

## A P P E N D I X D

## D



# DEPLOYING A VISUAL BASIC .NET APPLICATION

**After completing this appendix, you will be able to:**



- Understand how Visual Studio performs deployment
- Create a Setup project with a Wizard
- Configure a Setup project
- Test a Setup project

## D E P L O Y I N G   A   V I S U A L   B A S I C   . N E T   A P P L I C A T I O N

## INTRODUCTION TO DEPLOYMENT

Throughout this book, you have developed desktop Visual Basic applications, and throughout the development process, you have tested and debugged those applications from inside of the Visual Studio IDE. However, the end users of the applications you create will not likely run these applications from inside of Visual Studio. In fact, the end users of your applications will likely not even have Visual Studio installed on their computers. Rather, the end users will install your application just as they would install any other application. That is, the end user will run an Install.exe or Setup.exe program. The Install or Setup program will guide the end user through the installation process. The Setup program copies your application along with any necessary files to the end user's computer.

After the installation process is complete, the end user can execute your application from the Start menu or by double-clicking a desktop icon, just as they would execute any other application. The Setup program takes care of creating any necessary items on the Start menu. In addition, the Setup program can create shortcuts on the end user's desktop.

In this appendix, you will learn how to configure desktop applications so that they can be distributed to other computers. This distribution process is called *deployment*. During the deployment process, you, as the developer, build another type of project, called an installer. An *installer* consists of two files, typically named Setup.exe and Setup.msi. This installer travels with the desktop applications during the deployment process.

The deployment process is essential because unless the end user has Visual Studio or the .NET Framework installed on their computer, copying an executable file to another computer and trying to run it will not work. Note that it is also possible to deploy the .NET Framework as part of a Setup project.

## NOTES ABOUT THE SOLUTION DEPLOYED IN THIS APPENDIX

This appendix does not contain an application lesson in the same way as other chapters in this book. The subject of this appendix is deploying an application that has already been completed. Thus, you will use an application that was created in a previous chapter and complete steps to deploy that application to another computer. Regardless of the application that you are deploying, the process of creating a Setup project is the same.

## A P P E N D I X D

The process of deployment involves two categories of computers, as follows:

- » The computer used to create the Setup project (your computer) is called the *development computer*. As mentioned, the purpose of a Setup project is to create another program called an installer. The term *Setup project* and *Deployment project* are often used synonymously. This appendix uses the term Setup project to refer to a specific type of project supported by Visual Studio. The Setup project bundles all of the elements of an application so that it can be distributed to another computer. To distribute the application, you can copy the output of the Setup project to removable media such as floppy disks, a CD-ROM, or DVD. Setup files can also be placed on a Web site where they can be downloaded by anyone connected to the Internet or a company intranet.
- » The end user's computer is called the *target computer*. The end user typically runs the installer (the file named Setup.exe) that you created to install your application on their computer. By running the installer, all of the files necessary to run your application are copied to the target computer to a folder you specified when you created the installer. In addition, entries are added to the Start menu and optionally to the desktop so that the end user can run the application. The installer will also copy related support files such as Help files, data files, or other informational files. The target computer need not have Visual Studio installed. This is one of the main goals of building a Setup project.

In this appendix, you will deploy a project with the following characteristics:

- » The project that you deploy is a Windows Application project.
- » The project that you deploy has data files that will be deployed along with the executable file itself.

## INTRODUCTION TO DEPLOYMENT WITH VISUAL STUDIO

To deploy an application so that the end user can install it on the target computer, you begin by opening an existing Visual Basic solution and project. No matter the type of project being deployed, you perform a similar set of steps, as described in the following list:

1. Open an existing solution containing one or more projects.
2. Create a Setup project within the solution.
3. Configure the Setup project by specifying the type of project being deployed along with any support files that need to be deployed with the project. In addition, it is possible to define where the executable files will be placed on the target computer, and any items to be created on the target computer's Start menu.

## D E P L O Y I N G   A   V I S U A L   B A S I C   . N E T   A P P L I C A T I O N

4. Build the Setup project to create the installer (Setup.exe and related files). Note that you build a Setup project in the same way that you build a Windows Application project.
5. Test the Setup project on the local computer.
6. Distribute the Setup.exe program and any necessary files to the end user via removable media, the Internet, or a company intranet.

## CREATING A SETUP PROJECT WITH A WIZARD

### »NOTE

This appendix introduces only the most basic elements of the deployment process. To learn more about what you can do within the deployment process, examine the Visual Studio Help system.

The Setup Wizard automates the process of building a Setup project. In this appendix, you will use it to build a Setup project for a simple application. Note, however, that it is possible to create much more complex Setup projects than the one discussed in this appendix. For example, it is possible to conditionally deploy files as selected by the end user. In addition, it is possible to test the hardware and software configuration of the target computer to make sure that it is capable of running your application. It's also possible to create a Setup project that will require the end user to register your application.

