is the best way to implement this without exposing the value to the user?

   You should create and store a variable in ViewState.

Best practices

Mention some best practices in the context of your own business situations.

- Follow a naming convention for the variables that you save in View state, Session state, and Application state, as well as objects saved to the Cache.
- Disable Session state when you are not using it.
- Disable View state for a Web Form, when possible.
- Use caching for pages that change infrequently.

Lab Review Questions and Answers

1. How will you disable View state for a page?

   Answer: To disable the View state for a page, you need to set the EnableViewState attribute with a false value.

2. How will you disable View state for a page, but enable it for a control on that page?

   Answer: You need to set the EnableViewState attribute with a true value, the ViewStateMode attribute with a false value for the Page directive, and then set the ViewStateMode attribute with a true value for the control.

3. How will you test the cache storage?

   Answer: To test the cache storage, you can set a breakpoint on the code that retrieves the Countries item from the Cache object, and then run the Web application in the debug mode.

Module 14

Configuring and Deploying a Microsoft® ASP.NET Web Application

Contents:

Lesson 1

Configuring an ASP.NET Web Application

Contents:

Questions and Answers

Overview of Configuration Files

Question: How many web.config files can you use in each folder of a Web application?

Answer: You can use only one web.config file in each folder of a Web application.

Overview of a Machine.config File

Question: How many machine.config files does a Web server contain?

Answer: A Web server contains only one machine.config file for each version of the .NET Framework installed.

Overview of a Web.config File

Question: How can you have more than one web.config file for a specific Web application?

Answer: You can have a web.config file in the root folder of the Web application, and a web.config file in a subfolder.

Overview of ASP.NET Configuration Inheritance

Question: How will you ensure that there is no conflict between the settings on virtual and physical directories?

Answer: Configuration settings for virtual directories are independent of physical directory structure, and you must organize virtual directories carefully to avoid configuration problems. You should not nest virtual directories; however, if you do, use only one web.config file.

Overview of the Web Site Administration Tool

Question: How will you identify an inherited setting in the Web Site Administration Tool?

Answer: An inherited setting appears dimmed to indicate that it is disabled in the Web Site Administration Tool.

Retrieving Data from Web.config

Question: How will you protect the configuration file on the server?

Answer: You can protect the configuration file on the server by using Windows security settings to limit who can read the file. Avoid storing sensitive information—such as user credentials—in the appSettings element of the web.config file. You can also encrypt configuration settings.

Writing Data to Web.config

Question: How will you prevent users from modifying the security settings?

Answer: You can lock the configuration settings.

Detailed Demo Steps

Demonstration: How to Inherit Configuration Settings

Demonstration steps

1. Log on to 10267A-GEN-DEV as **Student**, with the password, **Pa$$w0rd**.
2. Open Microsoft Visual Studio® 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
3. Open the **ConfigInheritance** Web site from either the D:\Demofiles\M14\CS\ConfigInheritance or D:\Demofiles\M14\VB\ConfigInheritance folder.
    - On the **File** menu of Visual Studio 2010, click **Open Web Site**.
    - In the **Folder** box of the **Open Web Site** dialog box, type either **D:\Demofiles\M14\CS\ConfigInheritance** or **D:\Demofiles\M14\VB\ConfigInheritance**, and then click **Open**.
4. View the code for the **Default.aspx** Web Form and the markup for the **web.config** file from the main folder.
    - In Solution Explorer, right-click **Default.aspx**, and then click **Open**.
    - In Solution Explorer, right-click **web.config**, and then click **Open**.

    Note: View the value of the **Counter add** element that is stored in the web.config file.
5. Open **SubFolder**, and show the code for the **GetConfigSubFolder.aspx** Web Form.
    - In Solution Explorer, expand **SubFolder**, right-click **GetConfigSubFolder.aspx**, and then click **Open**.
6. Open the **web.config** file in SubFolder.
    - In Solution Explorer, right-click **web.config**, and then click **Open**.

    Note: Notice that the value of the **Counter add** element that is stored in the web.config file is different from the **Counter add** element that is in the web.config file in the main folder.
7. View the **Default.aspx** Web Form in the browser, and then highlight the value that was retrieved from the web.config file.
    - In Solution Explorer, right-click **Default.aspx**, and then click **View in Browser**.
8. View the **GetConfigSubFolder.aspx** Web Form in the browser.
    - In Solution Explorer, right-click **GetConfigSubFolder.aspx**, and then click **View in Browser**.

    Note: Notice that the value for the GetConfigSubFolder.aspx page is different from that of the Default.aspx page.
9. Close Windows Internet Explorer®.

- In the http://localhost:50156/ConfigInheritance/SubFolder/GetConfigSubFolder.aspx - Windows Internet Explorer window, click the **Close** button.

10. In the **SubFolder**, rename the **web.config** file to **oldWeb.config**.
    - In Solution Explorer, under **SubFolder**, right-click **web.config**, click **Rename**, type **oldWeb.config**, and then press ENTER.
11. View the **Default.aspx** Web Form in the browser, and then highlight the value that was retrieved from the web.config file.
    - In Solution Explorer, right-click **Default.aspx**, and then click **View in Browser**.
12. View the **GetConfigSubFolder.aspx** Web Form in the browser, and verify that the value is the same as the value in the web.config file in the main directory.
    - In Solution Explorer, right-click **GetConfigSubFolder.aspx**, and then click **View in Browser**.
    - In the http://localhost:50156/ConfigInheritance/SubFolder/GetConfigSubFolder.aspx - Windows Internet Explorer window, click the **Close** button.
    - In the ConfigInheritance - Microsoft Visual Studio window, click the **Close** button.

Demonstration: How to Configure the List View Page Size

Demonstration steps

1. Log on to the 10267A-GEN-DEV virtual machine as **Student**, with the password, **Pa$$w0rd**.
2. Open Microsoft Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
3. Open the **CustomerManagement** solution from either the D:\Demofiles\M14\VB or D:\Demofiles\M14\CS folder.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **Open Project**.
    - In the **Open Project** dialog box, in the **File name** box, type either **D:\Demofiles\M14\VB\CustomerManagement.sln** or **D:\Demofiles\M14\CS\CustomerManagement.sln**, and then click **Open**.
4. Set the **Default** Web Form as the start page for the project.
    - In Solution Explorer, right-click **Default.aspx**, and then click **Set As Start Page**.
5. Run the application.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.
6. View all the countries.
    - In the Contoso Customer Management – Windows Internet Explorer window, on the **Countries** menu, click **All**.

    Note: Notice that by default, 10 items display in the table. This is because the page size is set to 10.
7. Close Internet Explorer.

- In the Countries – Windows Internet Explorer window, click the **Close** button.

8. Open the **web.config** file.
    - In Solution Explorer, right-click **web.config**, and then click **Open**.
9. In the web.config file, add a new self-closing **add** element within the **appSettings** element, with a key attribute value of **ListViewPagerSize**, and a value attribute value of **5**.

```
<appSettings>
       <add key="ListViewPagerSize" value="5" />
</appSettings>
```

    - In the web.config window, type the following markup, after the closing **connectionStrings** element.

```
<appSettings>
       <add key="ListViewPagerSize" value="5" />
</appSettings>
```

10. Save and close the web.config file.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click **Save web.config**.
    - In the web.config window, click the **Close** button.
11. Navigate to **DynamicData/PageTemplates**, and open the **List Web Form** in Code view.
    - In Solution Explorer, expand **DynamicData**, expand **PageTemplates**, right-click **List.aspx**, and then click **View Code**.
12. In the **Page_Load** event handler of the partial **List** class, check whether the **ListViewPagerSize** key exists in the **appSettings** element of the web.config file. Do this by using the **AppSettings** property of the **System.Web.Configuration.WebConfigurationManager** class, and by using the **If** or **if** construct.
    - In the DynamicData/PageTemplates/List.aspx.vb or DynamicData/PageTemplates/List.aspx.cs window, in the **Page_Load** event handler of the partial **List** class, append the following code.

[Visual Basic]

```
' Check if key exists in web.configIf Not
System.Web.Configuration.WebConfigurationManager.AppSettings("ListViewPagerSize")
Is Nothing Then
End If
```

[Visual C#]

```
// Check if key exists in web.config
if
(System.Web.Configuration.WebConfigurationManager.AppSettings["ListViewPagerSize"]
!= null)
{
}
```

13. Retrieve the default page size value from the web.config file in the **Page_Load** event handler, in the **If**/**if** construct, by using the **AppSettings** property of the

System.Web.Configuration.WebConfigurationManager class, and save the retrieved value in a variable named **pagerSize**, of type **Integer** or **int**.

- In the **Page_Load** event handler of the partial List class, in the **If** or **if** construct, append the following code.

[Visual Basic]

```
' Get pager size from configuration file
Dim pagerSize As Integer = Integer.Parse( _
System.Web.Configuration.WebConfigurationManager.AppSettings("ListViewPagerSize"))
```

[Visual C#]

```
// Get pager size from configuration file
int pagerSize = int.Parse(
System.Web.Configuration.WebConfigurationManager.AppSettings["ListViewPagerSize"])
;
```

14. Set the page size of the **GridView** control to the value of the **pagerSize** variable.

    - In the **Page_Load** event handler of the partial **List** class, in the **If**/**if** construct, append the following code.

[Visual Basic]

```
' Set page size
GridView1.PageSize = pagerSize
```

[Visual C#]

```
// Set page size
GridView1.PageSize = pagerSize;
```

15. Save the changes, and close the List.aspx.vbs or List.aspx.cs file.

    - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click either **Save DynamicData/PageTemplates/List.aspx.vb** or **Save DynamicData/PageTemplates/List.aspx.cs**.
    - In the DynamicData/PageTemplates/List.aspx.vb or DynamicData/PageTemplates/List.aspx.cs window, click the **Close** button.

