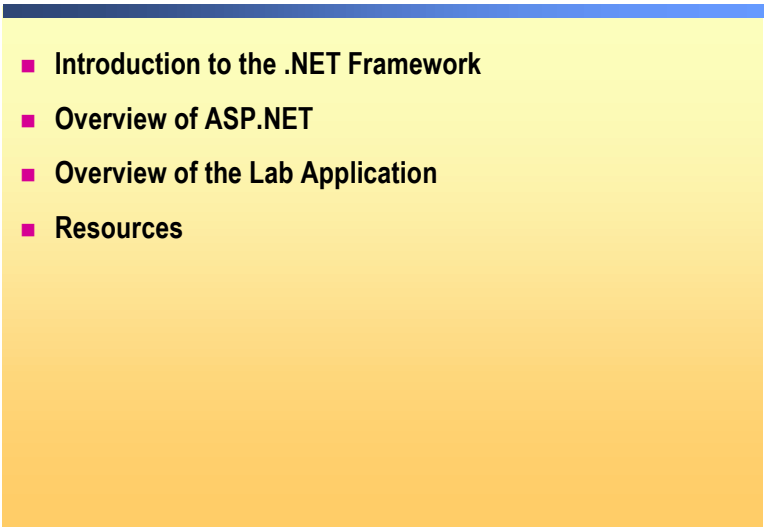
msdn® training

Module 1: Overview of the Microsoft .NET Framework

Contents

Microsoft®

Overview



Introduction

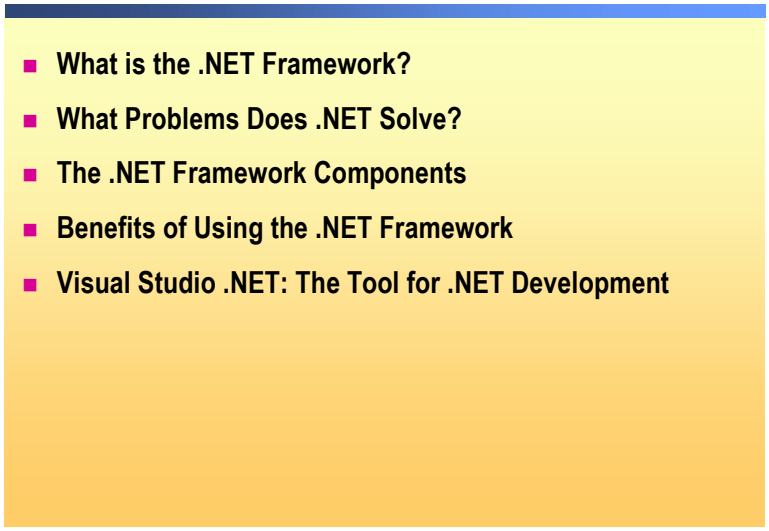
The Microsoft® .NET Framework represents a major change in the way that Web applications are built and run. Microsoft ASP.NET is one of numerous technologies that are part of the .NET Framework. In this module, you will learn about the .NET Framework and ASP.NET. You will then have an opportunity to examine the complete Web application that you will build in the labs throughout Course 2310B, *Developing Microsoft ASP.NET Web Applications Using Visual Studio .NET*.

Objectives

After completing this module, you will be able to:

- Explain the advantages of using the .NET Framework.
- Understand the key functionality and purpose of ASP.NET in developing Web applications.
- Understand the basic functionality of the Web site that you will build in the labs in Course 2310B.

Lesson: Introduction to the .NET Framework



Introduction

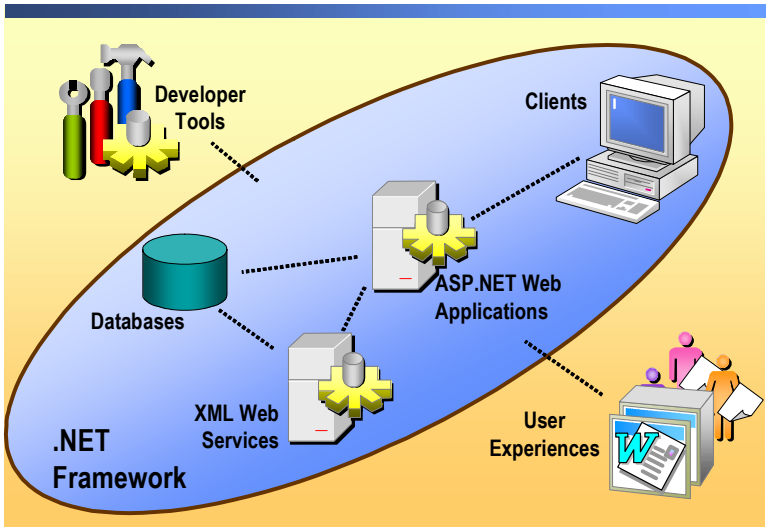
In this lesson, you will learn about the .NET Framework. You will learn about some of the problems that developers confront while developing Web applications and understand how the .NET Framework solves these problems. You will also be introduced to Microsoft Visual Studio® .NET, which is the development tool that you will use to develop Web applications with the .NET Framework.

