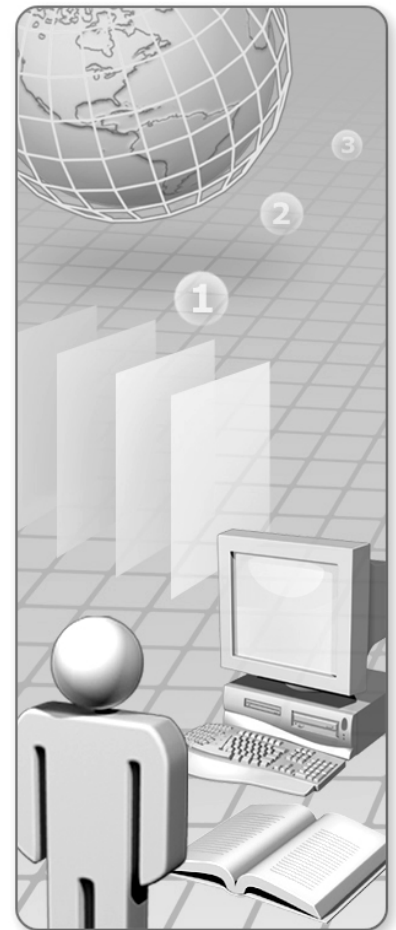


# Unit 8: Deploying a Web Application

---

## Contents

Overview	1
The Copy Web Site Utility	2
The Publish Web Site Utility	4
Windows Installer Setup Packages	6
Lab Scenario	8
Lab Tasks and Objectives	9
Lab: Deploying a Web Application	10
Lab Discussion	19



# Overview



- The Copy Web Site Utility
- The Publish Web Site Utility
- Windows Installer Setup Packages
- Lab Scenario
- Lab Tasks and Objectives
- Lab: Deploying a Web Application

## Introduction

This unit describes three different ways to deploy Web applications:

- Using the Copy Web Site utility to deploy a Web application in a non-compiled state
- Using the Publish Web Site utility to deploy a precompiled version of the Web application
- Building Microsoft® Windows® Installer packages to create a redistributable application with full setup logic

## Objectives

After completing this unit, you will be able to:

- Describe how to use the Copy Web Site utility to deploy a Web application.
- Describe how to use the Publish Web Site utility to precompile and deploy a Web application.
- Describe how to build Windows Installer packages for deploying a Web application.
- Deploy a Web application by using the Copy Web Site utility.
- Precompile and deploy a Web application by using the Publish Web Site utility.
- Build and run a Windows Installer setup application for deploying a Web application.

# The Copy Web Site Utility



- **Connecting to the Destination Web Site**
  - File-based Web site
  - Local IIS Web site
  - FTP site
  - Remote Web site
- **Copying Source Files to the Destination**
  - ASPX files
  - Code-behind files
  - Other Web files
- **Synchronizing Copied and Source Files**

## Introduction

The Copy Web Site utility enables you to deploy a Web site quickly and easily by copying the source files for the Web application, keeping the source files synchronized with the original development site. The Copy Web Site utility is integrated into the Microsoft Visual Studio® 2005 development environment.

## Connecting to the Destination Web Site

The Copy Web Site utility enables you to specify the destination for the copy operation by providing a connection that you can configure for your specific requirements. The destination can be any of the following types of location:

- A file-based Web site
- A local IIS Web site
- An FTP site
- A remote Web site

## Copying Source Files to the Destination

The Copy Web Site utility copies the files that you select to the destination. If the Web site contains source files that do not compile, these will not prevent it from being deployed, although it will not run successfully. You should therefore ensure that you can successfully build your Web application before deploying it by using the Copy Web Site utility.

You can use the Copy Web Site utility to copy any source files used to build the Web site, including:

- ASPX files
- Code-behind files
- Other Web files (such as static HTML files, images, and so on)

After you have copied files, you can edit them at the destination site if you need to make further changes. This can be useful if you are deploying the Web application from a development environment to a testing or staging environment.

Be aware that other users will be able to view and change your source code files. If you do not want to provide access to the source code, you should choose a different deployment mechanism, such as the Publish Web Site utility.