The Setup Wizard allows you to create a Setup project by filling out a series of dialog boxes. After creating the Setup project, you then build it to create the installer that the end user will execute. The following list introduces the five dialog boxes that you complete when using the Setup Wizard:

1. The first dialog box welcomes you to the Setup Wizard and describes the tasks you will perform as you build the Setup project.
2. In the second dialog box, you define the type of Setup project that you want to create. The type of Setup project that you create depends on the underlying type of project being deployed. Different types of Setup projects are created for Windows Applications and Web Applications, for example.
3. In the third dialog box, you tell the Setup Wizard what program outputs should be included in the Setup project. When a desktop application (Windows Application project) is compiled, Visual Studio produces a file with a suffix of .exe, indicating that the file is an executable file. Visual Studio uses the term *program output* to refer to this executable file. Other program outputs include such files as dynamic link libraries produced by building a Class Library project.

## A P P E N D I X D

4. In the fourth dialog box, you specify any other files that need to be included in the Setup project. These files might be required database files, Help files, or sample files. In summary, the additional files can consist of any files that you want to deploy to the target computer.
5. The final dialog box displays a summary report describing the actions taken by the Setup Wizard.
1. Start Visual Studio and open the solution named **Appendix.D\Chapter10ApplicationLessonPreview\Chapter10ApplicationLessonPreview.sln**. This is the same program as the one you created in the application lesson for Chapter 10.
2. Next, you will create a Setup project and add it to the solution using the Setup Wizard. On the menu bar, click **File, Add, New Project** to activate the New Project dialog box.
3. Expand the **Other Project Types** folder. Click the **Setup and Deployment** option. The Add New Project dialog box should appear similar to the one shown in Figure D-1.

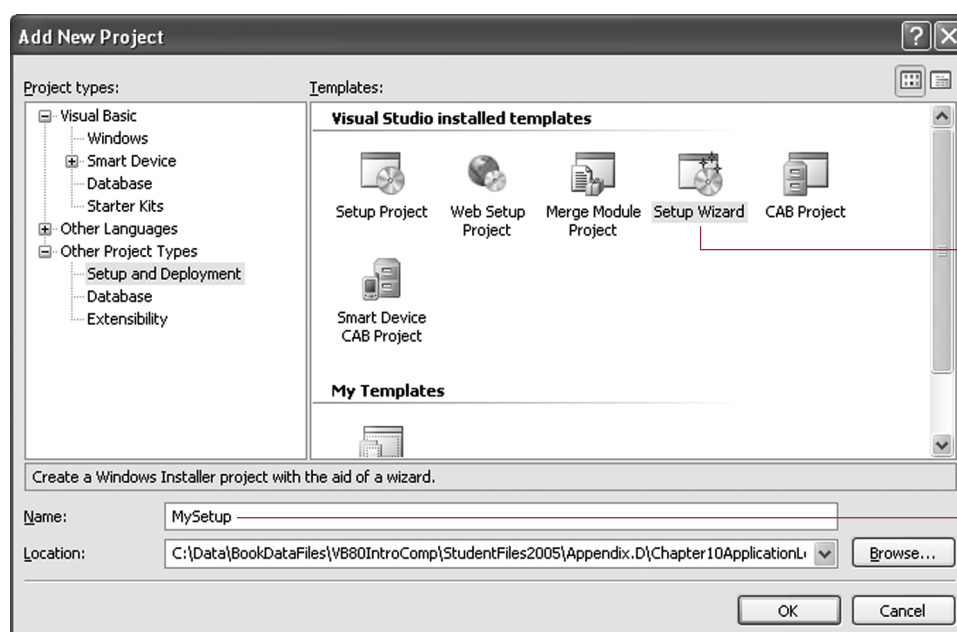


### »NOTE

The exploration exercises in this appendix differ from most of the exploration exercises found in this book. Instead of examining how preexisting code works or looking at a particular concept, the exploration exercises in this appendix show you how to create a Setup project.

### »NOTE

The Visual Basic Express edition does not contain the templates necessary to create a Setup project. Thus, if you are using the Express edition, you will not be able to complete the exploration exercises in this appendix.



Setup Wizard  
selected

Setup project  
name

Figure D-1: Add New Project dialog box

4. In the dialog box, click **Setup Wizard** in the Templates section. Set the name of the new project to **MySetup**. Click **OK** to activate the Setup Wizard.
5. Click **OK** to close the dialog box and activate the Setup Wizard. The Welcome dialog box opens, as shown in Figure D-2.

# DEPLOYING A VISUAL BASIC .NET APPLICATION

## NOTE

Note that there are many other types of Setup projects than the one introduced in this appendix. For instance, there are Setup projects for the Web and Setup projects used to deploy Class Library projects.