Lesson objectives

After completing this lesson, you will be able to:

- Differentiate between Microsoft .NET and the .NET Framework.
- List the benefits of using .NET, based on the perspective of the problems that .NET solves.
- Identify the features of Visual Studio .NET.

What is the .NET Framework?



Introduction

.NET is Microsoft's development model in which software becomes platform- and device-independent, and data becomes available over the Internet. The .NET Framework is the infrastructure of .NET.

What is the .NET platform?

.NET is built from the ground up on open architecture. .NET is a platform that can be used for building and running the next generation of Microsoft Windows® and Web applications. The goal of the Microsoft .NET platform is to simplify Web development. The.NET platform consists of the following core technologies:

- The .NET Framework
- The .NET Enterprise Servers
- Building block services
- Visual Studio .NET

The .NET platform spans clients, servers, and services, and it consists of:

- A programming model that enables developers to build Extensible Markup Language (XML) Web services and applications.
- A set of building block services that are a user-centric set of XML Web services that move control of user data from applications to users. For example, Microsoft Passport is a core component of the.NET initiative that makes it easier to integrate various applications.

- A set of .NET Enterprise Servers, including Windows 2000, Microsoft SQL Server™, and Microsoft BizTalk® Server, that integrate, run, operate, and manage XML Web services and applications.
- Client software, such as Windows XP and Windows CE, which helps developers deliver a comprehensive user experience across a family of devices.
- Tools, such as Visual Studio .NET, which can be used to develop XML Web services and Windows and Web applications for an enriched user experience.

What is the .NET Framework?

The .NET Framework provides the foundation upon which applications and XML Web services are built and executed. The unified nature of the .NET Framework means that all applications, whether they are Windows applications, Web applications, or XML Web services, are developed by using a common set of tools and code, and are easily integrated with one another.

The .NET Framework consists of:

- *The common language runtime* (known hereafter as runtime). The runtime handles runtime services, including language integration, security, and memory management. During development, the runtime provides features that are needed to simplify development.
- *Class libraries*. Class libraries provide reusable code for most common tasks, including data access, XML Web service development, and Web and Windows Forms.

What Problems Does .NET Solve?

- Even with the Internet, most applications and devices have trouble communicating with each other
- Programmers end up writing infrastructure instead of applications
- Programmers have had to limit their scope or continually learn new languages

Introduction

The .NET Framework was developed to overcome several limitations that developers have had to deal with when developing Web applications, and it makes strong use of the Internet as a means for solving these limitations.

Pre-.NET issues

Even with the advent of a global, easily accessible network for sharing information (the Internet), few applications work on more than one type of client or have the ability to seamlessly interact with other applications. This limitation leads to two major problems that developers must confront:

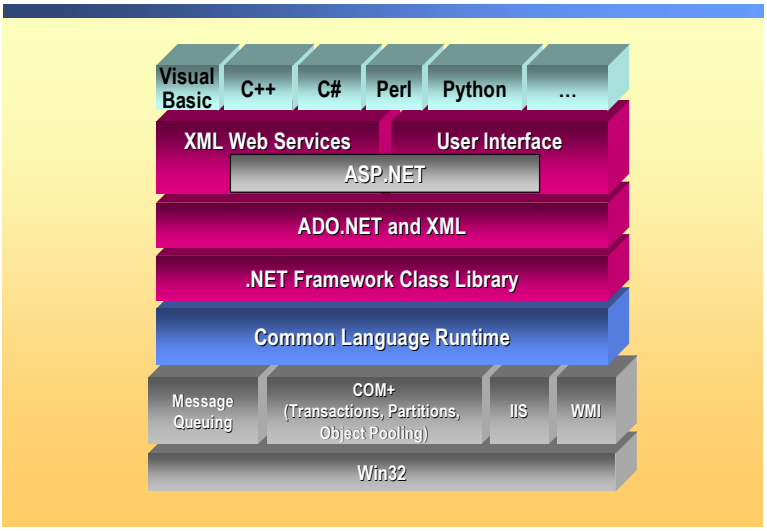
- Developers typically have to limit their scope.
- Developers spend the majority of their time rewriting applications to work on each type of platform and client, rather than spending their time designing new applications.

The .NET Framework solves the preceding two problems by providing the runtime, which is language-independent and platform-independent, and by making use of the industry-standard XML. Language independence in .NET allows developers to build an application in any .NET-based language and know that the Web application will work on any client that supports .NET.

The runtime also controls much of the application infrastructure so that developers can concentrate on the application-specific logic.

XML Web services use XML to send data, thereby ensuring that any XML-capable client can receive that data. Since XML is an open standard, most modern clients, such as computer operating systems, cellular telephones, personal digital assistants (PDAs), and game consoles, can accept XML data.

The .NET Framework Components



Introduction

The .NET Framework provides the necessary compile-time and run-time foundation to build and run .NET-based applications.

The .NET Framework

The .NET Framework consists of different components that help to build and run .NET-based applications:

- Platform Substrate

  The .NET Framework must run on an operating system. Currently, the .NET Framework is built to run on the Microsoft Win32® operating systems, such as Windows 2000, Windows XP, and Windows 98. In the future, the .NET Framework will be extended to run on other platforms, such as Windows CE.