## Synchronizing Source and Copied Files

After you have deployed a Web site by using the Copy Web Site utility, you can keep the source files and the deployed files synchronized. The Copy Web Site utility displays the status of all files, both in the source application and in the deployed Web site, and enables you to view the status of all the files. If a file has changed in either location, the Copy Web Site utility enables you to synchronize those changes so that both the source and the destination are kept up-to-date.

## The Publish Web Site Utility



- **Specifying the Destination Web Site**
  - File-based Web site
  - Local IIS Web site
  - FTP site
  - Remote Web site
- **Precompiling the Web Application**
  - Compiled code DLLs
  - Updateable ASPX files
- **Deploying the Precompiled Web Application**

### Introduction

The Publish Web Site utility enables you to deploy a compiled Web site without needing to copy the source files. Similar to the Copy Web Site Utility, the Publish Web Site utility is integrated into the Visual Studio 2005 development environment.

### Specifying the Destination Web Site

The Publish Web Site utility enables you to specify the destination for the publish operation by providing a connection that you can configure for your specific requirements. The destination can be any of the following types:

- A file-based Web site
- A local IIS Web site
- An FTP site
- A remote Web site

## Precompiling the Web Application

When you use the Publish Web Site utility, the application is built by Visual Studio 2005, and the source code in the Web site is compiled into DLL assemblies. If the source code contains errors, the build process will fail and you will be informed of the problem. This enables you to resolve any compilation errors before the Web site is actually deployed to the destination site.

As well as precompiling the source code for the Web application, the Publish Web Site utility also enables you to choose whether the users will be able to edit the ASPX files at the destination site. If you choose to prevent editing at the destination site, the Publish Web Site utility removes the markup from the ASPX files, and instead creates compiled DLL assemblies that contain definitions of the markup for each page. The ASPX files are still deployed, but they simply act as stubs for the DLL versions of the pages.

## Deploying the Precompiled Web Application

After you have chosen the destination for the published Web site and have specified whether the ASPX pages will be updateable, the compiled DLL assemblies and the ASPX pages are deployed. The publishing operation also deploys all other Web files (such as images and HTML files) that the Web application uses.

If you need to modify the code for a deployed Web site, you make changes to the source Web application and then re-publish the site. Similarly, if you need to make cosmetic changes to the deployed site, you will need to make those changes in the source Web application and then re-publish the site (unless you have specified that users can modify the ASPX files at the destination site, in which case they can be edited locally).

# Windows Installer Setup Packages



- **Creating a Web Setup Project**
  - MSI file
  - Setup.exe
- **Contents of an MSI Package**
  - Web files to be deployed
  - User interface for the installer
  - Registry settings
  - Custom actions to run as part of the setup
  - Launch conditions that control installation
- **Deploying the MSI Package**

## Introduction

You can use Visual Studio 2005 to create Windows Installer packages to package and deploy a Web application.

## Creating a Web Setup Project

The easiest way to create a setup application for a Web application is to add a Web Setup project to the Visual Studio 2005 solution that contains the Web site. You can add the contents of the Web site to the Web Setup project.

The setup project creates an MSI file that contains the output of the Web site. You can also choose whether to create a bootstrap program, setup.exe. The bootstrap program contains Windows Installer files that ensure that the MSI can run on the target computer.

## Contents of an MSI Package

The MSI file contains packaged versions of the Web files to be deployed. It also contains definitions of the following setup elements:

- The user interface for the installer. You can define a wide range of user interface properties in the Visual Studio 2005 IDE, from setting simple properties such as the title of the setup application to defining the number and type of dialogs that comprise the setup application.
- Registry settings to be deployed with the Web application (if required). You can specify any registry settings that must be added or edited to support the Web application when it is deployed.
- Custom actions to run as part of the setup. Custom actions are optional, advanced setup elements that contain code that must be run as part of the installation. For example, you can include a custom action that creates a database as part of the setup.
- Launch conditions that control installation. Launch conditions optionally control whether specific elements of the installation will proceed. For example, you can include a condition that requires a specific version of the Microsoft .NET Framework. If that condition is not met, you can display a message to the user performing the installation that the installation cannot continue, and then you can roll back the installation.