> **Note:** Notice the new page size in the List View Web Form.

16. Run the application.

    - In the CustomerManagement – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.

17. View all countries.

    - In the Contoso Customer Management – Windows Internet Explorer window, on the **Countries** menu, click **All**.

> **Note:** Notice that only five items display in the table. This is because the page size is set to 5.

18. Close Internet Explorer.
    - In the Countries – Windows Internet Explorer window, click the **Close** button.
    - In the CustomerManagement – Microsoft Visual Studio window, click the **Close** button.

Demonstration: How to Configure the Visitor Counter

Demonstration steps

1. Log on to the 10267A-GEN-DEV virtual machine as **Student**, with the password, **Pa$$w0rd**.
2. Open Microsoft Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
    - In the **User Account Control** dialog box, in the text box, type **Pa$$w0rd**, and then click **Yes**.
3. Open the **CustomerManagement** solution from either the D:\Demofiles\M14\VB or D:\Demofiles\M14\CS folder.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **Open Project**.
    - In the **Open Project** dialog box, in the **File name** box, type either **D:\Demofiles\M14\VB\CustomerManagement.sln** or **D:\Demofiles\M14\CS\CustomerManagement.sln**, and then click **Open**.
4. Open the **web.config** file.
    - In Solution Explorer, right-click **web.config**, and then click **Open**.
5. In the web.config file, add a new self-closing add element within the **appSettings** element, with a key attribute value of **VisitorCounter**, and a value attribute with a value of **0**.

```
<add key="VisitorCounter" value="0" />
```

    - In the web.config window, type the following markup,

```
<add key="VisitorCounter" value="0" />
```

      after the markup:

```
<add key="ListViewPagerSize" value="5" />
```

6. Save and close the web.config file.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click **Save web.config**.
    - In the web.config window, click the **Close** button.
7. Open the **Global.asax** file.
    - In Solution Explorer, right-click **Global.asax**, and then click **Open**.
8. In the **Application_Start** method, declare a local variable named **numVisitors**, of type **Long** or **long**, and initialize the variable with a value of **0**.
    - In the Global.asax window, in the **Application_Start** method, type the following code,

[Visual Basic]

```
Dim numVisitors As Long = 0
```

[Visual C#]

```
long numVisitors = 0;
```

after the code:

[Visual Basic]

```
Register(RouteTable.Routes)
```

[Visual C#]

```
Register(RouteTable.Routes);
```

9. Check whether the VisitorCounter key exists in the **appSettings** element of the web.config file in the **Application_Start** method. To check this, use the **AppSettings** property of the **System.Web.Configuration.WebConfigurationManager** class in an **If** or **if** construct.

[Visual Basic]

```
' Check if key exists in web.config
If Not
System.Web.Configuration.WebConfigurationManager.AppSettings("VisitorCounter") Is
Nothing Then
End If
```

[Visual C#]

```
// Check if key exists in web.config
if (System.Web.Configuration.WebConfigurationManager.AppSettings["VisitorCounter"]
!= null)
{
}
```

- In the Global.asax window, in the **Application_Start** method, append the following code,

[Visual Basic]

```
' Check if key exists in web.config
If Not
System.Web.Configuration.WebConfigurationManager.AppSettings("VisitorCounter") Is
Nothing Then
End If
```

[Visual C#]

```
// Check if key exists in web.config
if (System.Web.Configuration.WebConfigurationManager.AppSettings["VisitorCounter"]
!= null)
{
}
```

after the code:

[Visual Basic]

```
Dim numVisitors As Long = 0
```

[Visual C#]

```
long numVisitors = 0;
```

10. Retrieve the visitor counter value from the web.config file in the **Application_Start** method by using the **AppSettings** property of the **System.Web.Configuration.WebConfigurationManager** class, and save the retrieved value in the **numVisitors** variable.

[Visual Basic]

```
' Get visitor counter from configuration file
numVisitors = Long.Parse(
System.Web.Configuration.WebConfigurationManager.AppSettings("VisitorCounter"))
```

[Visual C#]

```
// Get visitor counter from configuration file
numVisitors = long.Parse(
System.Web.Configuration.WebConfigurationManager.AppSettings["VisitorCounter"]);
```

- In the Global.asax window, in the **If** or **if** construct of the **Application_Start** method, type the following code,

[Visual Basic]

```
' Get visitor counter from configuration file
numVisitors = Long.Parse(
System.Web.Configuration.WebConfigurationManager.AppSettings("VisitorCounter"))
```

[Visual C#]

```
// Get visitor counter from configuration file
numVisitors = long.Parse(
System.Web.Configuration.WebConfigurationManager.AppSettings["VisitorCounter"]);
```

after the code:

[Visual Basic]

```
' Check if key exists in web.config
If Not
System.Web.Configuration.WebConfigurationManager.AppSettings("VisitorCounter") Is
Nothing Then
```

[Visual C#]

```
if (System.Web.Configuration.WebConfigurationManager.AppSettings["VisitorCounter"]
!= null)
{
```

11. Save the visitor counter value to the Application state in the Visitors application variable, by modifying the existing assignment of the value to **0**.

[Visual Basic]

```
' Save application variable
Application("Visitors") = numVisitors
```

[Visual C#]

```
// Save application variable
Application["Visitors"] = numVisitors;
```

- In the Global.asax window, in the **Application_Start** method, modify the following code,

[Visual Basic]

```
' Save application variable
Application("Visitors") = 0
```

[Visual C#]

```
// Save application variable
Application["Visitors"] = 0;
```

as:

[Visual Basic]

```
' Save application variable
Application("Visitors") = numVisitors
```

[Visual C#]

```
// Save application variable
Application["Visitors"] = numVisitors;
```

12. Save the Global.asax file.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click **Save Global.asax**.

Additional Reading

Overview of a Machine.config File

For more information about the processModel section, see processModel Element (ASP.NET Settings Schema).

For more information about sections and section handlers in configuration files, see ASP.NET Configuration File Structure (Sections and Section Handlers).

Overview of ASP.NET Configuration Inheritance

For more information about the general attributes, see General Attributes Inherited by Section Elements.

For more information about managed threads, see Exceptions in Managed Threads.

Overview of the Web Site Administration Tool

See Web Site Administration Tool Security Tab.

See Web Site Administration Tool Application Tab.

See Web Site Administration Tool Provider Tab.

See Web Site Administration Tool Internals.

See Web Site Administration Tool Overview.

Retrieving Data from Web.config

For more information about reading application settings, see How to: Read Application Settings from the Web.config File.

For more information about reading connection strings, see How to: Read Connection Strings from the Web.config File.

Writing Data to Web.config

For more information about locking configuration settings, see How to: Lock ASP.NET Configuration Settings.

For more information about the configuration element, see KeyValueConfigurationElement Class.

For more information about the OpenWebConfiguration method, see WebConfigurationManager.OpenWebConfiguration Method (String).

Lesson 2

Deploying an ASP.NET Web Application

Contents:

Questions and Answers

Sharing Assemblies in the Global Assembly Cache

Question: What is the requirement for adding an assembly to the GAC?

Answer: The assembly must have a strong name.

Web Application Deployment

Question: What is XCOPY deployment?

Answer: XCOPY deployment is the process of manually copying files to a destination server by using the XCOPY command-line utility.

Overview of the Copy Web Site Tool

Question: How will you ensure that the files are the same on all sites?

Answer: Synchronize the files. For example, if a file on the remote site is more current than the version of the same file on the local site, synchronize the files. In this example, synchronization copies the file on the remote site to your local site.

Overview of the Publish Web Site Utility

Question: Can you deploy a compiled site to a remote server by using the Publish Web Site utility?

Answer: No, you cannot deploy a compiled site to a remote server by using the Publish Web Site utility. The utility can only copy the sites to the local computer, or to another computer on the local area network.

Overview of FTP Deployment

Question: Can the same content be accessible by using both the FTP and HTTP protocol?

Answer: Yes, the same content can be accessible by using both the FTP and HTTP protocol, if the server hosting the content has both an FTP and Web Server set up to serve the same content.

Additional Reading

Sharing Assemblies in the Global Assembly Cache

For information about COM interop, see Introduction to COM Interop.

For more information about assembly versioning, see Assembly Versioning.

Web Application Deployment

For more information about and downloading Visual Studio 2010 Web Deployment Projects, see Visual Studio® 2010 Web Deployment Projects - Beta 1.

Overview of the Copy Web Site Tool

For information about copying files from a local Web site to a remote site, see How to: Deploy Web Site Projects by using the Copy Web Site Tool.

Overview of the Publish Web Site Utility

For information about viewing inherited and local configuration settings, see How to: View Inherited and Local Configuration Settings Programmatically.

For more information about configuration settings, see General Configuration Settings (ASP.NET), and ASP.NET Configuration Settings.

For more information about configuring published Web sites, see How to: Configure Published Web Site Projects.

For more information about encrypting specific configuration settings, see Encrypting Configuration Information Using Protected Configuration.

Overview of FTP Deployment

For more information about working with FTP sites, see Walkthrough: Editing Web Sites with FTP in Visual Web Developer.

Module Reviews and Takeaways

Review questions and answers

1. Which files can you use to configure an ASP.NET Web application?

   You can use the machine.config or the root web.config file for machine-wide configuration. You can use web.config files for Web application and subdirectory configurations.

2. What are the options for deploying an ASP.NET Web application?

   The options for deploying an ASP.NET Web application are as follows:

- Copy the files manually by using XCOPY.
- Use the Copy Web Site tool.
- Use the Publish Web Site utility.
- Create a Web Setup project.
- If the target server is an FTP server, you can open and edit the files on the FTP server directly.

Real-world issues and scenarios

1. You want to restrict specific configuration settings to a single virtual directory for your Web application. How do you do this?