- Application Services

  When running on Windows 2000, application services, such as Component Services, Message Queuing, Internet Information Services (IIS), and Windows Management Instrumentation (WMI), are available to the developer. The .NET Framework exposes application services through classes in the .NET Framework class library.

- .NET Framework Class Library

  The .NET Framework class library exposes features of the runtime and simplifies the development of .NET-based applications. In addition, developers can extend classes by creating their own libraries of classes.

  The .NET Framework class library implements the .NET Framework. All applications (Web, Windows, and XML Web services) access the same .NET Framework class libraries, which are held in namespaces. All .NET-based languages also access the same libraries.

- Common Language Runtime

  The common language runtime simplifies application development, provides a robust and secure execution environment, supports multiple languages, and simplifies application deployment and management.

  The common language runtime environment is also referred to as a managed environment, in which common services, such as garbage collection and security, are automatically provided.

- Microsoft ADO.NET

  ADO.NET is the next generation of Microsoft ActiveX® Data Objects (ADO) technology. ADO.NET provides improved support for the disconnected programming model. ADO.NET also provides extensive XML support.

  Note To learn more about ADO.NET, see Modules 9, 10, and 11 in Course 2310B, *Developing Microsoft ASP.NET Web Applications Using Visual Studio .NET*.

- ASP.NET

  ASP.NET is a programming framework that is built on the common language runtime. ASP.NET can be used on a server to build powerful Web applications. ASP.NET Web Forms provide an easy and powerful way to build dynamic Web user interfaces (UIs).

- XML Web Services

  XML Web services are programmable Web components that can be shared among applications on the Internet or the intranet. The .NET Framework provides tools and classes for building, testing, and distributing XML Web services.

  Note To learn more about XML Web services, see Module 13 in Course 2310B, *Developing Microsoft ASP.NET Web Applications Using Visual Studio .NET*.

- User Interfaces

  The .NET Framework supports three types of UIs:

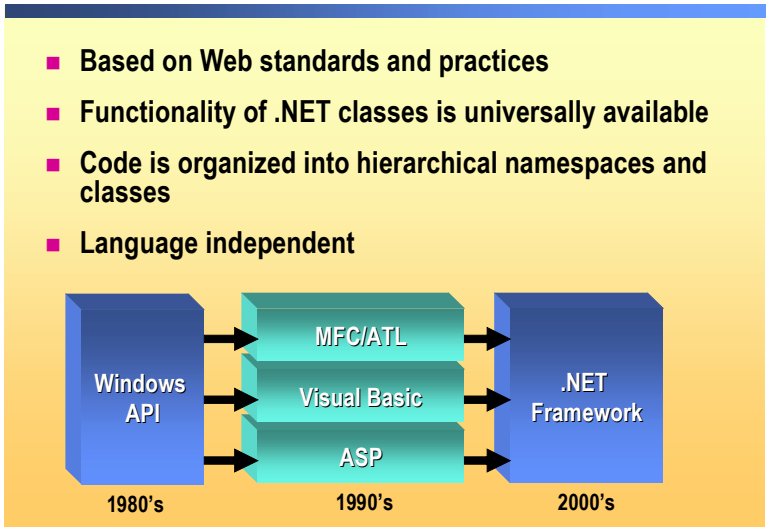
  - Web Forms, which work through ASP.NET and the Hypertext Transfer Protocol (HTTP)
  - Windows Forms, which run on Win32 client computers
  - The Command Console

- Languages

  Any language that conforms to the Common Language Specification (CLS) can run with the common language runtime. In the .NET Framework, Microsoft provides support for Microsoft Visual Basic® .NET, Microsoft Visual C++® .NET, C#, and Microsoft JScript® .NET. Third parties can provide additional languages.

  Note For more information on the .NET-based languages, see Module 3, "Using Microsoft .NET-Based Languages," in Course 2310B, *Developing Microsoft ASP.NET Web Applications Using Visual Studio .NET*.

Benefits of Using the .NET Framework



Introduction

During the early years of Windows application development, all applications were written to the Windows application programming interface (API) in C or C++.

With the advent of Visual Basic, and then the Internet, developers had to specialize in developing C and C++ (MFC/ATL) applications, Visual Basic applications, or Active Server Pages (ASP) applications. With the .NET Framework, you can use your skills to develop any type of application.

Benefits

The benefits of using the .NET Framework for developing applications include:

- Based on Web standards and practices

  The .NET Framework fully supports existing Internet technologies, including Hypertext Markup Language (HTML), HTTP, XML, Simple Object Access Protocol (SOAP), Extensible Stylesheet Language Transformation (XSLT), XML Path Language (XPath), and other Web standards.

- Designed using unified application models

  The functionality of a .NET class is available from any .NET-compatible language or programming model. Therefore, the same piece of code can be used by Windows applications, Web applications, and XML Web services.