## Deploying the MSI Package

After you have built the Windows Installer package, you can deploy the Web site by copying the installation files to the destination computer. If you included the bootstrap program in the Web Setup project, you can then simply run `setup.exe` and step through the setup application to install the Web site. If you did not include the bootstrap program in the Web Setup project, you can run the MSI file instead. However, this requires that the destination computer be configured correctly to enable the MSI file to run.



## Lab Scenario



### Adventure Works B2C Web Site

You are a developer in the Adventure Works organization, a fictitious bicycle manufacturer. You have been asked to assist in the development of the Business-to-Consumer (B2C) Web application and a related Business-to-Employee (B2E) extranet portal.

Decisions on the design of the application have already been made. You have been asked to carry out a number of specific tasks in order to implement various elements of this design. As part of the first phase of the B2C development, you have been asked to investigate the deployment options for the Adventure Works Web application.

### Resource Toolkit Scenarios

Before you start work on the lab, you should review the Scenario tab in the Resource Toolkit. The instructor will lead a group discussion of the scenarios for this lab.

## Lab Tasks and Objectives



Task	Objectives
<b>Deploy the source files for a Web site by using the Copy Web Site utility</b>	<ul style="list-style-type: none"> <li>• Copy the source files for a Web application</li> <li>• Edit the deployed Web site</li> <li>• Synchronize the source files for the Web application with the deployed Web site</li> </ul>
<b>Deploy the precompiled version for a Web site by using the Publish Web Site utility</b>	<ul style="list-style-type: none"> <li>• Identify and fix compilation errors that prevent the publishing of a Web site</li> <li>• Publish a Web site</li> <li>• Make cosmetic changes to the source application and re-publish the Web site</li> </ul>
<b>Create and run a Windows Installer package for deploying a Web site</b>	<ul style="list-style-type: none"> <li>• Create a Windows Installer package for a Web application</li> <li>• Design the user interface for the Windows Installer package</li> <li>• Run the Windows Installer package to deploy the Web site</li> </ul>

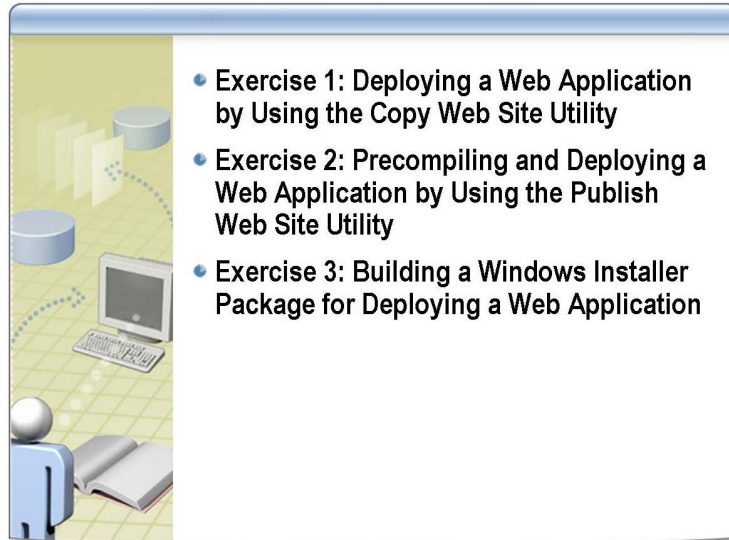
### Lab Tasks

There are three tasks in this lab. Each one is designed to help you achieve one or more learning objectives. Resources are provided in the Resource Toolkit to help you complete the tasks. You should try to complete all of the tasks.