   You can place a web.config file in the physical folder that corresponds to the virtual directory containing the configuration settings that are specific to the virtual directory only.

2. You want to add configuration settings that are shared between all.NET applications on a single computer. To which configuration file should you add the configuration settings?

   You can add the configuration settings to the machine.config file in the C:\Windows\Microsoft.NET\Framework\v4.0.30128\Config folder.

Best practices

Mention some best practices in the context of your own business situations.

- Any values that are configurable by an administrator should be saved to the web.config file for easy access.
- Turn off debugging and tracing in the web.config file before deployment.
- Specify a version, and sign your assemblies.
- Deploy the shared assemblies to the GAC.

Lab Review Questions and Answers

1. How will you set the default visitor counter value?

   Answer: To set the default visitor counter, you need to add to the web.config file a new self-closing add element within the appSettings element, with a key attribute value of VisitorCounter, and a value attribute with a value of 0.

2. Which property will you use to save a new key and a value for the **VisitorCounter** in the web.config file?

   Answer: To use a new key and a value for the VisitorCounter in the web.config file, you need to use the System.Web.Configuration.WebConfigurationManager.AppSettings property.

Module 15

Securing a Microsoft® ASP.NET Web Application

Contents:

Lesson 1

Overview of Web Application Security

Contents:

Question and Answers

What Are Authentication and Authorization Methods?

Question: Does authentication occur before or after authorization, or do they both occur simultaneously?

Answer: Authentication happens before authorization.

ASP.NET Authentication Methods

Question: Which authentication method uses IIS authentication?

Answer: Windows-based authentication uses IIS authentication.

Scenarios That Support ASP.NET Authentication Methods

Question: Why is Windows-based authentication not suitable for most Internet applications?

Answer: Windows-based authentication requires a client to have a Windows user account for logging on to the Web application. This can be impractical to manage if the Web application has many users, and these types of user accounts are created and deleted frequently, because of the administrative overhead. There is also a licensing issue when using a Windows user account.

Forms-Based Authentication and Authorization

Question: When using Forms-based authentication, how are authentication credentials saved after a successful logon?

Answer: Authentication credentials are saved to the client by using a cookie, if cookies are not explicitly disabled on the client.

Secure Sockets Layer Protocol

Question: How can you improve the security of Basic authentication?

Answer: You can improve the security of Basic authentication by communicating over a secure protocol, such as SSL.

Configuration File Protection

Question: What information can you encrypt in a Web configuration file?

Answer: You can encrypt information such as user names and passwords, database connection strings, and encryption keys that are located in a Web application configuration file.

Working with Protected Configuration

Question: What are the names of the two protected configuration providers?

Answer: The two protected configuration providers are DpapiProtectedConfigurationProvider, and RsaProtectedConfigurationProvider.

Accessing Protected Configuration

Question: Which command-line tool is used to encrypt and decrypt the web.config file?

Answer: The ASP.NET IIS Registration tool (or, aspnet_regiis.exe) is used to encrypt and decrypt the web.config file.

Additional Reading

What Are Authentication and Authorization Methods?

For more information about authorization and authentication, see Building Secure ASP.NET Applications: Authentication, Authorization, and Secure Communication.

For more information about Active Directory, see Active Directory.

ASP.NET Authentication Methods

For more information about Windows Authentication Provider, see Windows Authentication Provider.

For more information about Forms Authentication Provider, see Forms Authentication Provider.

For information about how to use Windows Live ID to authenticate users for your Web site, see Windows Live ID SDK.

IIS Authentication Mechanisms

For more information about the built-in user and group accounts in IIS 7, see Understanding Built-In User and Group Accounts in IIS 7.

For more information about using Windows Active Directory to authenticate domain users who have client certificates, see Map Client Certificates by Using Active Directory Mapping (IIS 7).

For more information about mapping client certificates one-to-one, see Map Client Certificates One-to-One (IIS 7).

For more information about mapping client certificates many-to-one, see Map Client Certificates Many-to-One (IIS 7).

For more information about configuring the Windows authentication method, see Configure Windows Authentication (IIS 7).

Secure Sockets Layer Protocol

For more information about setting up SSL, see How to Set Up SSL on IIS 7.

For more information about SSL certificates, see SSL Certificates.

Working with Protected Configuration

For more information about using a protected configuration provider, see Specifying a Protected Configuration Provider.

Lesson 2

Declaratively Configuring Authentication and Authorization

Contents:

Question and Answers

Configuring Windows-Based Authentication

Question: What must be disabled in IIS to use Windows authentication?

Answer: Anonymous access must be disabled in IIS to use Windows authentication.

Configuring Forms-Based Authentication

Question: What must be enabled in IIS to use Forms-based authentication?

Answer: Anonymous access must be enabled in IIS to use Forms-based authentication.

Configuring Authorization

Question: How can you allow or deny users access to a specific page?

Answer: You allow or deny users access to a specific page by adding a self-closing allow or deny element to the authorization element.

Additional Reading

Configuring Windows-Based Authentication

For more information about the Windows authentication method, see Explained: Windows Authentication in ASP.NET 2.0.

Configuring Forms-Based Authentication

For more information about Form-based authentication, see Explained: Forms Authentication in ASP.NET 2.0.

Configuring Authorization

For more information about ASP.NET authorization, see ASP.NET Authorization.

Lesson 3

Working Programmatically with Authentication and Authorization

Contents:

Question and Answers

Overview of Login Controls

Question: Which **Login** control allows a user to log on or log off?

Answer: The LoginStatus control allows a user to log on or log off.

Demonstration: How to Add Login Controls to a Master Page

Question: When using Windows authentication with the **LoginName** control, what name displays when viewing the rendered Web Form?

Answer: The name of the user logged on to the machine or network on which the Web site is running will display by the LoginName control.

Demonstration: How to Add Account Forms to a Web Site

Question: What is a Login Web Form?

Answer: A Login Web Form is a standard ASP.NET Web Form that contains one or more Login controls.

Accessing User Identity

Question: Which properties of the **IIdentity** interface can you use to ascertain the user identity, the IIS authentication mechanism, and user authentication?

Answer: The User.Identity object properties AuthenticationType, Name, and IsAuthenticated, can be used to ascertain the user identity, the IIS authentication mechanism, and user authentication.

Detailed Demo Steps

Demonstration: How to Add Login Controls to a Master Page

Demonstration steps

1. Log on to the 10267A-GEN-DEV virtual machine as **Student**, with the password, **Pa$$w0rd**.
2. Open Microsoft Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
3. Open the **CustomrManagement** solution from the D:\Demofiles\M15\VB or D:\Demofiles\M15\CS folder**.**
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **Open Project**.
    - In the **Open Project** dialog box, in the **File name** box, type either **D:\Demofiles\M15\VB\CustomerManagement.sln** or **D:\Demofiles\M15\CS\CustomerManagement.sln**, and then click **Open**.
4. Set the **Default** Web Form as the start page for the project.
    - In Solution Explorer, right-click **Default.aspx**, and then click **Set As Start Page**.
5. Open the master page **Site.master**, in Source view.
    - In Solution Explorer, right-click **Site.master**, and then click **View Markup**.
6. Add a **div** element with a **class** attribute set to the value of **login**, to the **body** element of the master page **Site.master**, after the **div** element with a **class** attribute set to the value of **appTitle**.

```
<div class="login">
</div>
```

    - In the Site.master window, locate the **div** element with a **class** attribute set to the value of **appTitle**, place the cursor after the closing **div** tag, press ENTER, and then type the following markup.

```
<div class="login">
</div>
```

7. Add a **LoginStatus** control named **MainLoginStatus**, to the **div** element with a **class** attribute value of **login**.

```
<asp:LoginStatus ID="MainLoginStatus" runat="server"
LogoutAction="RedirectToLoginPage" LogoutPageUrl="~/Account/Login.aspx" />
```

> **Note:** The **LoginStatus** control must have the **LogoutAction** attribute set to the value of **RedirectToLoginPage** to send the user to the login page. Optionally, the attribute, **LogoutPageUrl**, can be set to the value of **~/Account/Login.aspx**, to redirect to this specific page, instead of the one specified in the web.config file.

- In the Site.master window, place the cursor at the end of the opening **div** tag that you have added, and then press ENTER.
- In the Site.master window, point to **Toolbox**.
- In Toolbox, expand **Login**, and then double-click **LoginStatus**.
- Change the **ID** attribute of the **LoginStatus** control from **LoginStatus1** to **MainLoginStatus**.
- Add a **LogoutAction** attribute, and set the value to **RedirectToLoginPage**.
- Add a **LogoutPageUrl** attribute, and set the value to **~/Account/Login.aspx**.

```
<asp:LoginStatus ID="MainLoginStatus" runat="server"
LogoutAction="RedirectToLoginPage" LogoutPageUrl="~/Account/Login.aspx" />
```

8. Add a **LoginName** control named **MainLoginName** to the **div** element after the **MainLoginStatus** control.

```
<asp:LoginName ID="MainLoginName" runat="server" />
```

- In the Site.master window, place the cursor at the end of the self-closing **MainLoginStatus** element, and then press ENTER.
- In the Site.master window, point to **Toolbox**.
- In Toolbox, expand **Login**, and then double-click **LoginName**.
- Change the **ID** attribute of the **LoginName** control from **LoginName1** to **MainLoginName**.

```
<asp:LoginName ID="MainLoginName" runat="server" />
```

9. Save and close the master page.
   - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click **Save Site.master**.
   - In the Site.master window, click the **Close** button.
10. Run the application.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.
11. View the Login controls.

> **Note:** Notice the **Login** controls in the upper-right corner of the window, display a Login link. If the current user had been logged in, the **LoginName** control would have displayed the full name of the user, which in this case would be 10267A-GEN-DEV\student.

12. Close Internet Explorer.
    - In the Contoso Customer Management – Windows Internet Explorer window, click the **Close** button.