Figure D-2: Setup Wizard—Welcome dialog box

6. Click **Next** to display the Choose a project type dialog box shown in Figure D-3.

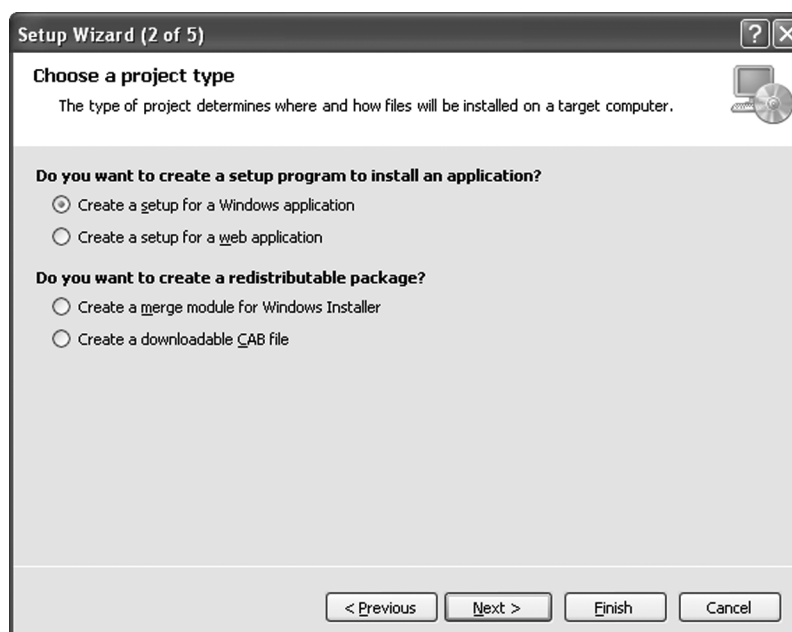


Figure D-3: Setup Wizard—Choose a project type dialog box

## A P P E N D I X D

7. The Choose a project type dialog box contains four radio buttons allowing you to specify the type of project being deployed. The option Create a setup for a Windows application is used to create a Setup project for a desktop Windows Application project. Make sure this radio button is selected. The other options allow you to create a Setup project for a Web application or to create a Setup project for a redistributable class library.
8. Click **Next** to display the Choose project outputs to include dialog box shown in Figure D-4.

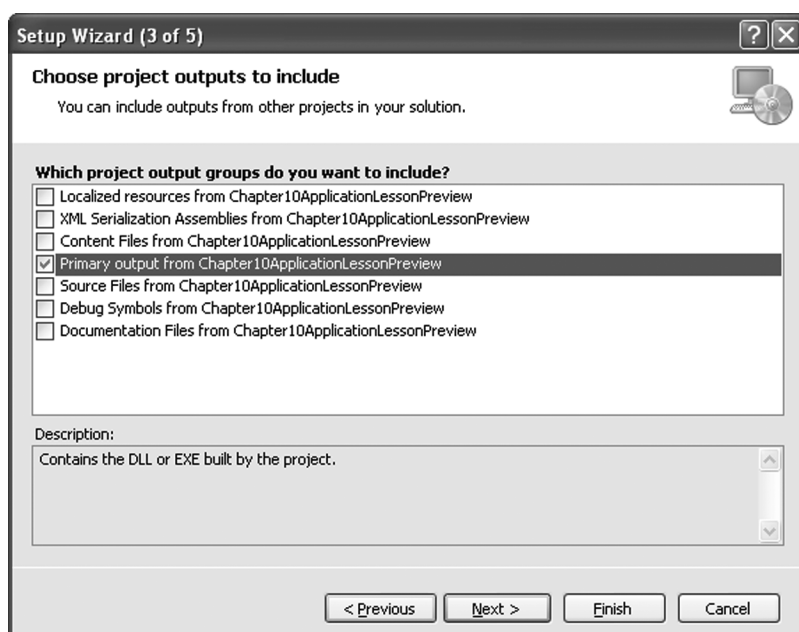


Figure D-4: Setup Wizard—Choose project outputs to include dialog box

9. The Choose project outputs to include dialog box allows you to select the parts of a project that will be deployed. As you can see, it is possible to deploy several different parts of a project, including the source code, documentation, and other project elements. Check the **Primary Output from Chapter10ApplicationLessonPreview** check box to cause the project's executable file to be deployed. Then, click **Next** to activate the Choose files to include dialog box shown in Figure D-5.