The tasks in this lab are:

- *Deploy the source files of a Web Site by using the Copy Web Site utility.* In this task, you will use the Copy Web Site utility to deploy the source files of a Web application. You will make changes to the markup and the code of the deployed Web site. You will then synchronize the source files with the updated deployed files. The resources for this task are titled:
  - *Features of the Copy Web Site Utility*
  - *How to: Deploy a Web Site by Using the Copy Web Site Utility*
- *Deploy the precompiled version of a Web Site by using the Publish Web Site utility.* In this task, you will use the Publish Web Site utility to attempt to deploy a Web application. You will see that compilation errors prevent the Web site from being published, and you will then resolve the compilation errors and publish the Web site again. You will identify cosmetic changes that must be made to the published Web site and will make those changes in the source files. You will then re-publish the site. The resources for this task are titled:
  - *Features of the Publish Web Site Utility*
  - *How to: Precompile and Deploy a Web Site by Using the Publish Web Site Utility*
- *Create and run a Windows Installer package for deploying a Web site.* In this task, you will create a Windows Installer package for deploying a Web application. You will specify settings for the user interface for the Windows Installer package, and you will add a license agreement file and associated dialog that presents the license agreement to the user. Finally you will run the Windows Installer package to deploy the Web site. The resources for this task are titled:
  - *Features of Windows Installer Setup Projects*
  - *How to: Create a Windows Installer Package for Deploying a Web Application*

## Lab: Deploying a Web Application



After completing this lab, you will be able to:

- Deploy a Web site by using the Copy Web Site utility.
- Deploy a Web site by using the Publish Web Site utility.
- Deploy a Web site by creating and running a Windows Installer package.

Estimated time to complete this lab: **90 minutes**



**Important** You can choose to perform this workshop by using either Microsoft Visual Basic® or Microsoft Visual C#®. Code samples and lab solutions are provided in both languages.

---

### Lab Solutions

There are Visual Basic and Visual C# solution files associated with the labs in this workshop. You can find the lab solution files in the folder E:\Labfiles\Solution on the virtual machines.

### Lab Setup

For this lab, you will use the available Microsoft® Virtual PC environment. Before you begin the lab, you must:

1. Start the **2543B-LON-DEV-01-08** virtual machine.
2. Log on to the **2543B-LON-DEV-01-08** virtual machine with the user name **Administrator** and the password **Pa\$\$w0rd**.
3. If the **Get Connected** dialog box for ActiveSync appears, click **Cancel**.




No additional setup is required for this lab.

## Exercise 1


### Deploying a Web Application by Using the Copy Web Site Utility

In this exercise, you will use the Copy Web Site utility to deploy the source files for a Web application. You will make changes to the markup and the code of the deployed Web site. You will then synchronize the source files with the updated deployed files.


#### Scenario

Tasks	Supporting information
1. Open the Adventure Works Web site.	 <b>Note</b> Ensure that you have logged on as <b>Administrator</b> (rather than as <b>Student</b> ) for this lab. If the <b>Get Connected</b> dialog box for ActiveSync appears, click <b>Cancel</b> . <ul style="list-style-type: none"> <li>a. If you want to complete this lab by using Visual Basic:               <ul style="list-style-type: none"> <li>• Open the <b>VB.sln</b> solution in the <b>E:\Labfiles\Starter\VB\CopyWeb</b> folder.</li> </ul> </li> <li>b. If you want to complete this lab by using Visual C#:               <ul style="list-style-type: none"> <li>• Open the <b>CS.sln</b> solution in the <b>E:\Labfiles\Starter\CS\CopyWeb</b> folder.</li> </ul> </li> </ul>
2. Create a connection to a new virtual directory as the destination for copying the Web site.	<ul style="list-style-type: none"> <li>a. Use the <b>Copy Web Site</b> command from the <b>Website</b> menu to copy the <b>CopyWeb</b> project. Click the <b>Connect</b> button near the top of the <b>Copy Web</b> window.</li> <li>b. Use the <b>Open Web Site</b> dialog box to perform the following tasks:               <ul style="list-style-type: none"> <li>• Connect to <b>Local IIS</b>.</li> <li>• Create a new virtual directory in the <b>Default Web Site</b> with an <b>Alias name</b> of <b>CopyWeb</b>, and a <b>Folder</b> of <b>E:\Labfiles\wwwroot\CopyWeb</b>.</li> <li>• Select the newly created <b>CopyWeb</b> virtual directory as the Web site to open.</li> </ul> </li> </ul>  See the following resources in the Resource Toolkit: <ul style="list-style-type: none"> <li>▪ “Features of the Copy Web Site Utility”</li> <li>▪ “How to: Deploy a Web Site by Using the Copy Web Site Utility”</li> </ul>
3. Select files to copy to the destination Web site.	<ul style="list-style-type: none"> <li>a. Select all files and folders in the <b>Source Web Site</b> list.</li> <li>b. Copy the selected files and folders from the source Web site to the remote Web site.</li> </ul>  See the resource in the Resource Toolkit, “How to: Deploy a Web Site by Using the Copy Web Site Utility.”