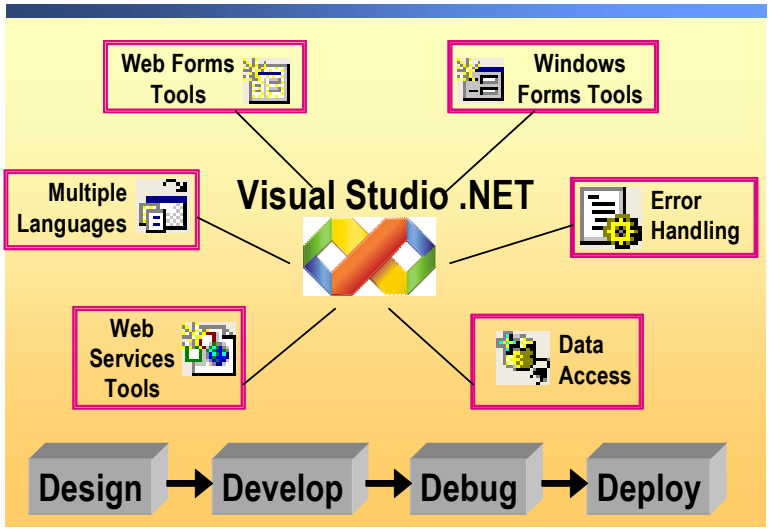
- Easy for developers to use

  In the .NET Framework, code is organized into hierarchical namespaces and classes. The .NET Framework provides a common type system, referred to as the unified type system, which can be used by any .NET-compatible language. In the unified type system, all language elements are objects. These objects can be used by any .NET application written in any .NET-based language.

- Extensible classes

  The hierarchy of the .NET Framework is not hidden from the developer. You can access and extend .NET classes (unless they are protected) through inheritance. You can also implement cross-language inheritance.

Visual Studio .NET: The Tool for .NET Development



Introduction

Visual Studio .NET constitutes the core of .NET development. Visual Studio .NET is a complete development environment in which you can design, develop, debug, and deploy your .NET applications and XML Web services.

Features of Visual Studio .NET

Visual Studio .NET, as a development tool, provides the following:

- Support for various development languages.

  Note For more information on the available .NET-based languages, see Module 3, "Using Microsoft .NET-Based Languages," in Course 2310B, *Developing Microsoft ASP.NET Web Applications Using Visual Studio .NET*.

- Tools for building Web applications, Windows applications, and XML Web services.
- Data access tools.
- Complete error handing, including local debugging, remote debugging, and tracing.

Optional practice

You will be using Visual Studio .NET throughout this course. In this practice, you will pin Visual Studio .NET to your **Start** menu, making it easily available, and then open Visual Studio .NET.

► Pin a program to the Start menu

- On the **Start** menu, click **All Programs**, point to **Microsoft Visual Studio .NET**, right-click **Microsoft Visual Studio .NET**, and then click **Pin to Start menu**.

  The Visual Studio .NET icon and name appear near the top left side of the **Start** menu.

► Start and then close Visual Studio .NET

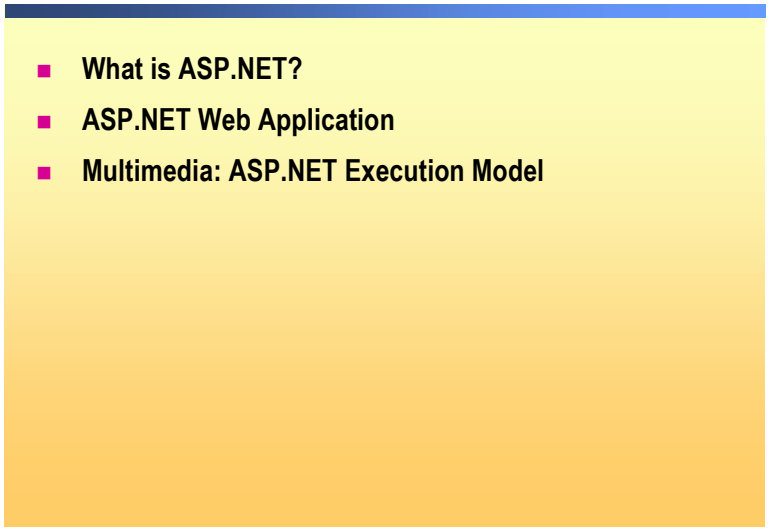
1. Click **Start**, and then on the list on the upper left side of the **Start** menu, click **Microsoft Visual Studio .NET**.

   Visual Studio .NET opens and displays the start page.

   Note You will learn more about the Visual Studio .NET IDE in Module 2, "Using Microsoft Visual Studio .NET," in Course 2310B, *Developing Microsoft ASP.NET Web Applications Using Visual Studio .NET*.

2. On the **File** menu, click **Exit** to close Visual Studio .NET.

Lesson: Overview of ASP.NET



Introduction

ASP.NET is a programming framework built on the .NET Framework that is used to build Web applications. ASP.NET Web Forms, which are part of an ASP.NET Web application, provide an easy way to build dynamic Web sites. ASP.NET also includes the needed technology to build XML Web services, which provide the building blocks for constructing distributed Web-based applications.

Lesson objectives

After completing this lesson, you will be able to:

- Explain the difference between ASP.NET Web Forms and Web applications.
- Describe the parts of an ASP.NET Web application.