# DEPLOYING A VISUAL BASIC .NET APPLICATION

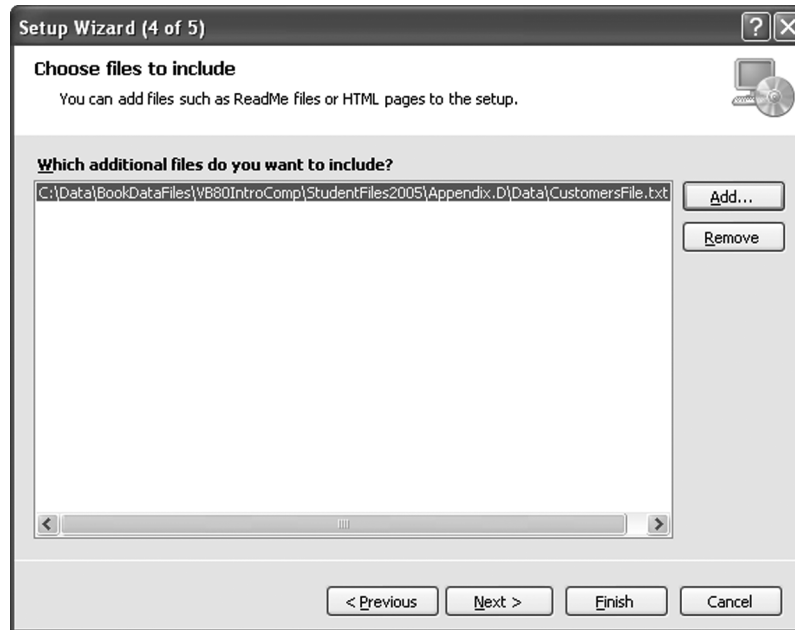


Figure D-5: Setup Wizard—Choose files to include dialog box

10. In addition to deploying the executable file for a project, it is also possible to deploy other files, such as database files, text files, Help files, or any other file that is needed by the end user on the target computer. Click the **Add** button. The Add Files dialog box opens, allowing you to select files to add to the Setup project. In the Add Files dialog box, select the file named **Appendix.D\Data\CustomersFile.txt**. Click the **Add** button to add the file to the Setup project. This is the sequential file associated with the application. After the file is selected, it appears on the Choose files to include dialog box.
11. Click the **Next** button to display the Create Project dialog box shown in Figure D-6.



## A P P E N D I X D

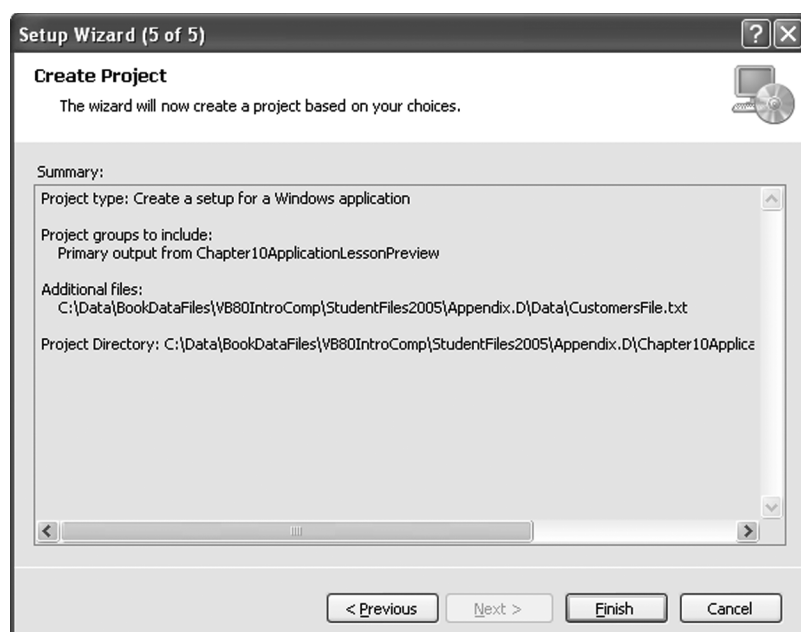


Figure D-6: Setup Wizard—Create Project dialog box

12. Click **Finish** to generate the Setup project. The Setup Wizard adds the Setup project to the solution. The new project appears in the Solution Explorer, as shown in Figure D-7.

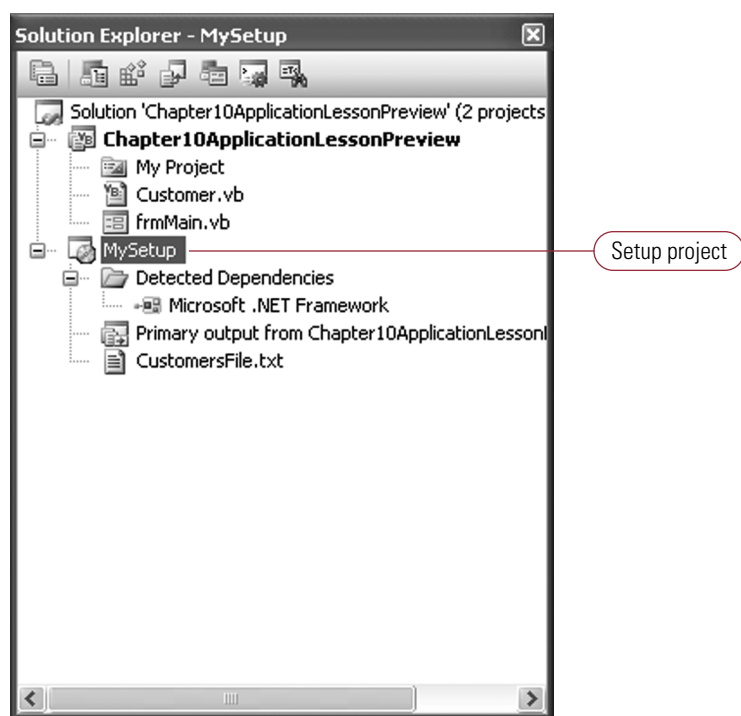


Figure D-7: The Solution Explorer displaying a Setup project

## DEPLOYING A VISUAL BASIC .NET APPLICATION

As shown in Figure D-7, the Solution Explorer contains an additional project named MySetup. The files contained by a Setup project are much different than those contained by an ordinary Windows Application project, as described in the following list:

- » The folder titled Detected Dependencies indicates that the .NET Framework must be installed for the application to run. That is, the application is dependent on the .NET Framework.
- » Primary Output From Chapter10ApplicationLessonPreview (Active) represents the executable file from the application being deployed.
- » The file named CustomersFile.txt is the data file you elected to deploy with the project.