*(continued)*

Tasks	Supporting information
4. Browse to the deployed Web site.	<ol style="list-style-type: none"> <li>Start Microsoft Internet Explorer, and then browse to <b>http://localhost/CopyWeb</b>.</li> <li>Notice that the time displayed near the top left of the page includes hours, minutes, and seconds.</li> <li>Notice also that the text for the following headings is displayed in dark blue: <ul style="list-style-type: none"> <li>Providing Quality Bikes and Accessories</li> <li>Building Biking Communities</li> </ul> </li> <li>Leave the Internet Explorer running—you will use it later in this exercise.</li> </ol>
5. Edit code for a deployed Web site file.	<ol style="list-style-type: none"> <li>Using Windows Explorer, navigate to the <b>E:\Labfiles\wwwroot\CopyWeb</b> folder.</li> <li>If you are working with Visual Basic, open the <b>TopLevel.master.vb</b> file in Notepad.</li> <li>If you are working with Visual C#, open the <b>TopLevel.master.cs</b> file in Notepad.</li> </ol> <p> <b>Note</b> Because you are emulating the scenario in which the destination Web site is on a <i>remote</i> server, you will edit the file in Notepad. The scenario emulates the situation in which Visual Studio is <i>not</i> present on the remote server.</p> <ol style="list-style-type: none"> <li>Edit the code in the <b>Page_Load</b> method so that the <b>Time</b> is displayed using the <b>ToShortTimeString</b> method, rather than the existing <b>ToLongTimeString</b> method.</li> <li>Save and close the file.</li> </ol>
6. Edit markup for a deployed Web site file.	<ol style="list-style-type: none"> <li>Open the <b>Default.aspx</b> file in the <b>E:\Labfiles\wwwroot\CopyWeb</b> folder by using Notepad.</li> <li>Locate the <b>div</b> element that contains the text <i>Providing Quality Bikes and Accessories</i>.</li> <li>Edit the <b>div</b> element so that its <b>color</b> attribute is <b>Lime</b>.</li> <li>Locate the <b>span</b> element that contains the text <i>Building Biking Communities</i>.</li> <li>Edit the <b>span</b> element so that its <b>color</b> attribute is <b>Lime</b>.</li> <li>Save and close the file.</li> </ol>

*(continued)*


Tasks	Supporting information
<p>7. Browse to the modified deployed Web site.</p>	<ol style="list-style-type: none"> <li>a. Return to Internet Explorer.</li> <li>b. Refresh the browser window.</li> <li>c. Notice that the time displayed near the top left of the page now includes only hours and minutes.</li> <li>d. Notice also that the text for the following headings are displayed in lime green: <ul style="list-style-type: none"> <li>• Providing Quality Bikes and Accessories</li> <li>• Building Biking Communities</li> </ul> </li> <li>e. Close Internet Explorer.</li> </ol>
<p>8. Synchronize the source files with the edited deployed Web site.</p>	<ol style="list-style-type: none"> <li>a. Return to Visual Studio.</li> <li>b. Refresh the view in the <b>Remote Web Site</b> list.</li> <li>c. Notice that the <b>Status</b> of the <b>Default.aspx</b> file and the code-behind file for <b>TopLevel.master</b> is displayed as <b>Changed</b>. This is because you edited these files.</li> <li>d. Select the changed files, and then synchronize them with the source Web site.</li> <li>e. After the synchronization is complete, review the <b>Default.aspx</b> file and the code-behind file for <b>TopLevel.master</b> in Visual Studio to verify that they contain the modifications you made to the deployed version of the Web site.</li> <li>f. Close all applications.</li> </ol> <p> See the resource in the Resource Toolkit, “How to: Deploy a Web Site by Using the Copy Web Site Utility.”</p>

## Exercise 2



### Precompiling and Deploying a Web Application by Using the Publish Web Site Utility

In this exercise, you will use the Publish Web Site utility to attempt to deploy a Web application. You will see that compilation errors prevent the Web site from being published, and you will then resolve the compilation errors and publish the Web site again. You will identify cosmetic changes that must be made to the published Web site, and you will make those changes in the source files. You will then re-publish the site.