What is ASP.NET?

- Evolutionary, more flexible successor to Active Server Pages
- Dynamic Web pages that can access server resources
- Server-side processing of Web Forms
- XML Web services let you create distributed Web applications
- Browser-independent
- Language-independent

Introduction

For many years, developers have been using ASP technology to build dynamic Web pages. Similar to ASP, ASP.NET runs on the Web server and provides a way for you to develop content-rich, dynamic, personalized Web sites. In addition, ASP.NET offers many improvements over ASP.

What is ASP.NET?

Developing ASP.NET Web applications in the .NET Framework is similar to developing Windows applications. The fundamental component of ASP.NET is the Web Form. A Web Form is the Web page that users view in a browser. An ASP.NET Web application comprises one or more Web Forms. A Web Form is a dynamic page that can access server resources.

For example, a traditional Web page can run script on the client to perform basic tasks. An ASP.NET Web Form, conversely, can also run server-side code to access a database, to generate additional Web Forms, or to take advantage of built-in security on the server.

In addition, because an ASP.NET Web Form does not rely on client-side scripting, it is not dependent on the client's browser type or operating system. This independence allows you to develop a single Web Form that can be viewed on practically any device that has Internet access and a Web browser.

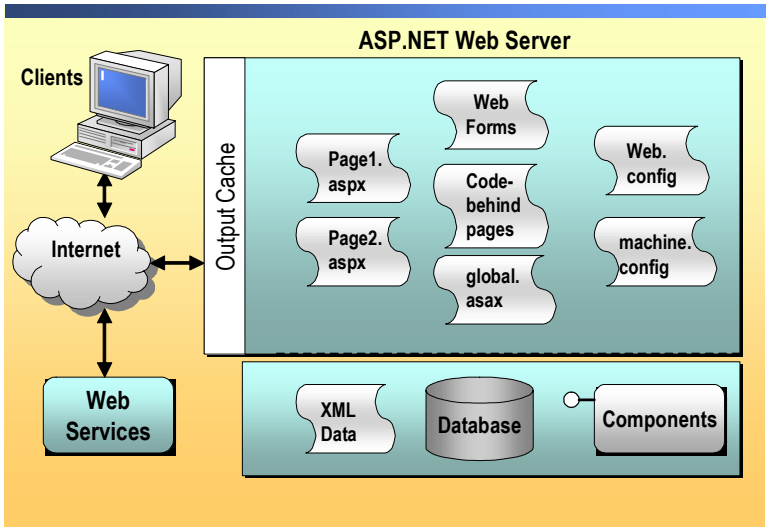
Because ASP.NET is part of the .NET Framework, you can develop ASP.NET Web applications in any .NET-based language.

XML Web services

The ASP.NET technology also supports XML Web services. XML Web services are distributed applications that use XML for transferring information between clients, applications, and other XML Web services.

Note You will learn how to consume and create XML Web services in Module 13, "Consuming and Creating XML Web Services," in Course 2310B, *Developing Microsoft ASP.NET Web Applications Using Visual Studio .NET.*

ASP.NET Web Application



Introduction

An ASP.NET Web application contains different parts and components. Creating ASP.NET Web applications involves using and working with all of its parts and components.

In this topic, you will learn what constitutes an ASP.NET application.

Part of an ASP.NET Web application

The parts of an ASP.NET Web application include:

- *Web Forms, or .aspx pages*

  Web Forms and .aspx pages provide the UI for the Web application.

- *Code-behind pages*

  Code-behind pages are associated with Web Forms and contain the server-side code for the Web Form.

- *Configuration files*

  Configuration files are XML files that define the default settings for the Web application and the Web server. Every Web application has one Web.config configuration file. In addition, each Web server has one machine.config file.

- *Global.asax file*

  Global.asax files contain the needed code for responding to application-level events that are raised by ASP.NET.

- *XML Web service links*

  XML Web service links allow the Web application to send and receive data from an XML Web service.

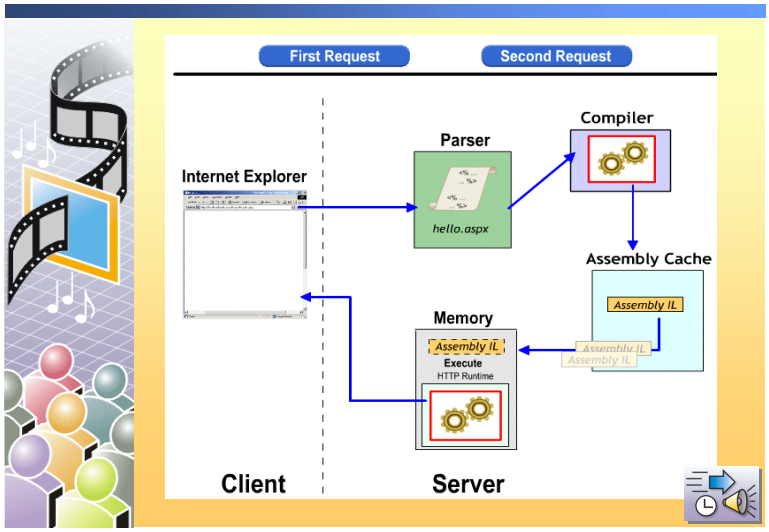
- *Database connectivity*

  Database connectivity allows the Web application to transfer data to and from database sources.