### INTRODUCTION TO THE CONFIGURATION MANAGER

When you build a solution and its projects for deployment to other computers, another dialog box called the Configuration Manager is used to configure how the solution will be deployed and the elements that will be deployed with the solution. Figure D-8 shows the Configuration Manager dialog box.

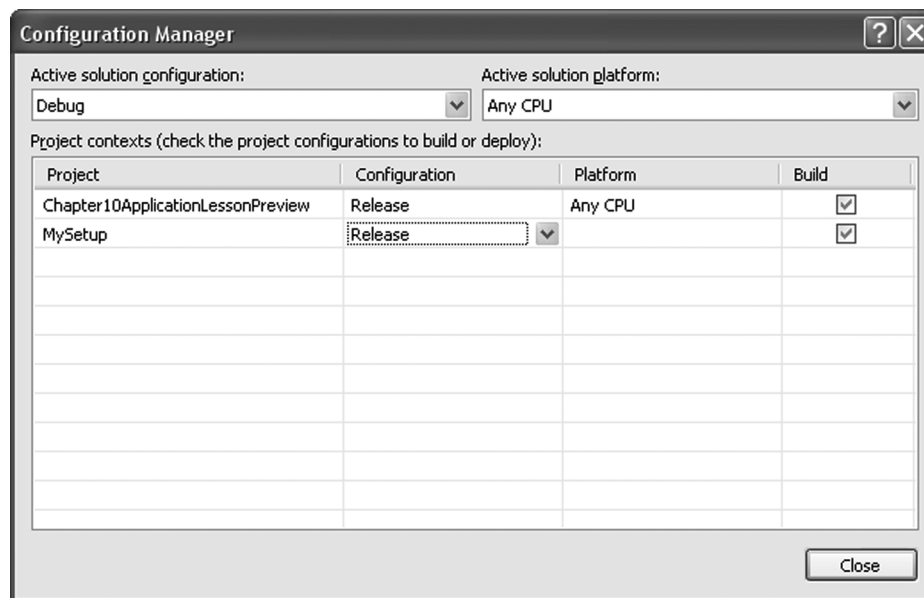


Figure D-8: Configuration Manager dialog box

As shown in Figure D-8, the Configuration Manager dialog box lists the projects in the solution. The solution shown in Figure D-8 has two projects. The first project is the Windows Application project being deployed. The second project is the Setup project.

## A P P E N D I X D

As shown in Figure D-8, the Configuration Manager dialog box has four columns. The purpose of each column is described in the following list:

- » The project name appears in the first column.
- » The Configuration appears in the second column. By default, there are two configuration options. The first, named Debug, should be used while you are debugging the application. The second, named Release, should be used when the application is being deployed to the target computer.
- » The third column allows you to select the type of CPU for which the application will be compiled.
- » The final column indicates whether the project will be built. If the check box is not checked, the project will not be compiled. If the project is not compiled, the executable file will not be included in the Setup project.

### THE PROJECT DEPENDENCIES DIALOG BOX

Another dialog box is important in the deployment process—the Project Dependencies dialog box. The Project Dependencies dialog box is made up of two tabs, as discussed in the following sections.

#### THE DEPENDENCIES TAB

The first tab is the Dependencies tab. Figure D-9 shows the Project Dependencies dialog box with the Dependencies tab active.

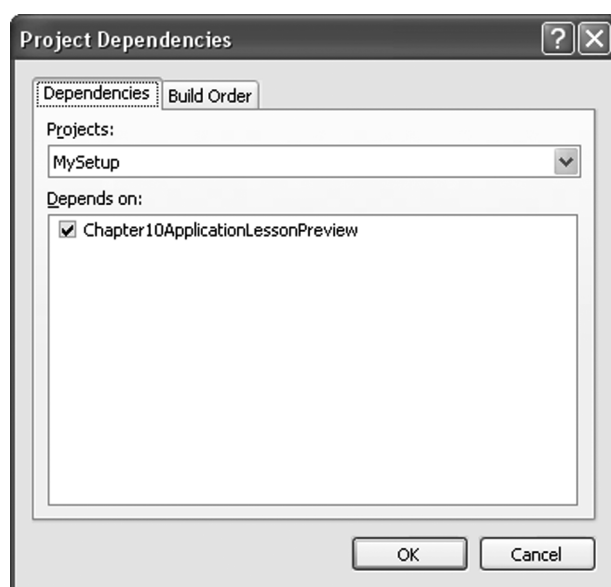


Figure D-9: Project Dependencies dialog box—Dependencies tab

## DEPLOYING A VISUAL BASIC .NET APPLICATION

From the Projects combo box, you select one of the projects in the solution. The Depends on list box lists the other projects in the solution. Checking the check box to the left of the project indicates that a project is dependent on another project. As shown in Figure D-9, the Setup project is dependent on the Windows Application project named Chapter10ApplicationLessonPreview.