### Scenario


Tasks	Supporting information
1. Open the Adventure Works Web site.	<p>a. If you want to complete this lab by using Visual Basic:</p> <ul style="list-style-type: none"> <li>Open the <b>VB.sln</b> solution in the <b>E:\Labfiles\Starter\VB\PublishWeb</b> folder.</li> </ul> <p>b. If you want to complete this lab by using Visual C#:</p> <ul style="list-style-type: none"> <li>Open the <b>CS.sln</b> solution in the <b>E:\Labfiles\Starter\CS\PublishWeb</b> folder.</li> </ul>
2. Identify build errors when attempting to publish a Web site.	<p>a. Use the <b>Publish Web Site</b> command in the <b>Build</b> menu to publish the <b>PublishWeb</b> project.</p> <p>b. Use the ellipsis button (...) to select the target location for the published Web site, as follows:</p> <ul style="list-style-type: none"> <li>Connect to <b>Local IIS</b>.</li> <li>Create a new virtual directory in the <b>Default Web Site</b> with an <b>Alias name</b> of <b>PubWeb</b>, and a <b>Folder</b> of <b>E:\Labfiles\wwwroot\PubWeb</b>.</li> <li>Select the newly created <b>PubWeb</b> virtual directory as the Web site to open.</li> </ul> <p>c. Configure the precompiled site to be updateable.</p> <p>d. The build process starts, but compilation errors prevent the Web site from being built or published. Review the build errors.</p> <p>e. View the <b>Default.aspx</b> code-behind file and resolve the build errors by replacing all three occurrences of <b>TableCell</b> with <b>HtmlTableCell</b>.</p> <p> See the following resources in the Resource Toolkit:</p> <ul style="list-style-type: none"> <li>“Features of the Publish Web Site Utility”</li> <li>“How to: Precompile and Deploy a Web Site by Using the Publish Web Site Utility”</li> </ul>

(continued)

Tasks	Supporting information
<p>3. Publish the Adventure Works Web site.</p>	<p>a. Publish the Web site again, using the same target location as before.</p> <p>b. After the Web site is published successfully, start Internet Explorer, and then browse to <b>http://localhost/PubWeb</b>.</p> <p>c. Notice that the time displayed near the top left of the page includes hours, minutes and seconds.</p> <p>d. Notice also that the text for the following headings are displayed in dark blue:</p> <ul style="list-style-type: none"> <li>• Providing Quality Bikes and Accessories</li> <li>• Building Biking Communities</li> </ul> <p>e. Leave Internet Explorer running—you will use it later in this exercise.</p> <p> See the resource in the Resource Toolkit, “How to: Precompile and Deploy a Web Site by Using the Publish Web Site Utility.”</p>
<p>4. Review the published files.</p>	<p>a. Using Windows Explorer, navigate to the <b>E:\Labfiles\wwwroot\PubWeb</b> folder.</p> <p>b. View the <b>Default.aspx</b> file by using Notepad.</p> <p> <b>Note</b> You can change any element of the page by editing this file, including setting the text color of the <b>div</b> and <b>span</b> elements to <b>Lime</b>, as required by the scenario. This is because you specified that the precompiled site was updateable. However, do <i>not</i> make any changes, and then close the file. You will make changes by using Visual Studio.</p> <p>c. Notice that none of the .aspx files has associated code-behind files. This is because the code for each file has been precompiled into a .dll assembly.</p> <p>d. View the contents of the <b>E:\Labfiles\wwwroot\PubWeb\bin</b> folder to review the .dll assemblies.</p> <p>e. Leave Windows Explorer running—you will use it later in this exercise.</p>
<p>5. Modify the source files of the Adventure Works Web site.</p>	<p>a. Return to Visual Studio 2005.</p> <p>b. Open the <b>Default.aspx</b> page in Source view.</p> <p>c. Locate the <b>div</b> element that contains the text Providing Quality Bikes and Accessories, and then edit the <b>div</b> element so that its <b>color</b> attribute is <b>Lime</b>.</p> <p>d. Locate the <b>span</b> element that contains the text Building Biking Communities, and then edit the <b>span</b> element so that its <b>color</b> attribute is <b>Lime</b>.</p> <p>e. Open the code-behind file for TopLevel.master, and then edit the code in the <b>Page_Load</b> method so that the <b>Time</b> is displayed by using the <b>ToShortTimeString</b> method, rather than the existing <b>ToLongTimeString</b> method.</p> <p>f. Save all files.</p>