- *Caching*

  Caching allows the Web application to return Web Forms and data more quickly after the first request.

Multimedia: ASP.NET Execution Model



Introduction

In this animation, you will see how ASP.NET works to send information to a requesting client.

First request

When the client requests a Web page for the first time, the following set of events take place:

1. The client browser issues a GET HTTP request to the server.
2. The ASP.NET parser interprets the source code.
3. If the code was not already compiled into a dynamic-link library (DLL), ASP.NET invokes the compiler.
4. Runtime loads and executes the Microsoft intermediate language (MSIL) code.

Second request

When the user requests the same Web page for the second time, the following set of events take place:

1. The client browser issues a GET HTTP request to the server.
2. Runtime loads and immediately executes the MSIL code that was already compiled during the user's first access attempt.

Lesson: Overview of the Lab Application

Introduction

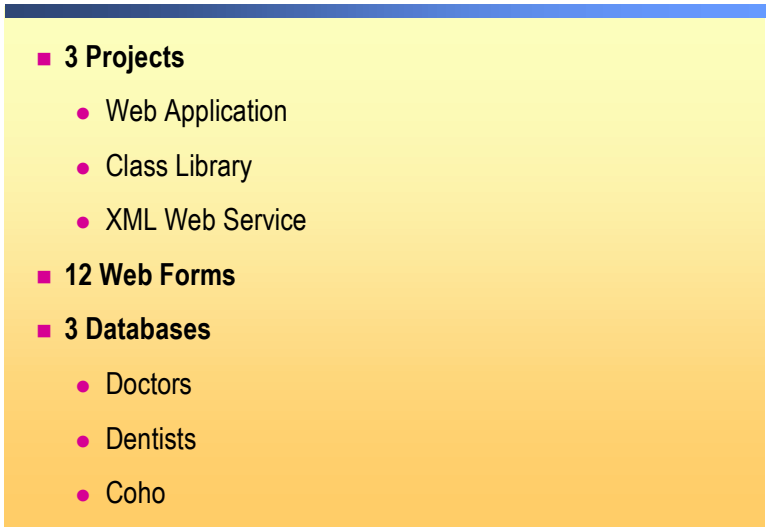
In this lesson, you will be introduced to the lab application that you will build throughout the remainder of Course 2310B, in Labs 2 through 17. You will also have the opportunity to explore a complete version of the lab application and understand some of its functionality.

Lesson objective

After completing this lesson, you will be able to:

- Understand the basic functionality and features of the Lab application that you will build in the remaining labs in this course.
- List the software that is required to run the labs in this course.

Lab Application Setup



Introduction

In the labs in this course, you will build a complete ASP.NET Web application that simulates an internal Web site for the fictitious company, Coho Winery. This fictitious Web site, which you will build on your computer, provides information to employees about their benefit options.

Details

The Benefits Web application that you will develop includes the following:

- Three projects, including a Web application, a class library, and an XML Web service.
- 12 Web Forms that are used for displaying the employee benefits information.
- Three SQL Server databases.

Note Throughout this course, you will be given the choice between developing lab solutions by using either Visual Basic .NET or C#.

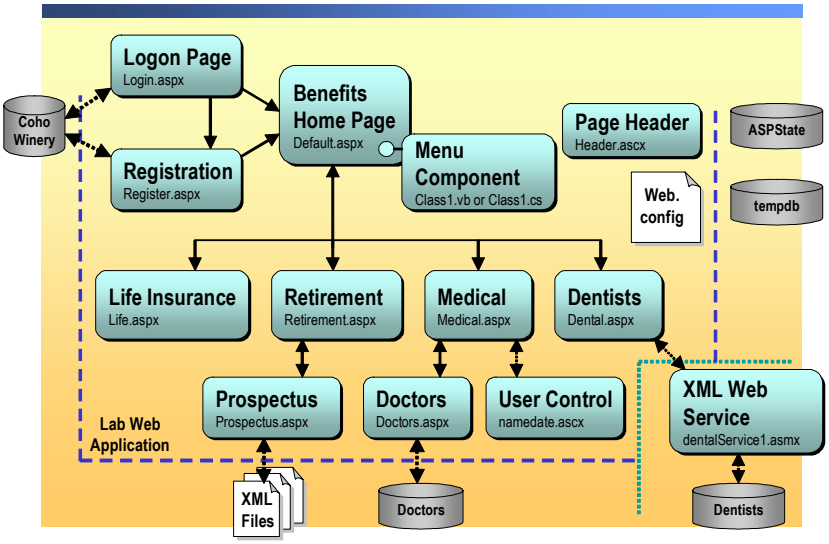
The Visual Basic .NET and C# components of the Benefits Web application are listed in the following table.