Demonstration: How to Add Account Forms to a Web Site

In this demonstration, you will see how to add Web Forms which are used for User accounts, to an existing Web Site. This demonstration will continue where you left off with the previous demonstration.

1. Open a second instance of Visual Studio 2010.
    - On the **Start** menu **of 10267A-GEN-DEV**, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
2. Create a new ASP.NET Web site.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **New Web Site**.
    - In the **New Web Site** dialog box, in the left pane, ensure that either **Visual Basic** or **Visual C#** is selected, in the middle pane, click **ASP.NET Web Site**, and then in the **Web location** list, click **File System**. In the adjacent text box, type either **D:\Demofiles\M15\VB\SampleWebSite** or **D:\Demofiles\M15\CS\SampleWebSite**, and then click **OK**.
3. Close second instance of Visual Studio 2010.
    - In the SampleWebSite – Microsoft Visual Studio window, click the **Close** button, or click the ALT+F4 keys.
4. Add the **Account** folder and default account content from the SampleWebSite Web site to the **CustomerManagement** Web site by copying the Account folder from the path **D:\Demofiles\M15\VB\SampleWebSite\Account** or **D:\Demofiles\M15\CS\SampleWebSite\Account**.
    - On the **Start** menu, in the search box type either **D:\Demofiles\M15\VB\SampleWebSite** or **D:\Demofiles\M15\CS\SampleWebSite**, and then press ENTER.
    - In the Windows Explorer window, right-click **Account**, and then click **Copy**.
    - Press the BACKSPACE key.
    - Right-click **CustomerManagement**, and then click **Paste**.
5. Close the Windows Explorer window.
6. In Solution Explorer, refresh the Web site content to ensure the addition of the **Account** folder.
    - In Solution Explorer, right-click either **D:\Labfiles\Starter\M15\VB\CustomerManagement** or **D:\Labfiles\Starter\M15\CS\CustomerManagement**, and then click **Refresh Folder**.
7. Expand the **Account** folder.
8. Open the **ChangePassword.aspx** Web Form in Source view.
    - In Solution Explorer, under **Account**, double-click **ChangePassword.aspx**.
9. Replace the name of the ContentPlaceHolder named **MainContent**, with the name of the ContentPlaceHolder control used in the existing master page, named **MainContentPlaceHolder**.
    - In the Account/ChangePassword.aspx window, replace **MainContent** in the **BodyContent** Content control with **MainContentPlaceHolder**.

10. Remove the **Content** control with an **ID** attribute value of **HeaderContent**.
    - In the Account/ChangePassword.aspx window, locate and select the **Content** control with an **ID** attribute value of **HeaderContent**.

```
<asp:Content ID="HeaderContent" runat="server" ContentPlaceHolderID="HeadContent">
</asp:Content>
```

    - On the **Edit** menu, click **Delete**, or press the DELETE key, to delete the selected markup.
11. Save and close **ChangePassword.aspx** Web Form.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click **Save Account/ChangePassword.aspx**, or press CTRL+SHIFT+S.
    - In the Account/ChangePassword.aspx window, click the **Close** button.
12. Open the **ChangePasswordSuccess.aspx** Web Form in Source view.
    - In Solution Explorer, under **Account**, double-click **ChangePasswordSuccess.aspx**.
13. Replace the name of the ContentPlaceHolder named **MainContent**, with the name of the ContentPlaceHolder control used in the existing master page, named **MainContentPlaceHolder**.
    - In the Account/ChangePasswordSuccess.aspx window, replace **MainContent** in the **BodyContent** Content control with **MainContentPlaceHolder**.
14. Remove the **Content** control with an **ID** attribute value of **HeaderContent**.
    - In the Account/ChangePasswordSuccess.aspx window, locate and select the **Content** control with an **ID** attribute value of **HeaderContent**.

```
<asp:Content ID="HeaderContent" runat="server" ContentPlaceHolderID="HeadContent">
</asp:Content>
```

    - On the **Edit** menu, click **Delete**, or press DELETE, to delete the selected markup.
15. Save and close **ChangePasswordSuccess.aspx** Web Form.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click **Save ChangePasswordSuccess.aspx**, or press CTRL+SHIFT+S.
    - In the Account/ChangePasswordSuccess.aspx window, click the **Close** button.
16. Open the **Login.aspx** Web Form in Source view.
    - In Solution Explorer, under **Account**, double-click **Login.aspx**.
17. Replace the name of the ContentPlaceHolder named **MainContent**, with the name of the ContentPlaceHolder control used in the existing master page, named **MainContentPlaceHolder**.
    - In the Account/Login.aspx window, replace **MainContent** in the **BodyContent** Content control with **MainContentPlaceHolder**.
18. Remove the **Content** control with an **ID** attribute value of **HeaderContent**.
    - In the Account/Login.aspx window, locate and select the **Content** control with an **ID** attribute value of **HeaderContent**.

```
<asp:Content ID="HeaderContent" runat="server" ContentPlaceHolderID="HeadContent">
</asp:Content>
```

- On the **Edit** menu, click **Delete**, or press DELETE, to delete the selected markup.

19. Save and close **Login.aspx** Web Form.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click **Save Account/Login.aspx**, or press CTRL+SHIFT+S.
    - In the Account/Login.aspx window, click the **Close** button.
20. Open the **Register.aspx** Web Form in Source view.
    - In Solution Explorer, under **Account**, double-click **Register.aspx**.
21. Replace the name of the ContentPlaceHolder named **MainContent**, with the name of the ContentPlaceHolder control used in the existing master page, named **MainContentPlaceHolder**.
    - In the Account/Register.aspx window, replace **MainContent** in the **BodyContent** Content control with **MainContentPlaceHolder**.
22. Remove the **Content** control with an ID attribute value of **HeaderContent**.
    - In the Account/Register.aspx window, locate and select the **Content** control with an ID attribute value of **HeaderContent**.

```
<asp:Content ID="HeaderContent" runat="server" ContentPlaceHolderID="HeadContent">
</asp:Content>
```

    - On the **Edit** menu, click **Delete**, or press DELETE, to delete the selected markup.
23. Save and close **Register.aspx** Web Form.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click **Save Account/Register.aspx**, or press CTRL+SHIFT+S.
    - In the Account/Register.aspx window, click the **Close** button.
24. Open the Login Web Form.
    - In Solution Explorer, under Account, double-click **Login.aspx**.
25. Set the page title as **Contoso Customer Management – User Login**.
    - In the Login.aspx window, locate the **Title** attribute of the Page directive, and replace **Log In** with **Contoso Customer Management - User Login**.
26. Save and close the **Login** Web Form.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **File** menu, click **Save Account/Login.aspx**.
    - In the Account/Login.aspx window, click the **Close** button.
27. Run the application.
    - In the CustomerManagement – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.
28. Click **Login**.
    - In the Contoso Customer Management – Windows Internet Explorer window, click **Login**.

Note: Notice that you are redirected to the new Login page.

29. Close Internet Explorer.
    - In the Log In – Windows Internet Explorer window, click the **Close** button.

Additional Reading

Overview of Login Controls

For more information about Login controls, see ASP.NET Login Controls Overview.

Accessing User Identity

For more information about the HttpContext class, see HttpContext Class.

For more information about the IPrincipal interface, see IPrincipal Interface.

For more information about the WindowsPrincipal class, see WindowsPrincipal Class.

For more information about the RolePrincipal class, see RolePrincipal Class.

For more information about the IIdentity interface, see IIdentity Interface.

For more information about the WindowsIdentity class, see WindowsIdentity Class.

For more information about the FormsIdentity class, see FormsIdentity Class.

Module Reviews and Takeaways

Review questions and answers

1. What is authentication?

   Authentication is the process of obtaining identification credentials—such as a name and a password—from a user, and validating those credentials against some authority, such as a database.

2. How can you specify the type of authorization that you want to enforce in the **system.web** element?

   You can create an authorization element to specify the type of authorization that you want to enforce.

3. What are the ASP.NET login controls?

   The ASP.NET login controls are ChangePassword, CreateUserWizard, Login, LoginName, LoginStatus, LoginView, and PasswordRecovery.

Real-world issues and scenarios

1. You need to use Forms authentication for your Web application. How will you do this?

   You can add or modify an **authentication** element in the web.config file, and set the **mode** attribute to a value of **Forms**.

2. You need to restrict access to your web application to only authenticated users. How will you do this?

   You can add or modify an **authorization** element in the web.config file, and then add a **deny** element with a users attribute set to a value of "**?**".

Best practices

Mention some best practices in the context of your own business situations.

- Never expose values by using a querystring that must remain secure.
- Provide strong names and version for your assemblies.
- Encrypt the connection strings in your web.config file.
- Apply authentication and authorization when possible, to secure access to resources.
- Encrypt traffic to and from your Web application by using the SSL protocol.

Lab Review Questions and Answers

1. How will you remove the default connection string for the **LocalSqlServer** from the web.config file?

   Answer: You can remove the default connection string for the LocalSqlServer from the web.config file by adding a remove element in the connectionString element.

2. How will you prevent access to the Import Countries page for all users?

   Answer: To disallow access to the Import Countries page for all users, you need to add a deny element to the web.config file with a users attribute value of *, in the authorization element.

Module 16

Implementing Advanced Technologies Supported by Microsoft® Visual Studio® 2010 for Web Development

Contents:

Lesson 1

Working with ASP.NET MVC 2 Framework

Contents:

Question and Answers

Overview of ASP.NET MVC 2

Question: Which components make up an ASP.NET MVC application?

Answer: The Model, View, and Controller components make up an ASP.NET MVC application.

Demonstration: Creating an ASP.NET MVC 2 Project

Question: What is the name of the Visual Studio 2010 project template for creating an ASP.NET MVC application?

