Use the Package and Deployment Wizard to distribute your Visual Basic 6 programs

One of my most popular books is Learn to program with Visual Basic Examples.

In the book, I answer 100 of the most common Visual Basic 6 questions I've ever been asked. Undoubtedly, this question, how to package and deploy a VB program you write, is the most popular.

I've decided to present here the full text of Question #69---how to use the Package and Deployment Wizard to distribute your programs.

I hope you find it useful---and think about buying the book.

Question #69--- I gave my husband an executable, but it doesn't run on his PC

"Hi, my name is Margaret, and I'm from Independence, Missouri," caller number nine said. "My husband Harry is on an extended business trip to Washington, DC and so I've been taking a Visual Basic course at our local Community College here in Independence. I wrote a program last week that I was very proud of, and I wanted to send it to Harry. I generated an executable of the program, just like we do in the classroom at college, and sent it off to Harry. But when he tried to run it on his PC in his office, it wouldn't run… something about a missing DLL. Can you tell me what I may have done wrong?"

"This is a pretty common problem Margaret," I said. "The Visual Basic executable you sent to your husband Harry cannot run on its own---it needs some Visual Basic DLL's to go along with it."

"Oh...call me Bess," she said. "Everyone does. But why did the executable run in the classroom---I wrote the program at home---is that because those Visual Basic DLL's are installed there?"

"That's right, Bess" I said. "If a PC contains Visual Basic, then the DLL's your executable needs to run are already installed on that PC. The problem is that Harry needs them to see your executable in action---and they're not on his PC."

"How can I get him those DLL's," she replied. "If you give me a list of them, I can copy them onto a diskette, and mail them to him."

"I'm afraid it's not that simple Bess," I said, "I could give you a list of the DLL's that you need to give him. But there are a few other files he needs also. It would be easy for you to miss one. Plus, Harry needs to install these in specific directories on his PC, and in some cases, he needs to register them in his Windows Registry."

"That sounds pretty difficult," Bess said. "Maybe I should just wait until he's back home."

"That's probably not necessary," I said. "I'm not sure of which version and edition of Visual Basic you are running at home or in the classroom, but most versions have an Application Setup Wizard which will lead you through the process of copying the necessary support files---DLL's among them---onto a diskette. This wizard will generate a professional looking setup program so that all Harry needs to do is click on the Setup.exe file on the diskette using Windows explorer, and your program will be installed on his PC, ready to run."

"That sounds great," Bess said, "but it sounds too easy. Can you show me how to run the Wizard?"

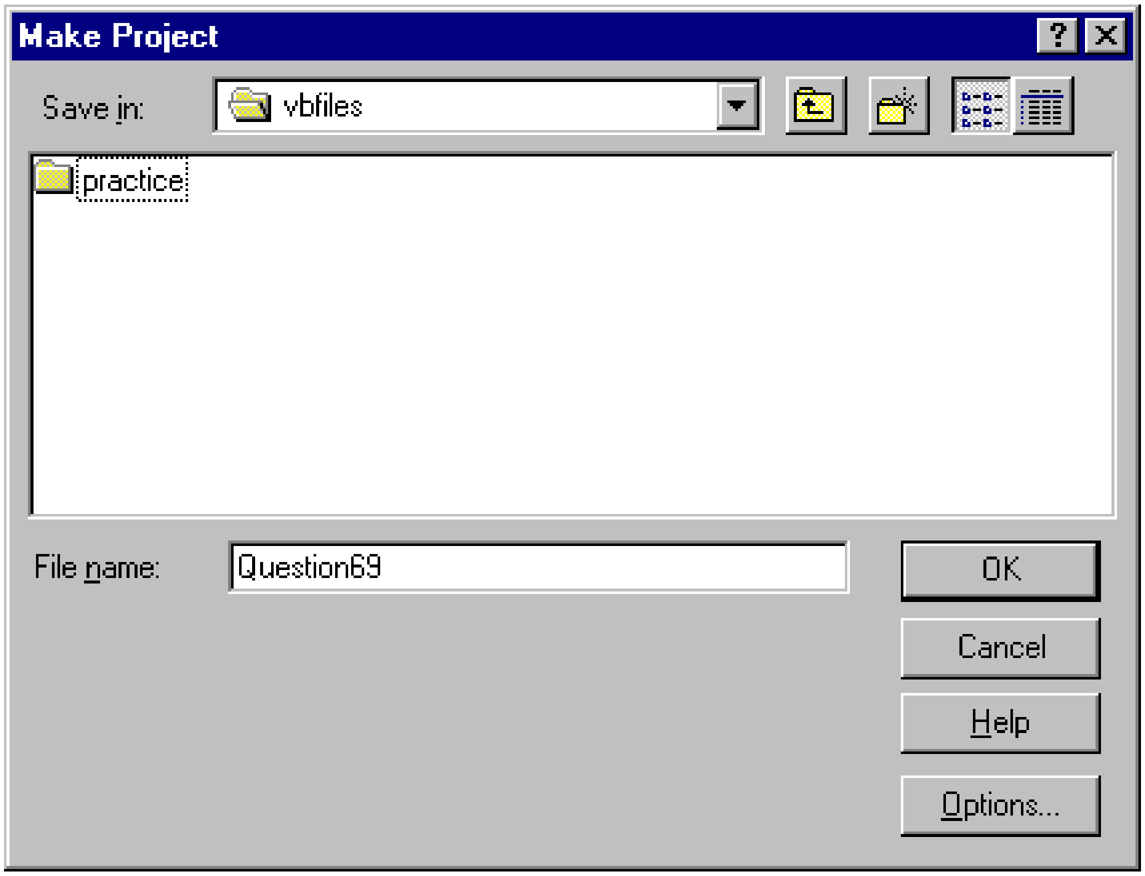
"I sure can," I said. "Let me first create a simple little program---a form with a command button---to use in our demonstration. Then I'll generate an executable, and finally use the Setup Wizard to create a set of setup diskettes."