Visual Basic .NET Page	C# Page	Description	Labs
default.aspx default.aspx.vb	default.aspx default.aspx.cs	The home page. Calls the BenefitsList component and lists the benefits options in a **CheckBoxList** control.	4, 5, 14
dental.aspx dental.aspx.vb	dental.aspx dental.aspx.cs	Calls the XML Web service to get a listing of dentists.	13
doctors.aspx doctors.aspx.vb	doctors.aspx doctors.aspx.cs	Lists the primary care physicians from the doctors SQL Server database.	9, 10, 11, 15
dsDoctors.vb dsDoctors.xsd	dsDoctors.cs dsDoctors.xsd	Contains the schema of the **DataSet** that is used in the doctors.aspx Web form.	9
growth.xml lgcap.xml midcap.xml smcap.xml	growth.xml lgcap.xml midcap.xml smcap.xml	XML files with the prospectus information.	12
header.ascx	header.ascx	User control that is the header of each page.	4, 5, 15
life.aspx life.aspx.vb	life.aspx life.aspx.cs	Input form for life insurance data. Uses validation controls.	4, 7, 14
login.aspx login.aspx.vb	login.aspx login.aspx.cs	Logon page for the Web site.	17
medical.aspx medical.aspx.vb	medical.aspx medical.aspx.cs	Input form for medical data. The page used for selecting primary care physician from the doctors.aspx page.	8, 15
mutual_funds.xml	mutual_funds.xml	XML file containing the list of mutual funds.	12
namedate.ascx namedate.ascx.vb	namedate.ascx namedate.ascx.cs	User control that is used to gather name and birth date information. Used on the medical.aspx page.	8, 14
nestedData.aspx nestedData.aspx.vb	nestedData.aspx nestedData.aspx.cs	Allows you to create a nested XML file from the several tables that comprise the Doctor database.	12
prospectus.aspx prospectus.aspx.vb	prospectus.apsx prospectus.aspx.cs	Displays a retirement fund prospectus from a given XML file by using a style sheet file.	12
prospectus_style.xsl	prospectus_style.xsl	Style sheet file for displaying the prospectus.	12
register.aspx register.aspx.vb	register.aspx register.aspx.cs	Adds new users to the Coho Winery database.	16
retirement.aspx retirement.aspx.vb	retirement.aspx retirement.aspx.cs	Displays XML data from the mutual_funds.xml file in an HTML format.	12, 14
securitytest.aspx securitytest.aspx.vb	securitytest.aspx securitytest.aspx.cs	Displays the identity and the authentication method that is used to authenticate the current user, to test the security system.	16

(*continued*)

Visual Basic .NET Page	C# Page	Description	Labs
signout.aspx	signout.aspx	Allows the user to sign out of the system.	16
signout.aspx.vb	signout.aspx.cs		
Web.config	Web.config	This is a configuration file.	14, 15, 16
BenefitsListVB component project	BenefitsListCS component project	This project contains a Visual Basic .NET or C# component named Class1.vb or Class1.cs that returns a list of benefit options.	3
DentalService Web service project	DentalService Web service project	This project contains an XML Web service named DentalService.asmx (DentalService.asmx.vb and DentalService.asmx.cs). This XML Web service reads information from the dentists SQL Server database and returns the information as a **DataSet**.	13

Demonstration: The Lab Solution



Introduction

In this demonstration, the instructor will show you the features and functionality of the Benefits Web application.

The default.aspx Web Form is the home page for the Benefits Web application. From this page, users can browse to other Web Forms, including to pages to enter information about specific benefits, such as Life Insurance, Retirement, Medical, and Dental coverage.

► Run the application

1. On the **Start** menu, click **Visual Studio .NET**.
2. Open **2310LabApplication**.

   Point out the three projects by using Solution Explorer. Each project is provided in Visual Basic .NET and C# versions.

   Point out the databases by using Server Explorer.
3. Build **2310LabApplication**.
4. Open Microsoft Internet Explorer and navigate to http://localhost/BenefitsVB/default.aspx to demonstrate the Visual Basic .NET solution and navigate to http://localhost/BenefitsCS/default.aspx to demonstrate the C# solution.

   The Web application is secured, so you are redirected to the login.aspx page to log in. This is set up in the Web.config file. The login.aspx page compares user data to the user names and passwords that are stored in the Coho SQL Server database.

Note If you receive a SQL permission error when trying to browse the Benefits Web site, open Windows Explorer, browse to C:\Program Files\MSDNTrain\2310\LabFiles\Lab16VB\Starter and then double-click **Lab16.bat**. Close Windows Explorer, and then refresh Microsoft Internet Explorer.

5. Click **Click here!**

   Because you are a new user, the register.aspx page is used to add yourself to the list of registered users for the Benefits Web application.

6. Enter user information and then click **Save**.

   After the register.aspx page adds you to the database and logs you on, the default.aspx page is displayed because that is the page you originally requested.

   The default.aspx page has a user control which is a page banner with links across the top. This user control reads from the same component as the list of check boxes on the page.

7. Select some check boxes and click **Submit**.

   When **Submit** is clicked, the page reads which check boxes are selected and displays an output list.