Answer: The name of the Visual Studio 2010 project template for creating an ASP.NET MVC application is ASP.NET MVC 2 Web Application.

Compatibility of MVC 2 with ASP.NET Web Forms

Question: Does the ASP.NET MVC Web application work with .NET Framework 2.0?

Answer: No, the ASP.NET MVC Web application only works with .NET Framework 3.5 SP1 and newer versions.

Detailed Demo Steps

Demonstration: Creating an ASP.NET MVC 2 Project

Demonstration steps

1. Log on to the 10267A-GEN-DEV virtual machine as **Student**, with the password, **Pa$$w0rd**.
2. Open Microsoft Visual Studio 2010.
    - On the Start menu of 10267A-GEN-DEV, point to All Programs, click Microsoft Visual Studio 2010, and then click Microsoft Visual Studio 2010.
3. Create a new project named **MyFirstMvcApplication** in either the D:\Demofiles\M16\VB or D:\Demofiles\M16\CS folder.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **New Project**.
    - In the **New Project** dialog box, in the left pane, click either **Visual Basic** or **Visual C#,** at the top of the middle pane, ensure that **.NET** Framework **4** is selected, and in the middle pane, click **ASP.NET MVC 2 Web Application**. In the **Name** box, type **MyFirstMvcApplication**, in the **Location** box, type either **D:\Labfiles\Starter\M16\VB** or **D:\Labfiles\Starter\M16\CS**, and then click **OK**.
4. In the **Create Unit Test Project** dialog box, select the **No, do not create a unit test project** option, and then click **OK**.
5. Run the application.
    - In the MyFirstMvcApplication – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.

    Note: Notice that the URL for the Home Page is http://localhost:49236. No file name is shown.
6. Click **About**.
    - In the Home Page – Windows Internet Explorer window, click **About**.

    Note: Notice that the URL for the About page is http://localhost:49236/Home/About. Still no file name is shown.
7. Close Windows® Internet Explorer®.
    - In the About Us – Windows Internet Explorer window, click the **Close** button.
8. In the **HomeController.vb** or **HomeController.cs** window, notice the two methods, **Index** and **About**, which were just previously called in the browser.
9. In Solution Explorer, right-click **Global.asax**, and then click **Open**.
    - In the Global.asax.vb or Global.asax.cs window, in the **RegisterRoutes** method, notice the map routing, and in particular that the map routing, is the default route, which links the **Home** controller with the **Index** action.
10. Close the Global.asax.vb or Global.asax.cs window.
    - In the Global.asax.vb or Global.asax.cs window, click the **Close** button.

11. In Solution Explorer, expand **Views**, expand **Home**, and then notice the two views, **About**, and **Index**.

> **Note:** The About and Index views are returned by the corresponding **About** and **Index** methods of the Home controller.

12. Open the About view.
    - In Solution Explorer, double-click **Views/Home/About.aspx**.

> **Note:** The About view looks very similar to an ASP.NET Web Forms content page.

13. Close the About view.
    - In the About.aspx window, click the **Close** button.
14. In Solution Explorer, expand **Shared**, and then point out the master page **Site.Master**.

> **Note:** The Views/Shared folder stores views and master pages that are shared across multiple controllers.

15. Open the Site.Master master page.
    - In Solution Explorer, double-click **Views/Shared/Site.Master**.

> **Note:** The Site.master view looks somewhat similar to an ASP.NET Web Forms master page.

16. Close the Site.Master view.
    - In the Site.Master window, click the **Close** button.
17. Close Visual Studio 2010.
    - In the MyFirstMvcApplication – Microsoft Visual Studio window, click the **Close** button.

Additional Reading

Overview of ASP.NET MVC 2

For more information about the ASP.NET MVC 2 Framework, see ASP.NET MVC Overview.

Features of the ASP.NET MVC 2 Framework

For more information about the MVC life-cycle model, see Understanding the MVC Application Execution Process.

Compatibility of MVC 2 with ASP.NET Web Forms

For more information about the ASP.NET MVC and ASP.NET Web Forms compatibility, see Compatibility of ASP.NET Web Forms and ASP.NET MVC.

Lesson 2

Working with Silverlight 4

Contents:

Question and Answers

Overview of Silverlight

Question: Can you use Silverlight with the .NET Framework?

Answer: Yes, you can use Silverlight with the .NET Framework. You can create Silverlight-based applications by using languages such as Visual Basic and Visual C#, and by using development tools, such as Visual Studio 2010.

Silverlight Architecture

Question: What is the .NET Framework for Silverlight?

Answer: The .NET Framework for Silverlight is a subset of the .NET Framework that contains components and libraries, including data integration, extensible Windows® controls, networking, base class libraries, garbage collection, and CLR.

Installing Silverlight

Question: Do you need to bundle the Silverlight runtime with your Silverlight-based applications?

Answer: No, you need not bundle the Silverlight runtime with your Silverlight-based applications. To run Silverlight-based applications, a small plug-in is required in the user's browser. If a user does not have the plug-in, they are automatically prompted to install it when accessing a Silverlight-based Web page. The plug-in automatically downloads from the Microsoft Web site, if the user chooses to install the plug-in.

Silverlight Tools for Developing Applications

Question: Name at least two of the Visual Studio 2010 templates included in the Silverlight 4 Tools for Visual Studio 2010?

Answer: The Silverlight 4 Tools for Visual Studio 2010 include a Silverlight Application template, a Silverlight Class Library template, and a Silverlight Navigation Application template.

Silverlight Project Files

Question: Which class can you use to create the user interface for the Silverlight application?

Answer: You can use the MainPage class to create the user interface for the Silverlight application.

Demonstration: How to Add a StackPanel and Calendar to a Grid Layout

Question: Why do you add a StackPanel control to the Grid layout?

Answer: You add a StackPanel control to the Grid layout to arrange a small subsection of the UI on the Web page. In this demonstration, the StackPanel control vertically arranges the Calendar and Button controls on the grid layout.

Demonstration: How to Respond to Events in a Silverlight Application

Question: What is the use of the **x:Name** attribute?

Answer: The **x:Name** attribute uniquely identifies object elements for the purpose of accessing the instantiated element from the code-behind file. After **x:Name** is applied to a backing programming model, it can be considered equivalent to the variable holding an object reference, as returned by a constructor.

Demonstration: How to Resize Controls in a Grid Layout

Question: What is the purpose of assigning the value of **Auto** to the **RowDefinition Height** attributes of the **Grid.RowDefinitions**?

Answer: The **Grid.RowDefinition** control supports an auto-sizing mode, which automatically modifies the height of the grid or row, based on the size of the content contained within it.

Detailed Demo Steps

Demonstration: How to Create a New Silverlight Project

Demonstration steps

To create a Silverlight 4 project by using Visual Studio 2010, you need to perform the following steps:

1. Open Microsoft Visual Studio 2010.
    - On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.
2. Create a Silverlight project.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **New Project**.
    - In the **New Project** dialog box, in the left pane, click either **Visual Basic** or **Visual C#**.
    - In the middle pane, click **Silverlight Application**, and then click **OK**.
3. Select a method for hosting the Silverlight application.
    - In the **New Silverlight Application** dialog box, ensure that the **Host the Silverlight application in a new Web site** check box is selected, and then click **OK**.

    Note: You need to select the **Host the Silverlight application in a new Web site** option, if you want to add a separate ASP.NET Web site or ASP.NET Web Application Project to your solution to host the Silverlight application. If you select this option, you must also specify the **New Web project name,** and **New Web Project Type**. If you do not select the **Host the Silverlight application in a new Web site** option, you need to use a Web site to host your Silverlight application, and an HTML test page will be generated to host your application.
4. Build the Silverlight application project.
    - In the SilverlightApplication1 – Microsoft Visual Studio window, on the **Build** menu, click **Build Solution**.
5. Run the Silverlight application project.
    - In the SilverlightApplication1 – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.
6. Close Internet Explorer.
    - In the SilverlightApplication1 – Windows Internet Explorer window, click the **Close** button.
7. Close Visual Studio 2010.
    - In the SilverlightApplication1 – Microsoft Visual Studio window, click the **Close** button.

Demonstration: How to Add a Grid Layout to a Simple Silverlight Application

Demonstration steps

1. Open Microsoft Visual Studio 2010.

- On the **Start** menu of 10267A-GEN-DEV, point to **All Programs**, click **Microsoft Visual Studio 2010**, and then click **Microsoft Visual Studio 2010**.

2. Open the **HelloSilverlight** solution from either the D:\Demofiles\M16\CS or D:\Demofiles\M16\VB folder.
    - In the Start Page – Microsoft Visual Studio window, on the **File** menu, click **Open Project**.
    - In the **Open Project** dialog box, in the **File name** box, type either **D:\Demofiles\M16\VB\HelloSilverlight.sln** or **D:\Demofiles\M16\CS\HelloSilverlight.sln**, and then click **Open**.
3. Open the MainPage.xaml document from the **HelloSilverlight** project.
    - In Solution Explorer, right-click **MainPage.xaml**, and then click **Open**.
4. In XAML view, locate the **Grid** control.
    - In the MainPage.xaml window, in the **XAML** pane, locate the **Grid** control.
5. Modify the **Background** attribute of the **Grid** control, and set its value to **Beige**.

```
<Grid x:Name="LayoutRoot" Background="Beige">
```

    - In the opening tag of the **Grid** control, modify the **Background** attribute value to **Beige**.

```
Background="Beige"
```

6. Add the **ShowGridLines** attribute to the **Grid** control, and set its value to **True**.

```
<Grid x:Name="LayoutRoot" Background="Beige" ShowGridLines="True">
```

    - In the opening tag of the **Grid** control, add the **ShowGridLines** attribute and set its value to **True**.

```
ShowGridLines="True"
```

7. Define three rows and two columns within the **Grid** control.

```
<Grid.RowDefinitions>
       <RowDefinition Height="40"/>
       <RowDefinition Height="220"/>
       <RowDefinition Height="40"/>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
       <ColumnDefinition Width="75" />
       <ColumnDefinition Width="325"/>
</Grid.ColumnDefinitions>
```

    - Add the following XAML within the opening and closing tags of the **Grid** control.

```
<Grid.RowDefinitions>
       <RowDefinition Height="40"/>
       <RowDefinition Height="220"/>
       <RowDefinition Height="40"/>
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
       <ColumnDefinition Width="75" />
       <ColumnDefinition Width="325"/>
```

```
</Grid.ColumnDefinitions>
```

8. Run the application.

    - In the HelloSilverlight – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.