I started a new Visual Basic project and placed a single command button on the form.

"The first thing we need to do," I said, "is to save this project, then compile the program so that we have something to 'Distribute'."

I saved the Project as **Question69** in my \VBFILES\Practice directory, and then selected Make-Project1.exe from the Visual Basic Menu Bar.

"Rather than accept the default name of Project1.exe," I said, "let's generate the executable as 'Question69', and save it in the \VBFILES\Practice Directory."

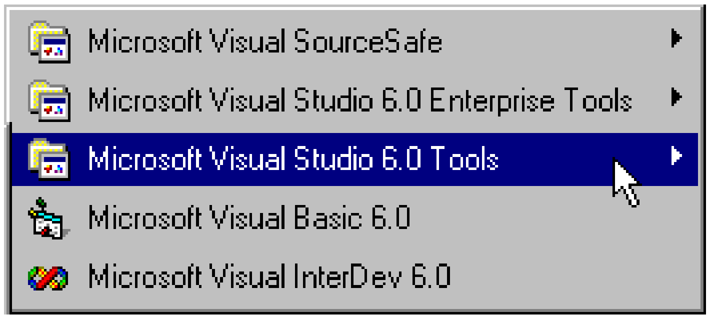


"At this point," I said, "we now have an executable called Question69.exe. Now we could easily run this program using the Start-Run button, since all of the Visual Basic support DLL's are already installed on this PC. But this executable will not run on any PC that does not have Visual Basic installed. I should mention that in rare instances, the Visual Basic DLL's might be present on a PC which has not had Visual Basic installed, but that is a rare occurrence."