8. Click **Life Insurance**.

   The Life Insurance page uses server controls, validation controls, and a summary validation control. A calendar control displays todays date.

9. Enter incorrect information and click **Save**.

   The Error messages are displayed in a Validation Summary control.

10. Enter correct information and click **Save**.

11. Click **Medical**.

    This page uses a user control and session variables to display the same name and birth date as you entered on the Life Insurance page.

12. Click **Select a doctor**.

    This link redirects you to the doctors.aspx page. The doctors.aspx page uses a database connection and **DataGrid** control. The doctors.aspx page calls stored procedures and performs sorting and paging of the data.

13. Click **Select** to select a doctor, and then click **Submit**.

    The selected doctor information is passed back to the medical.aspx page in the Uniform Resource Locator (URL).

14. Click **Retirement Account**.

    This page obtains its information from various XML data documents.

    A page counter displays the number of visits that have been made to the page.

15. Click **Dental**.

    The dental.aspx page connects to the DentalService XML Web service, which reads a list of dentists from the **dentists** SQL Server database.

Lesson: Resources

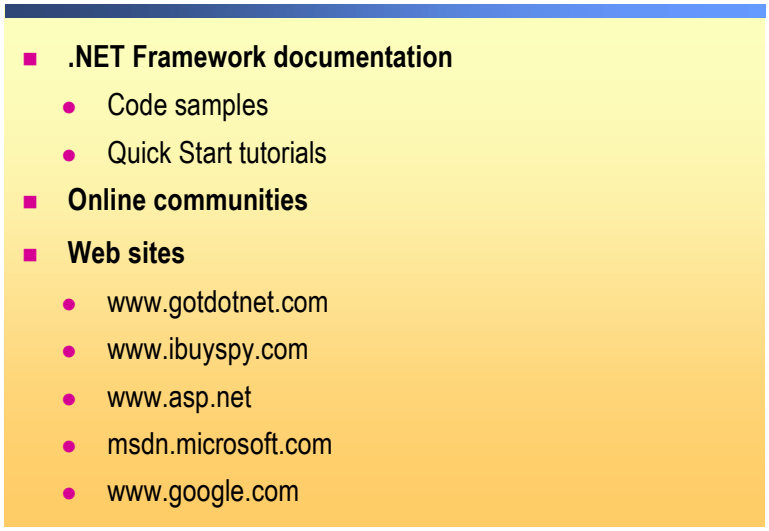
Introduction

In this lesson, you will learn about some of the resources that are available to you as you develop Web applications in Visual Studio .NET.

Lesson objective

After completing this lesson, you will be able to find the information that you need for developing ASP.NET Web applications.

.NET Resources



Resources

While learning to develop in .NET, you may need to research solutions to problems. The following list of resources is intended to give you a starting point for finding more information about .NET, along with answers to specific questions:

- *The .NET Framework documentation*. The .NET Framework documentation is found in two places.
  - On a computer with Visual Studio .NET installed, you can access the documentation from within Visual Studio .NET (on the **Help** menu, or by pressing **F1**).
  - Another way to view the documentation is on the **Start** menu, point to **All Programs**, point to **Microsoft .NET Framework SDK**, and then click **Documentation**.

  The .NET Framework documentation contains code samples in many languages, in addition to Quick Start tutorials.
- *Online communities*. Online communities are .NET-specific newsgroups.
- *External Web sites*. There are a few external Web sites that provide in-depth knowledge about .NET development:
  - The first site, http://www.gotdotnet.com, provides detailed developer information, code samples, links to.NET communities, and more.
  - The second site, http://www.ibuyspy.com, is a fictitious company store that shows how ASP.NET development can be used for e-commerce. The http://www.ibuyspy.com Web site is part of the more general http://www.asp.net site, which is an entire Web site that is dedicated to ASP.NET developers.
  - You can also access the resources that are available at http://www.msdn.microsoft.com.
  - The Web site http://www.google.com is another good source to search for information on .NET development.

Review

- Introduction to the .NET Framework
- Overview of ASP.NET
- Overview of Lab Application
- Resources

1. What is the Microsoft .NET Framework?

   The .NET Framework is a set of technologies that are designed to transform the Internet into a full-scale distributed computing platform. The .NET Framework provides new ways to build applications from collections of XML Web services. The .NET Framework fully supports the existing Internet infrastructure (HTTP, XML, and SOAP).

2. What are the core technologies in the .NET platform?

   The .NET Framework, .NET Enterprise Servers, .NET building block services, and Visual Studio .NET.

3. List the components that comprise the .NET Framework.

   Common language runtime, .NET Framework class library, data and XML, XML Web services, Web Forms, and Windows Forms.

4. What is the purpose of the common language runtime?

   The common language runtime provides an environment in which you can execute code. The runtime also provides a compiler, language independence and portability. The runtime also performs garbage collection.

5. What is the purpose of Common Language Specification?

   The CLS defines a set of features that all .NET-compatible languages should support.

6. What is an XML Web service?

   An XML Web service is a programmable Web component that can be shared among Web applications on the Internet or the intranet.

7. What is a managed environment?

   A managed environment is one in which the environment provides services, such as garbage collection, security, and other similar features.