### THE BUILD ORDER TAB

The second tab is the Build Order tab. The Build Order tab shown in Figure D-10 lists the order in which the projects in the solution will be compiled. Generally, the Setup project should be compiled last.

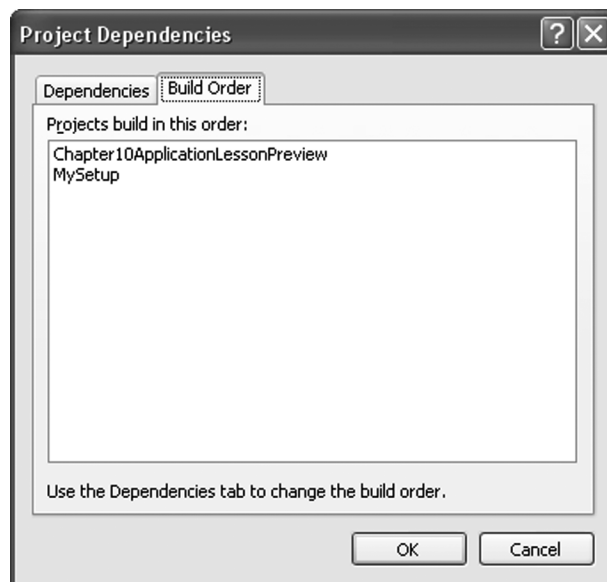


Figure D-10: Project Dependencies dialog box—Build Order tab

In this exploration exercise, you will configure the project dependencies and configure the solutions for deployment.

1. On the menu bar, click **Build**, and then click **Configuration Manager**. The Configuration Manager dialog box opens, as shown in Figure D-8.
2. In the Configuration column, click the list arrow to select the **Release** configuration, if necessary.
3. In the Build column, make sure that both check boxes are checked.
4. In the Active solution configuration dialog box, select **Release**, if necessary.
5. Click **Close** to close the dialog box.
6. On the menu bar, click **Project**, and then click **Project Dependencies**. Note that the project named MySetup depends on the project named Chapter10Application LessonPreview.

## A P P E N D I X D

7. Click the **Build Order** tab. Note that the project named Chapter10Application LessonPreview is built before the project named MySetup.
8. Click **OK** to close the dialog box.

At this point, you have created the Setup project using the Setup Wizard and configured the solution for deployment. However, the Setup project, as it presently exists, will not configure the Start menu on the target computer. You will configure the Start menu in the next section of this appendix.

## CONFIGURING THE START MENU ON THE TARGET COMPUTER

To configure the Setup project so that the application will appear on the Start menu of the target computer, you use the File System Editor supported by the Setup project. The File System Editor is one of six editors supported by Setup projects. Figure D-11 shows the File System Editor.

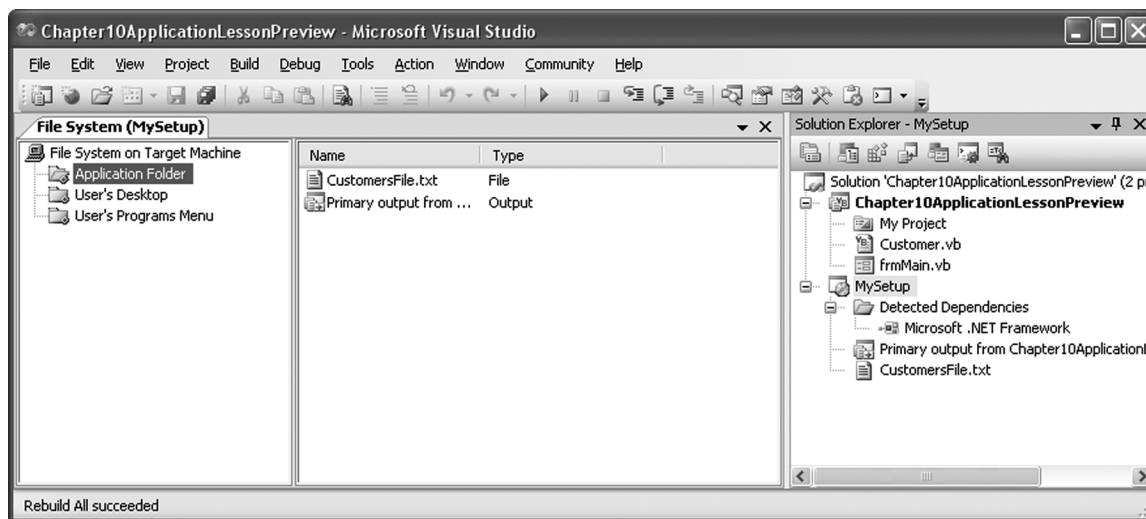


Figure D-11: The File System Editor

As shown in Figure D-11, the File System Editor is made up of two panes. The left pane displays three folders, as described in the following list:

- » The folder named Application Folder contains the files that will be deployed to the target computer.