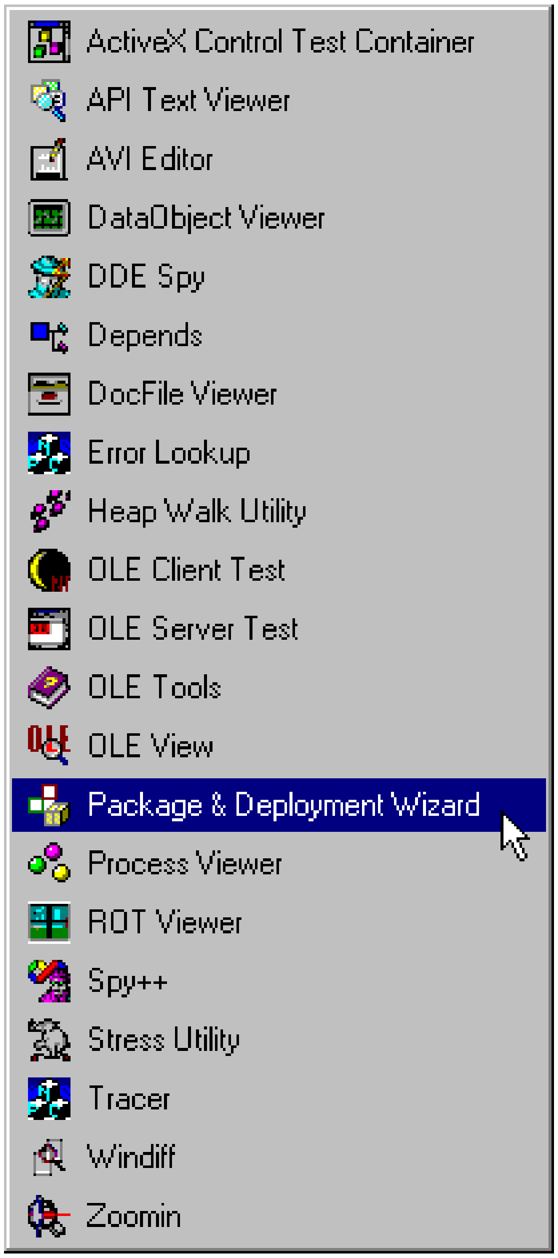
"Do we run the Wizard now," Bess asked.

"Yes Bess," I replied, "now it's time to run the Wizard. Perhaps the hardest part of that process will be finding it. In Visual Basic 6, the Wizard can be found either in the Visual Basic 6 group, on this PC in the Visual Studio 6.0 Tools folder, or can also be run from within Visual Basic itself, under the Add-In menu. In Visual Basic 5, the wizard will only be found in the Visual Basic program group itself."

I opened up the Visual Basic program group on my studio PC, and selected 'Microsoft Visual Studio 6.0 Tools from the menu…


Microsoft Visual SourceSafe
Microsoft Visual Studio 6.0 Enterprise Tools
Microsoft Visual Studio 6.0 Tools
Microsoft Visual Basic 6.0
Microsoft Visual InterDev 6.0

… then I selected the Package and Deployment Wizard….



"Didn't you say that the Wizard was called the Application Setup Wizard?" Bess asked.

"My apologies Bess," I said, "that's the name of the Wizard in Visual Basic 5 world. The name of the Wizard changed in Visual Basic 6, mainly because Microsoft added an additional step to the distribution process in which we need to first create something called a Package."

"So both do essentially the same thing?" Bess asked. "But the Visual Basic 6 version is doing something more. Does that complicate things?"