> **Note:** Notice that the browser opens and has a background color that you specified, as shown in the following illustration. You can also view the dotted lines that indicate the **Grid** that you defined.

9. Close Internet Explorer.

    - In the HelloSilverlight – Windows Internet Explorer window, click the **Close** button.

> **Note:** Keep the solution open in Visual Studio, as you will build on this demonstration in the next topic.

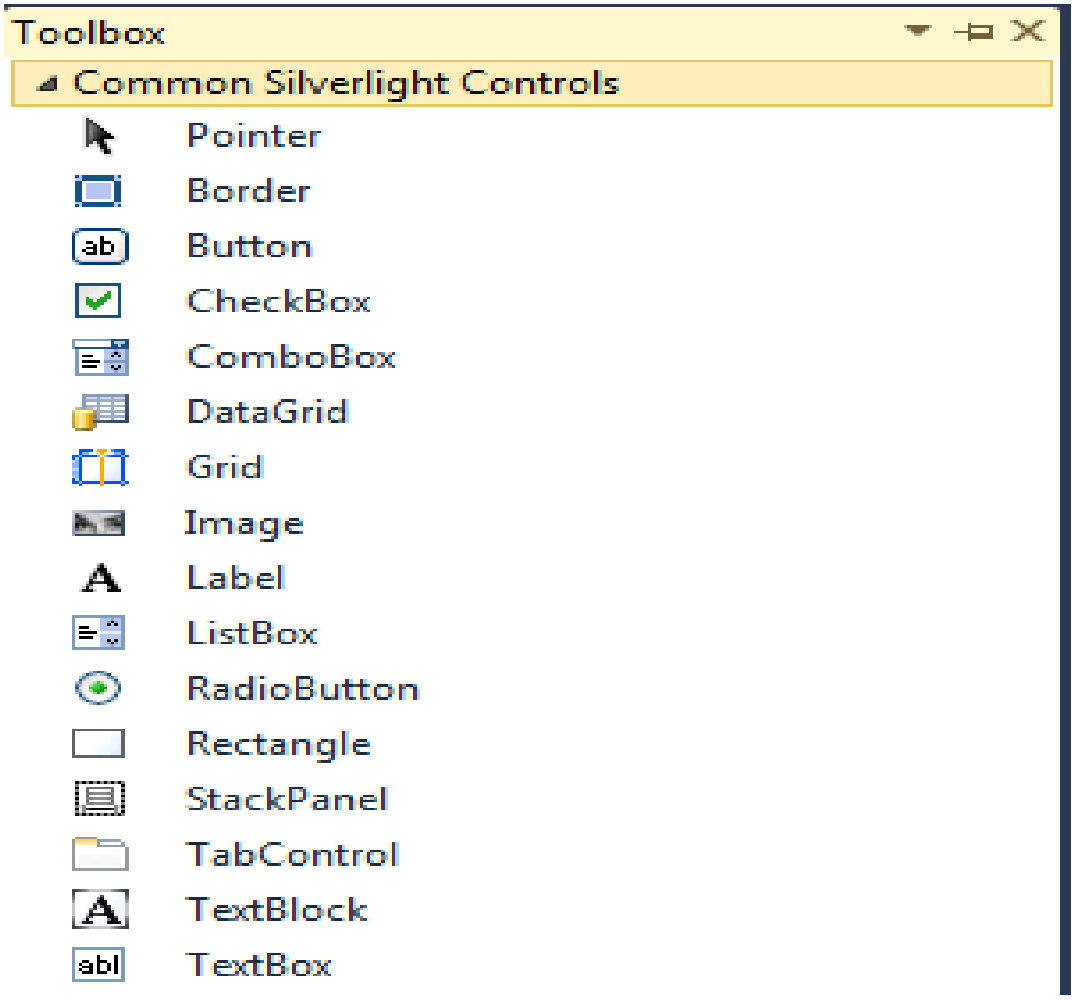
Demonstration: How to Place Controls in a Grid Layout

Demonstration steps

To place controls in a Grid layout, you need to perform the following steps:

1. Add a **TextBlock** control to the XAML view.

    - In Toolbox, under **Common Silverlight Controls**, drag a **TextBlock** control after the **Grid.ColumnDefinitions** closing tag in the XAML view.



2. Add a **Text** attribute with a value of **Name**:, to the **TextBlock** control.

    - In the self-closing **TextBlock** element, add the following **Text** attribute with a value of **Name:**.

```
<TextBlock Text="Name: " />
```

3. Drag another **TextBlock** control from Toolbox to the MainPage.xaml window, below the existing **TextBlock** control, and set a **Text** attribute with a value of **Date:**.

    - In Toolbox, under **Common Silverlight Controls**, drag a **TextBlock** control after the **Name TextBlock** element in the XAML view.
    - In the self-closing **TextBlock** element, add the following **Text** attribute with a value of **Date:**.

```
<TextBlock Text="Date: " />
```

4. Drag another **TextBlock** control from Toolbox to the MainPage.xaml window, below the existing **TextBlock** controls, and set a **Text** attribute with a value, **Message:**.

    - In Toolbox, under **Common Silverlight Controls**, drag a **TextBlock** control after the **Date TextBlock** element in the XAML view.
    - In the self-closing **TextBlock** element, add the following **Text** attribute with a value of **Message:**.

```
<TextBlock Text="Message: " />
```

5. Define the row and column for the **Name TextBlock**.

    - In the self-closing **Name TextBlock** element, add the following attributes.

```
Grid.Row="0" Grid.Column="0"
```

6. Define the row and column for the **Date TextBlock**.

    - In the self-closing **Date TextBlock** element, add the following attributes.

```
Grid.Row="1" Grid.Column="0"
```

7. Define the row and column for the **Message TextBlock**.

    - In the self-closing **Message TextBlock** element, add the following attributes.

```
Grid.Row="2" Grid.Column="0"
```

8. Add the following **ColumnSpan** attached attribute to the **Message TextBlock** to allow the text to span across both columns.

    - In the self-closing **Message TextBlock** element, add the following attribute.

```
Grid.ColumnSpan="2"
```

9. Drag a **TextBox** control from Toolbox to the MainPage.xaml window, after the **Name TextBlock** control.

- In Toolbox, under **Common Silverlight Controls**, drag a **TextBox** control after the self-closing **Name TextBlock** element in the XAML view.

10. Set the **Text** attribute with a value of **Your Name**, and define the row and column attributes for the **TextBox** control.
    - In the opening tag of the **TextBox** control, set the following properties.

```
<TextBox Text="Your Name" Grid.Row="0" Grid.Column="1" />
```

11. Run the application.
    - In the HelloSilverlight – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.

> **Note:** The Grid fills the cell (325 pixels wide), because the **HorizontalAlignment** and **VerticalAlignment** attributes are set to **Stretch** by default.

12. Close Internet Explorer.
    - In the HelloSilverlight – Windows Internet Explorer window, click the **Close** button.

> **Note**: Keep the solution open in Visual Studio, as you will build on this demonstration in the next topic.

Demonstration: How to Add a StackPanel and Calendar to a Grid Layout

Demonstration steps

To add a **StackPanel** and **Calendar** control to a Grid layout, you need to perform the following steps:

1. Set the **Width** property of the **TextBox** control to **150**.
    - In the **Your Name TextBox** element, set the **Width** property to **150**.

```
Width="150"
```

2. Set the **HorizontalAlignment** property of the **TextBox** control to **Left**.
    - In the **Your Name TextBox** element, set the alignment to **left** by adding the following **HorizontalAlignment** attribute.

```
HorizontalAlignment="Left"
```

3. Drag a **StackPanel** control from the Toolbox to the MainPage.xaml window, after the **Your Name TextBox** control.
    - In Toolbox, under **Common Silverlight Controls**, drag a **StackPanel** control after the **TextBox** element in the XAML view.
    - In the **StackPanel** element, type the following markup.

```
Grid.Column="1" Grid.Row="1" Orientation="Vertical"
```

4. Drag a **Calendar** control from the Toolbox to the MainPage.xaml window, within the **StackPanel** element.

- Change the self-closing **StackPanel** element to an opening and a closing tag.

```
<StackPanel Grid.Column="1" Grid.Row="1" Orientation="Vertical">
</StackPanel>
```

- In Toolbox, under **All Silverlight Controls**, drag a **Calendar** control within the opening and closing tags of the **StackPanel** element in the XAML view.

5. Add the **SelectionMode** attribute with a value of **SingleDate**, and the **HorizontalAlignment** attribute with a value of **Left**, to the **Calendar** control.

- In the Calendar element, type the following markup.

```
SelectionMode="SingleDate" HorizontalAlignment="Left"
```

Note: If you set the **SelectionMode** to **SingleDate,** you can select only one date at a time.

6. Drag a **Button** control from Toolbox to the MainPage.xaml window, after the **Calendar** control, and within the **StackPanel** element.

- In Toolbox, under **Common Silverlight Controls**, drag a **Button** control after the closing tag of the **Calendar** control, and within the **StackPanel** element in the XAML view.

Note: At present, the **Orientation** property of the **StackPanel** is set to **Vertical**. Thus, the Button will appear after the Calendar.