## DEPLOYING A VISUAL BASIC .NET APPLICATION

- » The folder named User's Desktop allows you to define the files that will appear on the user's desktop on the target computer.
- » The folder named User's Programs Menu allows you to define the files that will appear on the Start menu of the target computer.

To configure the installer so that a program or other item will appear on the Start menu, you add the desired program to the User's Programs Menu folder.

In this exploration exercise, you will configure the installer so that the application will appear on the Start menu of the target computer.

1. The File System Editor should be visible. If it is not, right-click the Setup project named **MySetup** in the Solution Explorer. On the context menu, click **View**, and then click **File System**.
2. In the File System Editor, right-click the **User's Programs Menu** folder. Click **Add**, **Project Output** on the context menu. The Add Project Output Group dialog box opens, as shown in Figure D-12.

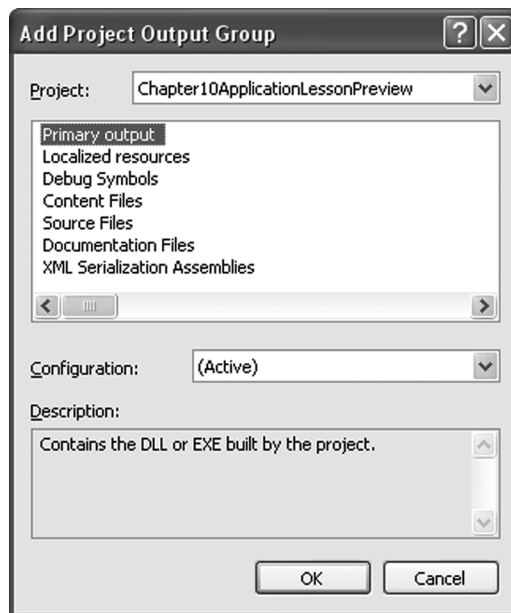


Figure D-12: Add Project Output Group dialog box

3. Select the item named **Primary output**, if necessary. This option causes the executable program to be added to the Start menu on the target computer.
4. Click **OK** to close the dialog box. The item appears in the File System Editor.

## A P P E N D I X D

In this example, you have added a single item to the Start menu on the target computer. However, it is possible to add several items and to create submenus on the target computer.



# BUILDING AND TESTING THE INSTALLER

The final step in the deployment process is to actually build the Setup project and test it. The process to build a Setup project is the same as the process to build any other project. That is, you click Build, and then click Rebuild Solution. After the Setup project has been built, it can be tested.

In this exploration exercise, you will build and test the Setup project you just created.

1. On the menu bar, click **Build, Rebuild Solution** to build both the Windows Application project and the Setup project.
2. In the Solution Explorer, right-click the Setup project named **MySetup**. On the context menu, click **Install**. This starts the installer as the end user will see it. Figure D-13 shows the Welcome dialog box.



Figure D-13: Welcome dialog box

3. Click **Next** to display the Select Installation Folder dialog box shown in Figure D-14.

## D E P L O Y I N G   A   V I S U A L   B A S I C   . N E T   A P P L I C A T I O N

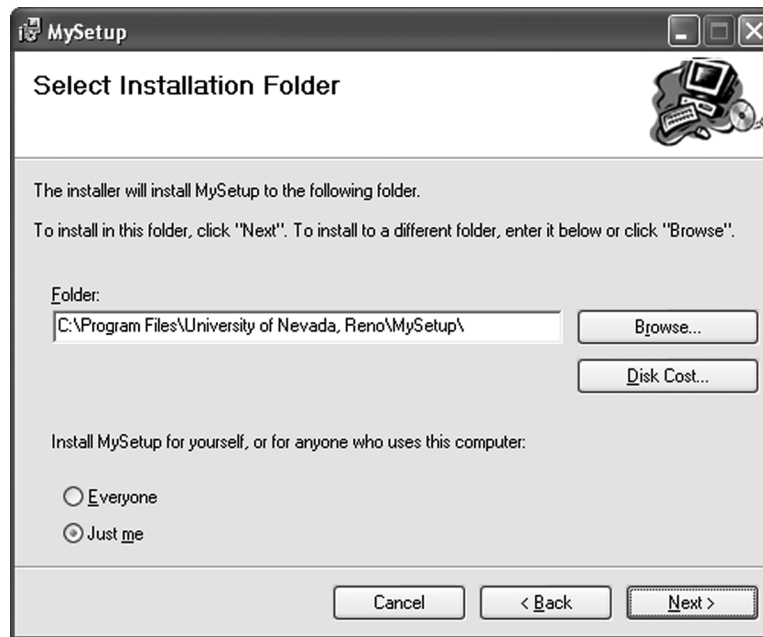


Figure D-14: Select Installation Folder dialog box

4. The Select Installation Folder dialog box allows the end user to change the location where the application will be installed. Click **Next** to display the Confirm Installation dialog box shown in Figure D-15.