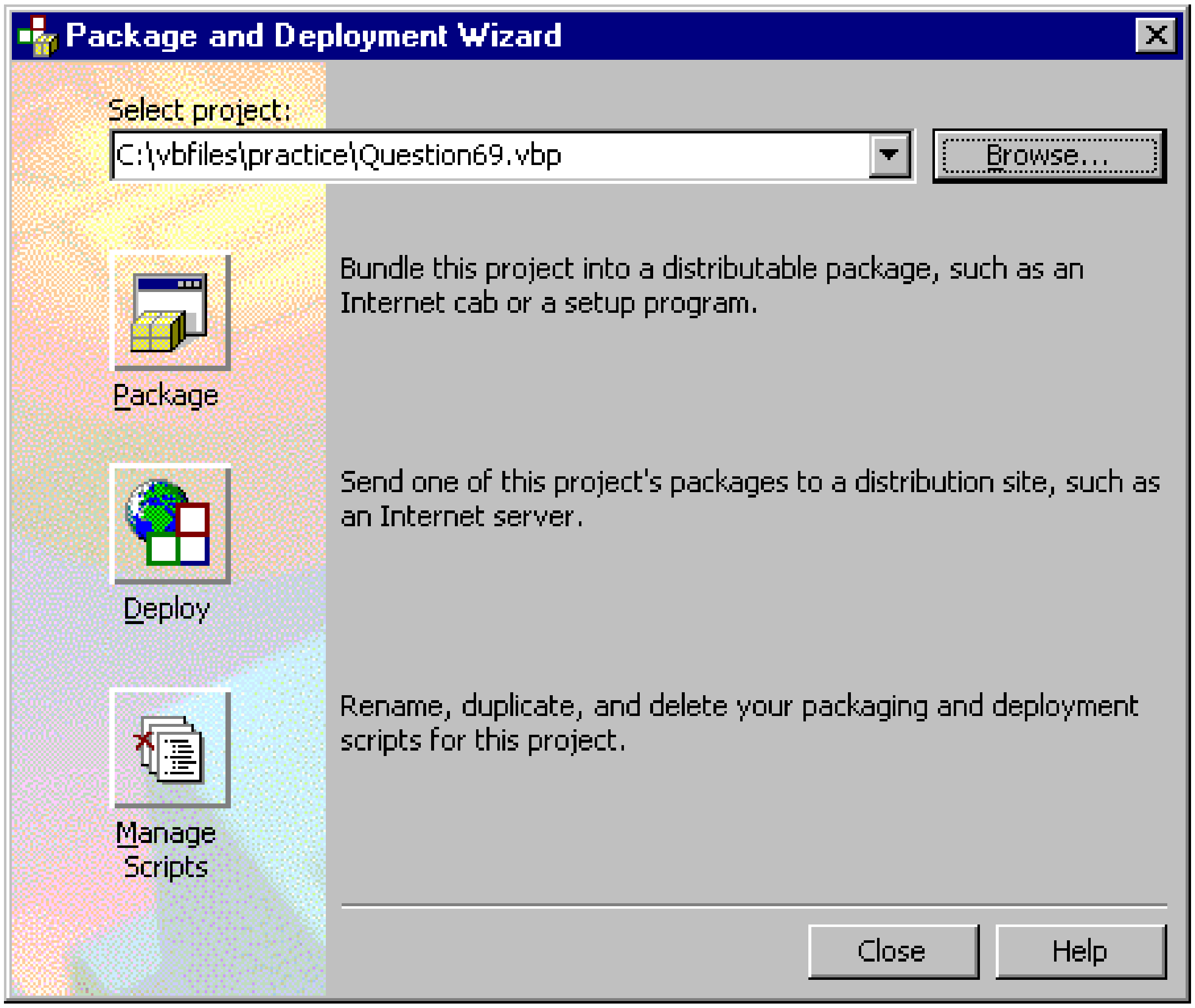
"A bit," I said. "People using Visual Basic 5 don't need to worry about first creating a package, that's all. They go right to the Deployment Part of the Wizard. It's a few more steps in Visual Basic 6."

I waited for a moment to see if Bess had any questions, and then selected the Wizard in the tools submenu.

"Our first step then," I continued, "is to select a project---in this instance, Question69.vbp in our Practice directory. We can do that now by using the Browse button in the Wizard to find and select it…"

I did that.

"Once our project is selected," I said, "we then need to click on the Package Icon…"