7. Set the **Width**, **Height**, **HorizontalAlignment**, and **Content** properties for the **Button** control.

- In the **Button** element, set the **Width**, **Height**, **HorizontalAlignment**, and **Content** properties by using the following markup.

```
Width="75" Height="25" HorizontalAlignment="Left" Content="OK"
```

Note: A **Button** control does not have a **Text** property, but has a **Content** property to add other elements, such as images or other controls.

8. Save the changes.

- In the HelloSilverlight – Microsoft Visual Studio window, on the **File** menu, click **Save All**.

Note: At this stage, all of the controls have been added. Your XAML should be similar to the following markup.

```
<UserControl xmlns:my="clr-
namespace:System.Windows.Controls;assembly=System.Windows.Controls"
x:Class="HelloSilverlight.MainPage"
     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
     xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
     xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
     mc:Ignorable="d"
     d:DesignHeight="300" d:DesignWidth="400">
     <Grid x:Name="LayoutRoot" Background="Beige" ShowGridLines="True">
          <Grid.RowDefinitions>
               <RowDefinition Height="40"/>
```

```
            <RowDefinition Height="220"/>
            <RowDefinition Height="40"/>
        </Grid.RowDefinitions>

        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="75" />
            <ColumnDefinition Width="325"/>
        </Grid.ColumnDefinitions>
        <TextBlock Text="Name: " Grid.Row="0" Grid.Column="0" />
        <TextBox Text="Your Name" Grid.Row="0" Grid.Column="1" Width="150"
        HorizontalAlignment="Left" />
        <StackPanel Grid.Column="1" Grid.Row="1" Orientation="Vertical">
            <my:Calendar SelectionMode="SingleDate" HorizontalAlignment="Left" />
            <Button Width="75" Height="25" HorizontalAlignment="Left" Content="OK"
             />
        </StackPanel>
        <TextBlock Text="Date: " Grid.Row="1" Grid.Column="0" />
        <TextBlock Text="Message: " Grid.Row="2" Grid.Column="0"
        Grid.ColumnSpan="2" />
    </Grid>
</UserControl>
```

9. Run the application.
    - In the HelloSilverlight – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.

> **Note:** The following illustration shows an example of the browser window. At this state, you can type text in the **Your Name** text box, you can select dates in the calendar, but you cannot use the **OK** button.

10. Close Internet Explorer.
    - In the HelloSilverlight – Windows Internet Explorer window, click the **Close** button.

> **Note:** Keep the solution open in Visual Studio, as you will build on this demonstration in the next topic.

Demonstration: How to Respond to Events in a Silverlight Application

Demonstration steps

To respond to events in a Silverlight application, you need to perform the following steps:

1. Add the **x:Name** attribute with the value of **NameTextBox**, to the **Your Name TextBox** control.
    - In the XAML view, locate the **Your Name TextBox** control, and add the **Name** attribute with a value of **NameTextBox**.

```
x:Name="NameTextBox"
```

> **Note:** The **x:Name** attribute uniquely identifies an element.

2. Add the **x:Name** attribute with the value of **DateCalendar**, to the **Calendar** control.
    - In the XAML view, locate the **Calendar** control, and add the **Name** attribute with a value of **DateCalendar**.

```
x:Name="DateCalendar"
```

3. Add the **x:Name** attribute with the value of **MessageTextBlock**, to the **Message TextBlock** control.

    - In the XAML view, locate the **Message TextBlock** control, and add the **Name** attribute with a value of **MessageTextBlock**.

```
x:Name="MessageTextBlock"
```

4. Add the **x:Name** attribute with the value of **OKButton**, to the **Button** control.

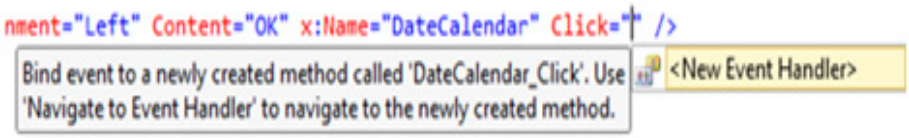
    - In the XAML view, locate the **Calendar** control, and add the **Name** attribute with a value of **OKButton**.

```
x:Name="OKButton"
```

5. Add the **Click** event handler with the value of **OKButton_Click**, to the **Button** control.

    - In the XAML view, in the **Button** control, type **Click**, and then press the TAB key.

Note: Notice that an IntelliSense window appears, as shown in the following illustration.



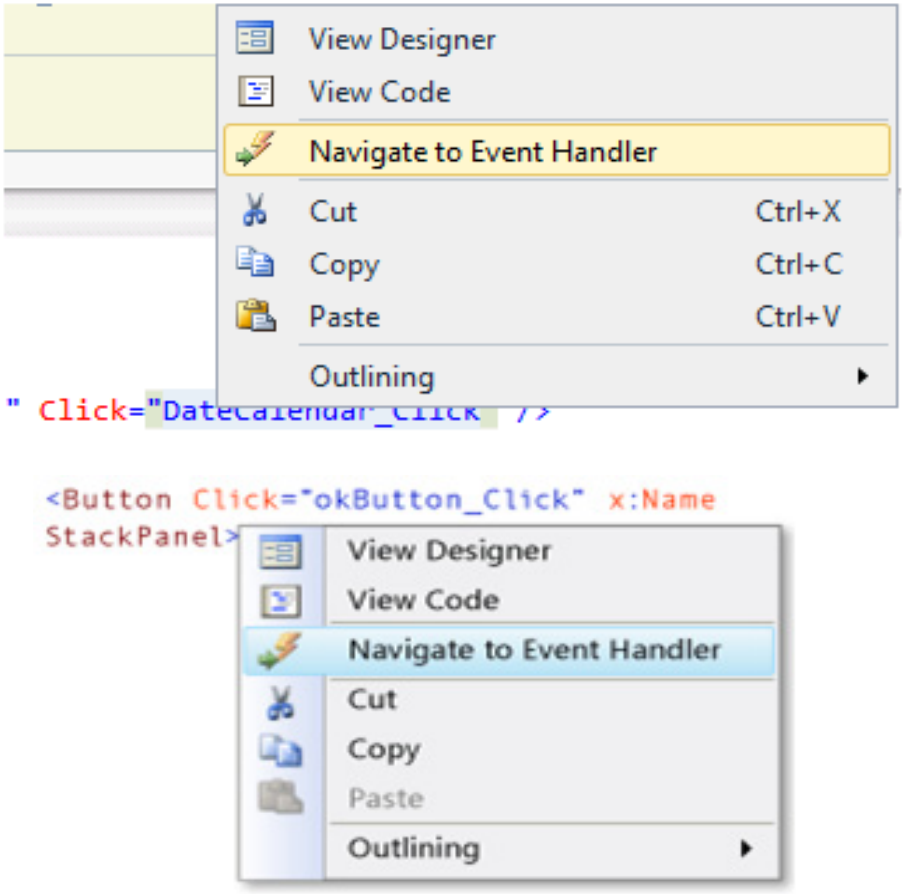
- In the IntelliSense window, double-click **<New Event Handler>**.

Note: Notice that the default name for the **Click** event handler **OKButton_Click** appears in XAML, and an event handler is created in the code-behind file.

6. Open the MainPage.xaml.vb or MainPage.xaml.cs code window by using the **Navigate to Event Handler** option.

    - In the MainPage.xaml window, right-click the markup, **Click="OKButton_Click"**, and in the shortcut menu, select **Navigate to Event Handler**.

Note: The following illustration displays the shortcut menu.



7. Build the solution.

   - In the HelloSilverlight – Microsoft Visual Studio window, on the **Build** menu, click **Build Solution**.

   Note: Building the solution ensures that IntelliSense includes the controls that you just named.

8. Add code to the **OKButton_Click** event handler, in the MainPage.xaml code-behind file, to retrieve the selected date from the **Calendar** control, and the name from the **TextBox** control. Display a message in the **Message TextBlock** control.

   - In the MainPage.xaml.cs or MainPage.xaml.vb code window, add the following code to the **OKButton_Click** event handler.

[Visual Basic]

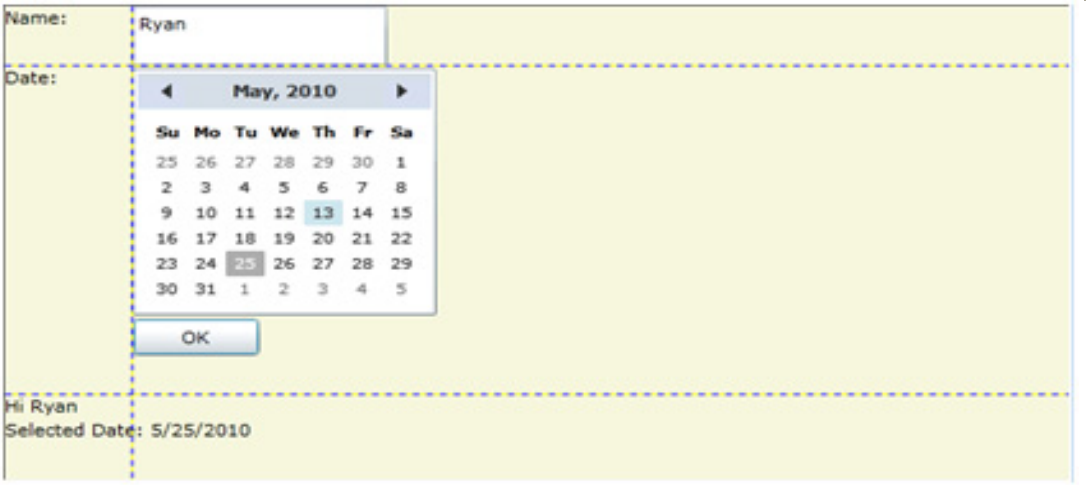
```
Dim dateString As String
If DateCalendar.SelectedDate Is Nothing Then
      dateString = "<No date selected>"
Else
      dateString = DateCalendar.DisplayDate.ToShortDateString()
End If
MessageTextBlock.Text = "Hi " & NameTextBox.Text & vbCrLf &
      "Selected Date: " & dateString
```

[Visual C#]

```
string dateString;
if (DateCalendar.SelectedDate == null)
{
      dateString = "<No date selected>";
}
else
{
      dateString = DateCalendar.DisplayDate.ToShortDateString();
}
MessageTextBlock.Text = "Hi " + NameTextBox.Text + "\n" +
      "Selected Date: " + dateString;
```

9. Run the application.
    - In the HelloSilverlight – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.
    - In the HelloSilverlight window, in the **Your Name** text box, type **Ryan**, select a date in the **Calendar**, and then click **OK**.