*(continued)*


Tasks	Supporting information
<p>6. Re-publish the Adventure Works Web site.</p>	<p>a. Publish the Web site again to the same location as before. This time, do <i>not</i> allow the precompiled site to be updateable. Delete existing files in the destination location when prompted.</p> <p>b. After the Web site is published successfully, return to Internet Explorer, and then refresh the page.</p> <p>c. Notice that the time format displayed near the top left of the page now includes only hours and minutes.</p> <p>d. Notice also that the text for the following headings are displayed in lime green:</p> <ul style="list-style-type: none"> <li>• Providing Quality Bikes and Accessories</li> <li>• Building Biking Communities</li> </ul> <p>e. Close Internet Explorer.</p> <p> See the resource in the Resource Toolkit, “How to: Precompile and Deploy a Web Site by Using the Publish Web Site Utility.”</p>
<p>7. Review the published stub files for ASPX pages.</p>	<p>a. Return to Windows Explorer.</p> <p>b. Open the <b>Default.aspx</b> file in the <b>E:\Labfiles\wwwroot\PubWeb</b> folder by using Notepad.</p> <p>c. Notice that this file is simply a marker file and that you can no longer change any element of the page. This is because you specified that the precompiled site was <i>not</i> updateable. Close the file.</p> <p>d. Close all applications.</p>

## Exercise 3



# Building a Windows Installer Package for Deploying a Web Application

In this exercise, you will create a Windows Installer package for deploying a Web application. You will specify settings for the user interface for the Windows Installer package, and you will add a license agreement file. Finally, you will run the Windows Installer package and deploy the Web site.

## Scenario

Tasks	Supporting information
1. Open the Adventure Works Web site.	<p>a. If you want to complete this lab by using Visual Basic:</p> <ul style="list-style-type: none"> <li>Open the <b>VB.sln</b> solution in the <b>E:\Labfiles\Starter\VB\MSIWeb</b> folder.</li> </ul> <p>b. If you want to complete this lab by using Visual C#:</p> <ul style="list-style-type: none"> <li>Open the <b>CS.sln</b> solution in the <b>E:\Labfiles\Starter\CS\MSIWeb</b> folder.</li> </ul>
2. Add and configure a Web Setup project to the solution.	<p>a. Add a new project to the solution by using the following information:</p> <ul style="list-style-type: none"> <li>Project Type: <b>Setup and Deployment</b></li> <li>Template: <b>Web Setup Project</b></li> <li>Name: <b>MSISetup</b></li> </ul> <p>b. Add the <b>Project Output</b> of the <b>Active</b> configuration for the <b>MSIWeb</b> project to the <b>MSISetup</b> project.</p> <p>c. Set the following properties for the <b>MSISetup</b> project:</p> <ul style="list-style-type: none"> <li>Author: <b>Adventure Works</b></li> <li>Description: <b>Setup Application for Adventure Works</b></li> <li>Manufacturer: <b>Adventure Works</b></li> <li>Product Name: <b>Adventure Works</b></li> <li>RemovePreviousVersions: <b>True</b></li> <li>Title: <b>Adventure Works MSI Package</b></li> </ul> <p> See the following resources in the Resource Toolkit:</p> <ul style="list-style-type: none"> <li>“Features of Windows Installer Setup Projects”</li> <li>“How to: Create a Windows Installer Package for Deploying a Web Application”</li> </ul>
3. Design the user interface for the Windows Installer package.	<p>a. In Solution Explorer, right-click <b>MSISetup</b>, point to <b>View</b>, and then click <b>User Interface</b>.</p> <p>b. In the <b>Install</b> section of the tree view, right-click the <b>Start</b> node, and then click <b>Add Dialog</b>.</p> <p>c. In the <b>Add Dialog</b> dialog box, click <b>License Agreement</b>, and then click <b>OK</b>.</p>