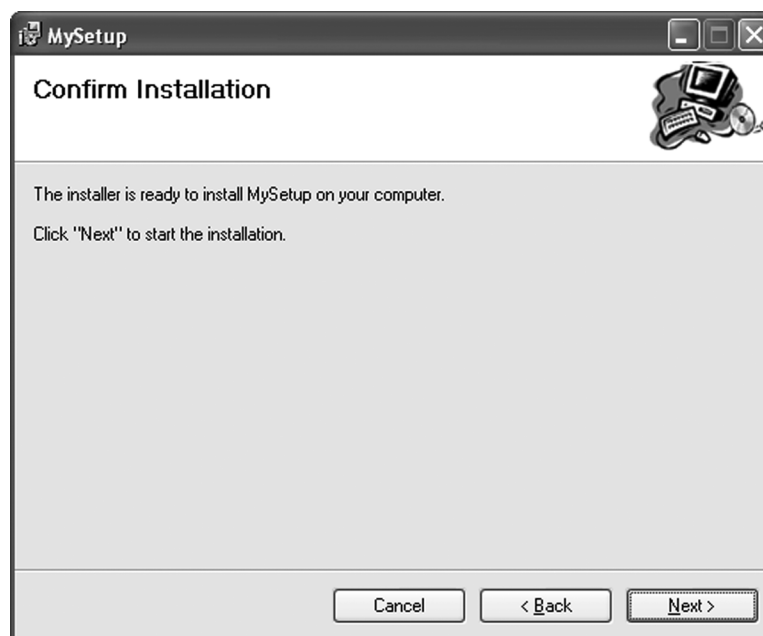


Figure D-15: Confirm Installation dialog box



## A P P E N D I X D

5. Click **Next** to run the installer. A dialog box opens displaying the progress of the installation. When the installation is complete, the dialog box shown in Figure D-16 opens.

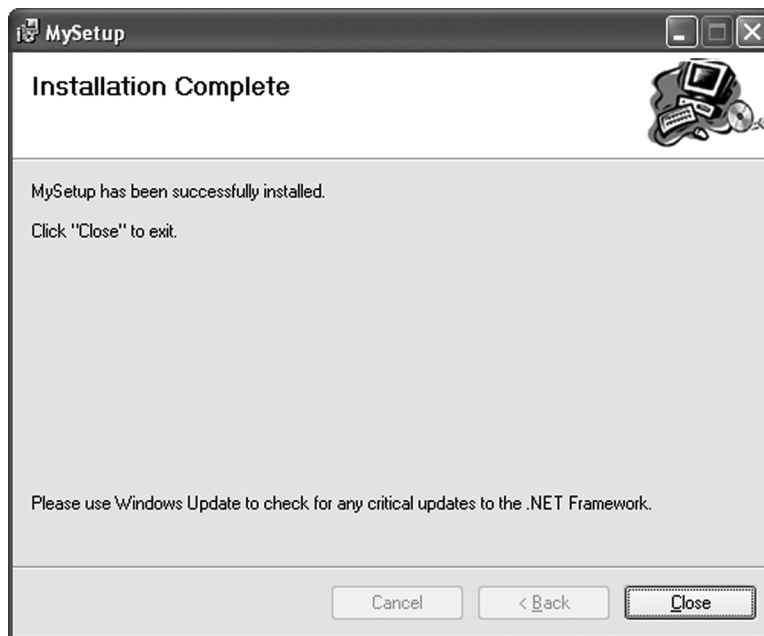


Figure D-16: Installation Complete dialog box

6. Click **Close** to close the dialog box.
7. Click the **Start** menu, and select **All Programs**. The application that you installed appears on the Start menu.

That's it. You have just created your first installer. As mentioned, it's possible to create much more complex installers than the one presented in this appendix.

## APPENDIX SUMMARY

- » The process of distributing applications to other computers is called deployment. To deploy an application, you create another type of project called an installer. An installer consists of two files named Setup.exe and Setup.msi.
- » The deployment process involves two computers. Your computer is called the development computer. The end user's computer is called the target computer.

## D E P L O Y I N G   A   V I S U A L   B A S I C   . N E T   A P P L I C A T I O N

- » The easiest way to create an installer is to complete the dialog boxes that make up the Setup Wizard. The Setup Wizard allows you to define the type of Setup project you will create and the outputs that will be included in the Setup project along with any additional files.
- » By default, the installer created by the Setup Wizard will not add items to the Start menu on the target computer. This task is accomplished using the File System Editor.
- » After an installer has been created, it can be tested from inside of Visual Studio by selecting the Setup project in the Solution Explorer, and clicking Install.