Note: The following illustration shows an example of the browser window with the name and date selected.



10. Close Internet Explorer.
    - In the HelloSilverlight – Windows Internet Explorer window, click the **Close** button.

Note: Keep the solution open in Visual Studio, as you will build on this demonstration in the next topic. Do answer questions at this stage.

Demonstration: How to Resize Controls in a Grid Layout

Demonstration steps

To resize controls in a Grid layout of a Silverlight application, you need to perform the following steps:

1. Change the values of the **RowDefinition Height** attributes in the **Grid.RowDefinitions** element.

- In the MainPage.xaml window, in the **Grid.RowDefinitions** element, change the values of the **RowDefinition Height** attributes as follows.

```
<RowDefinition Height="Auto"/>
<RowDefinition Height="*" MinHeight="240"/>
<RowDefinition Height="Auto"/>
```

Note: Auto-sizing indicates that the contents determine the size. Star sizing (*) indicates that the size is a weighted proportion of the remaining space.

2. Change the values of the **ColumnDefinition Width** attributes in the **Grid.ColumnDefinitions** element.

    - In the MainPage.xaml window, in the **Grid.ColumnDefinitions** element, change the values of the **ColumnDefinition Width** attributes as follows.

```
<ColumnDefinition Width="Auto" />
<ColumnDefinition Width="*"/>
```

3. Add the **Margin** attribute with a value of **10,5,10,5**, to the **Name**, **Date**, and **Message TextBlock** controls.

    - In the MainPage.xaml window, in the **Name TextBlock** control, type the following markup.

```
Margin="10,5,10,5"
```

    - In the MainPage.xaml window, in the **Date TextBlock** control, type the following markup.

```
Margin="10,5,10,5"
```

    - In the MainPage.xaml window, in the **Message TextBlock** control, type the following markup.

```
Margin="10,5,10,5"
```

Note: The **Margin** attribute with a value of **10,5,10,5**, indicates to add a 10-pixel margin on the left and right, and a 5-pixel margin on the top and bottom.

4. Add the Margin attribute with the value of **0,5,0,5**, to **TextBox**, **Calendar**, and **Button** controls.

    - In the MainPage.xaml window, in the **TextBox** control, type the following markup.

```
Margin="0,5,0,5"
```

    - In the MainPage.xaml window, in the **Calendar** control, type the following markup.

```
Margin="0,5,0,5"
```

    - In the MainPage.xaml window, in the **Button** control, type the following markup.

```
Margin="0,5,0,5"
```

5. Add the **FontSize** attribute with a value of **20**, to the **Message TextBlock** control, to increase the size of the font.

    - In the MainPage.xaml window, in the **Message TextBlock** control, type the following markup.

```
FontSize="20"
```

6. Save the changes.

    - In the HelloSilverlight – Microsoft Visual Studio window, on the **File** menu, click **Save All**.

Note: Your XAML should be similar to the following markup.

```
<UserControl xmlns:my="clr-
 namespace:System.Windows.Controls;assembly=System.Windows.Controls"
 x:Class="HelloSilverlight.MainPage"
     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
     xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
     xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
     mc:Ignorable="d"
     d:DesignHeight="300" d:DesignWidth="400">
     <Grid x:Name="LayoutRoot" Background="Beige" ShowGridLines="True">
          <Grid.RowDefinitions>
               <RowDefinition Height="Auto"/>
               <RowDefinition Height="*" MinHeight="240"/>
               <RowDefinition Height="Auto"/>
          </Grid.RowDefinitions>
          <Grid.ColumnDefinitions>
               <ColumnDefinition Width="Auto" />
               <ColumnDefinition Width="*"/>
          </Grid.ColumnDefinitions>
          <TextBlock Text="Name: " Grid.Row="0" Grid.Column="0" Margin="10,5,10,5"
          />
          <TextBox Text="Your Name" Grid.Row="0" Grid.Column="1" Width="150"
          HorizontalAlignment="Left" x:Name="NameTextBox" Margin="0,5,0,5" />
          <StackPanel Grid.Column="1" Grid.Row="1" Orientation="Vertical">
               <my:Calendar SelectionMode="SingleDate" HorizontalAlignment="Left"
               x:Name="DateCalendar" Margin="0,5,0,5" />
               <Button Width="75" Height="25" HorizontalAlignment="Left"
               Content="OK" x:Name="OKButton" Click="OKButton_Click"
               Margin="0,5,0,5" />
          </StackPanel>
          <TextBlock Text="Date: " Grid.Row="1" Grid.Column="0" Margin="10,5,10,5"
          />
          <TextBlock Text="Message: " Grid.Row="2" Grid.Column="0"
          Grid.ColumnSpan="2" x:Name="MessageTextBlock" Margin="10,5,10,5"
          FontSize="20" />
     </Grid>
</UserControl>
```

7. Run the application.

    - In the HelloSilverlight – Microsoft Visual Studio window, on the **Debug** menu, click **Start Without Debugging**.

8. Resize the browser window, and then click the **OK** button.

Note: Notice that the third row of the Grid increases in height to accommodate the text.

9. Close Internet Explorer.
    - In the HelloSilverlight – Windows Internet Explorer window, click the **Close** button.
10. Close Visual Studio 2010.
    - In the HelloSilverlight – Microsoft Visual Studio window, click the **Close** button.

Additional Reading

Overview of Silverlight

For more information about the JavaScript API, see JavaScript API for Silverlight.

For more information about the managed API, see Managed API for Silverlight.

For more information about networking and communication, see Networking and Communication.

For more information about out-of-browser support, see Out-of-Browser Support.

Installing Silverlight

To download the Microsoft Silverlight 4 Tools for Visual Studio 2010, see Microsoft Silverlight 4 Tools for Visual Studio 2010.

To download the Microsoft Silverlight plug-in, see Get Microsoft Silverlight.

For more information about WCF RIA Services, see WCF RIA Services.

Silverlight Tools for Developing Applications

For a list of the controls included in the Silverlight SDK, see Controls by Function.

For more information about the Windows Presentation Foundation and Silverlight Designer, see WPF and Silverlight Designer Overview.

For more information about the Deep Zoom Composer and to download it, see Deep Zoom Composer.

For more information about the Silverlight.js JavaScript helper file, see Silverlight.js.

For more information about the optional Silverlight.supportedUserAgent.js add-on for the Silverlight.js JavaScript helper file, see Silverlight.supportedUserAgent.js.

For more information about integrating Silverlight with a Web page, see Integrating Silverlight with a Web Page.

Module Reviews and Takeaways

Review questions and answers

1. Which framework facilitates TDD?

   ASP.NET MVC 2. 0 facilitates test TDD.

2. Which technology can you use to develop cross-browser and cross-platform enabled applications?

   You can use Silverlight to develop cross-browser and cross-platform enabled applications.

Real-world issues and scenarios

1. You want to deliver high-quality video and audio that streams to your users without installing client-based applications for playing the streamed content. How do you achieve this?

   You can create a Silverlight application that includes the **MediaElement** control.

Best practices

- When working with XAML, follow proper naming conventions, and try to copy the naming conventions that you use for your other Web development technologies, such as ASP.NET. This means that you should also add an **x:Name** attribute to every element in XAML, where possible.

Lab Review Questions and Answers

1. Which Silverlight control will you use to play a media file?

   Answer: You use the MediaElement control to play a media file.

2. How do you add a file to a Silverlight application package as a resource?

   Answer: You set the Build Action property to a value of Resource.

Resources

Contents:

Microsoft Learning

This section describes various Microsoft Learning programs and offerings.

- Microsoft Skills Assessments

  Describes the skills assessment options available through Microsoft.

- Microsoft Learning

  Describes the training options available through Microsoft — face-to-face or self-paced.

- Microsoft Certification Program

  Details how to become a Microsoft Certified Professional, Microsoft Certified Database Administrators, and more.

- Microsoft Learning Support
  - To provide comments or feedback about the course, send e-mail to support@mscourseware.com.
  - To ask about the Microsoft Certification Program (MCP), send e-mail to mcphelp@microsoft.com

Send Us Your Feedback

You can search the Microsoft Knowledge Base for known issues at Microsoft Help and Support before submitting feedback. Search using either the course number and revision, or the course title.

Note Not all training products will have a Knowledge Base article – if that is the case, please ask your instructor whether or not there are existing error log entries.

Courseware Feedback

Send all courseware feedback to support@mscourseware.com. We truly appreciate your time and effort. We review every e-mail received and forward the information on to the appropriate team. Unfortunately, because of volume, we are unable to provide a response but we may use your feedback to improve your future experience with Microsoft Learning products.

Reporting Errors

When providing feedback, include the training product name and number in the subject line of your e-mail. When you provide comments or report bugs, please include the following:

- Document or CD part number
- Page number or location
- Complete description of the error or suggested change

Please provide any details that are necessary to help us verify the issue.

Important All errors and suggestions are evaluated, but only those that are validated are added to the product Knowledge Base article.