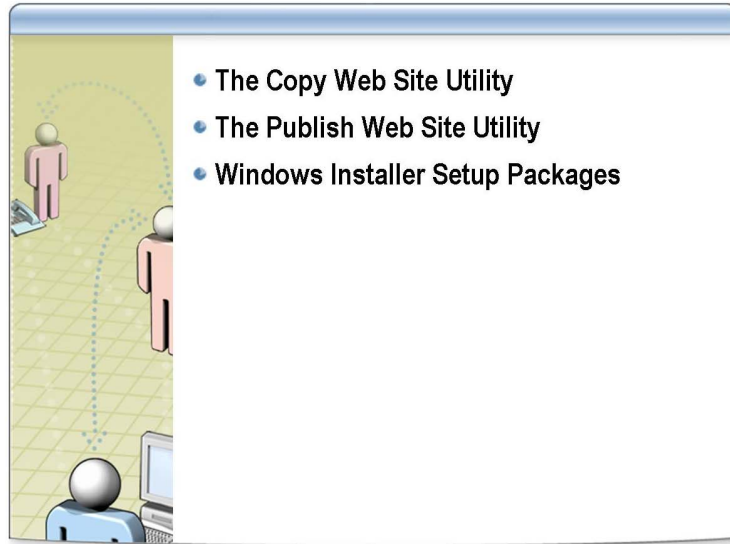
*(continued)*

Tasks	Supporting information
3. <i>(continued)</i>	<p>d. Drag the icon for the <b>License Agreement</b> so that it appears directly below the icon for the <b>Welcome</b> dialog box.</p> <p>e. Set the <b>LicenseFile</b> property to the <b>AdvWorksLicense.rtf</b> file.</p> <p> <b>Tip</b> Click <b>Browse</b> in the drop-down list for the <b>LicenseFile</b> property, and then browse to the <b>Web Application Folder</b> folder. Next, click <b>Add File</b>, and then use the <b>Add Files</b> dialog box to select the <b>AdvWorksLicense.rtf</b> file.</p> <p>f. Repeat steps b through e to add a <b>License Agreement</b> dialog box to the <b>Start</b> section of the <b>Administrative Install</b> sequence.</p> <p> See the resource in the Resource Toolkit, “How to: Create a Windows Installer Package for Deploying a Web Application.”</p>
4. Build the Windows Installer package.	<p>a. Build the <b>MSISetup</b> project.</p> <p>b. If you are working with Visual Basic, when the build process is complete, use Windows Explorer to browse to the <b>E:\Labfiles\Starter\VB\MSISetup\Debug</b> folder.</p> <p>c. If you are working with Visual C#, when the build process is complete, use Windows Explorer to browse to the <b>E:\Labfiles\Starter\CS\MSISetup\Debug</b> folder.</p>
5. Run the Windows Installer package.	<p>a. Run the <b>Setup.exe</b> file.</p> <p>b. Step through the installation wizard, following these instructions:</p> <ul style="list-style-type: none"> <li>Accept the license agreement.</li> <li>Select a Virtual Directory of <b>MSIWeb</b>.</li> </ul> <p>c. Start Internet Explorer, and then browse to <b>http://localhost/MSIWeb</b>.</p> <p>d. After the Web site appears, close Internet Explorer.</p>

## Lab Shutdown

After you complete the lab, you must shut down the **2543B-LON-DEV-01-08** virtual machine and discard any changes.

## Lab Discussion



In the lab, you:

- Deployed a Web site by using the Copy Web Site utility.
- Deployed a Web site by using the Publish Web Site utility.
- Deployed a Web site by creating and running a Windows Installer package.

The instructor will lead a group discussion of these tasks. You should be prepared to contribute to the discussion, especially if you encountered problems completing the exercises.