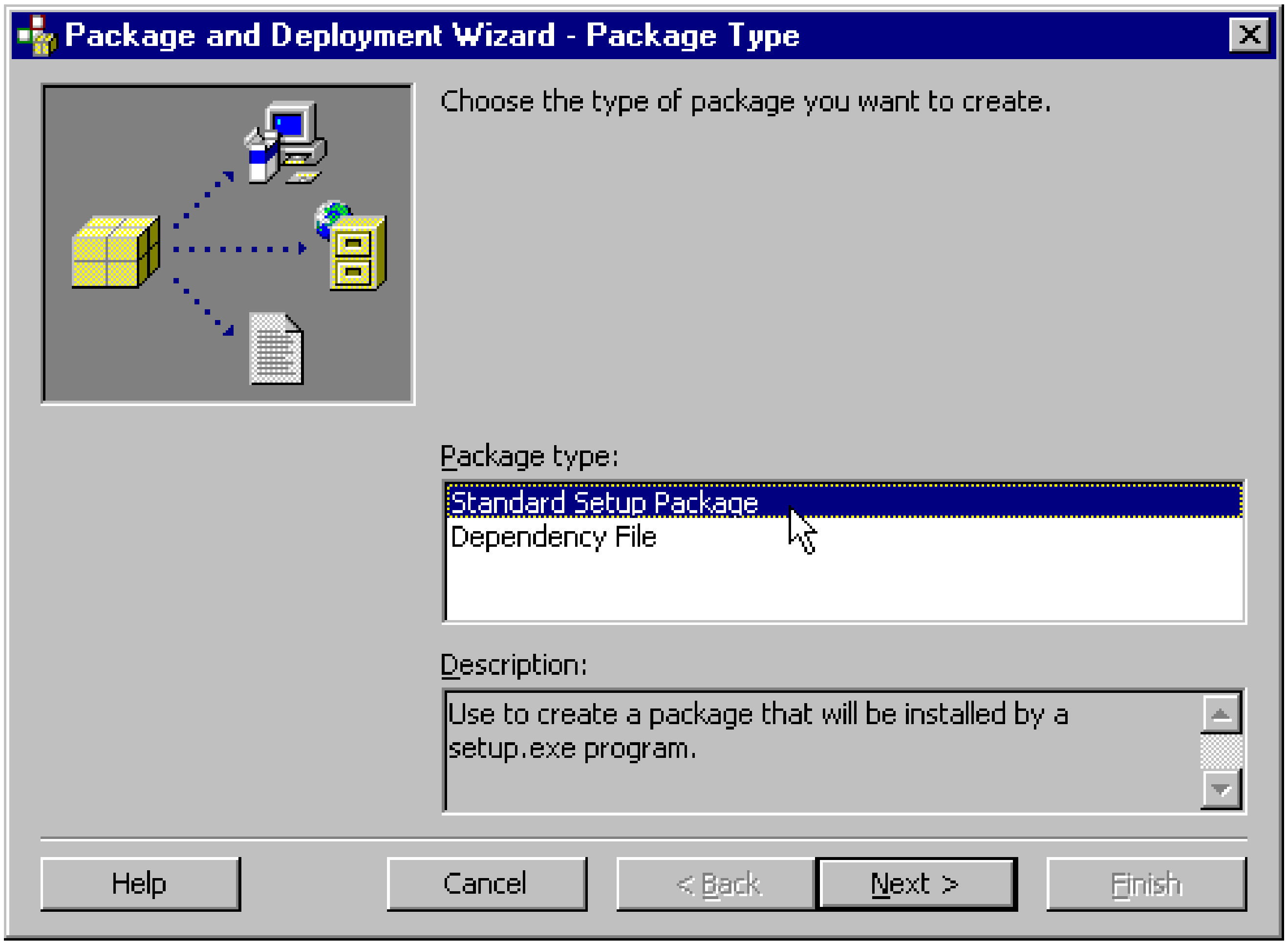
Select project: C:\vbfiles\practice\Question69.vbp

Bundle this project into a distributable package, such as an Internet cab or a setup program.

Send one of this project's packages to a distribution site, such as an Internet server.

Rename, duplicate, and delete your packaging and deployment scripts for this project.

I did, and after a few seconds, the following screen shot appeared.



Choose the type of package you want to create.

Package type:

Standard Setup Package
Dependency File

Description:

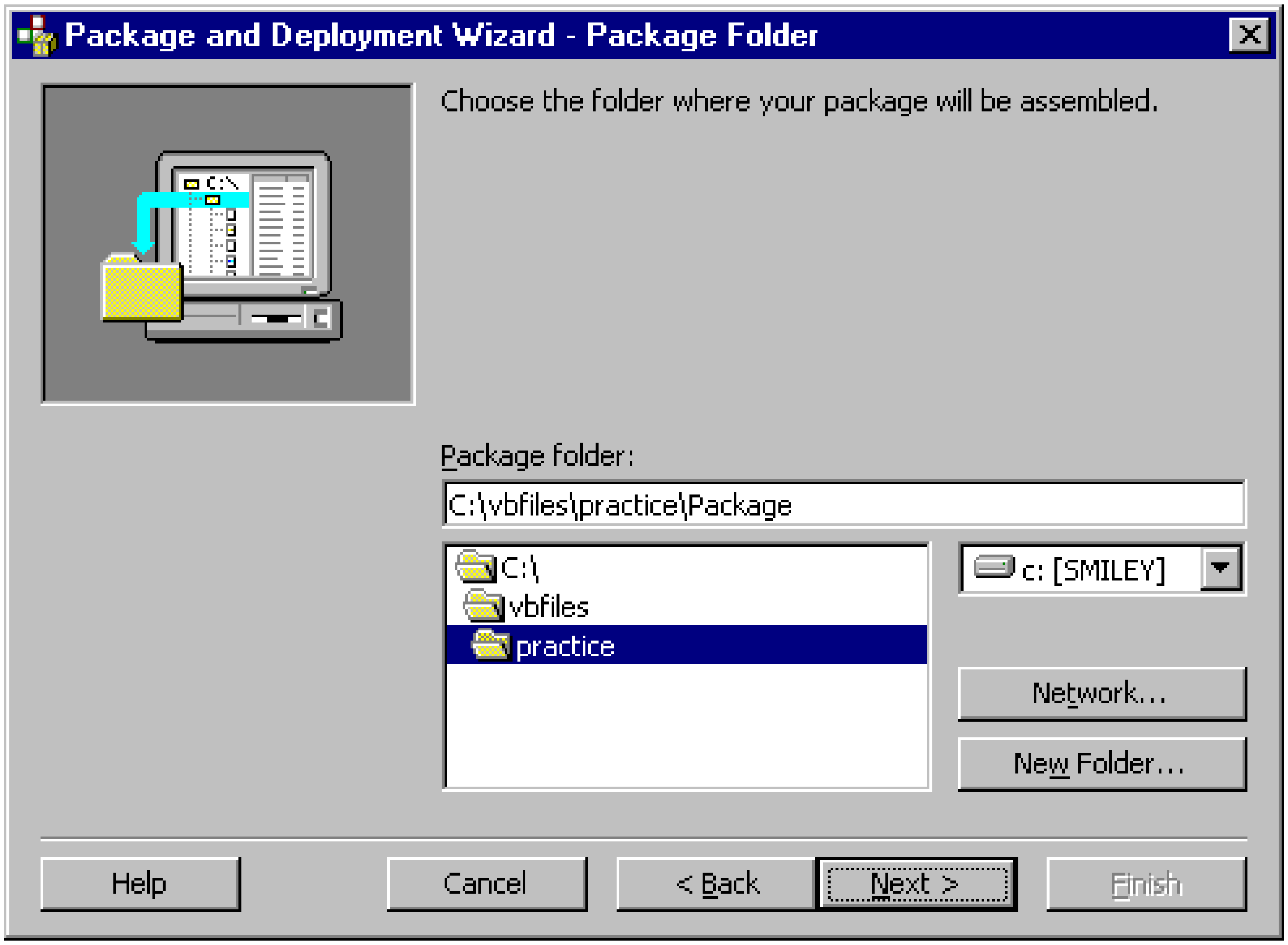
Use to create a package that will be installed by a setup.exe program.

"Just let the Wizard lead you through this Bess," I said. "For the most part, we'll just accept the Wizard's defaults, and click the Next button."

"What is this screen asking us to do?" Bess asked.

"The Wizard is asking us to specify the Package type," I said. "We want to create a Standard Setup Package, which will then allow us to setup.exe program. A Dependency File would just list the files necessary to run our executable on another PC---like the list of DLL's you originally asked me for."

I then selected **Standard Setup Package**, clicked on the Next button, and the following screen shot was displayed.



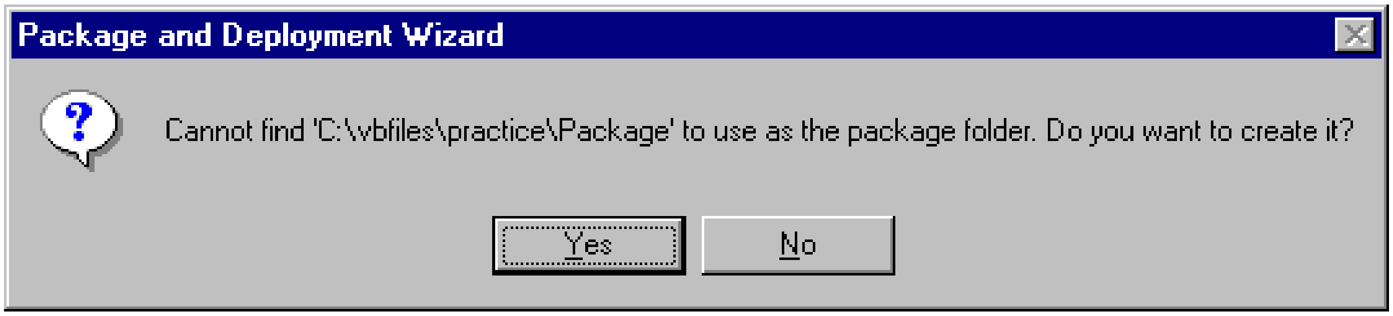
Choose the folder where your package will be assembled.

Package folder:

C:\vbfiles\practice\Package

"This screen is prompting us to select a location for our Package directory or folder," I said. "The Package is just a group of temporary files, and the Wizard wants to know where we want to store them. By default, the Wizard will create a Package directory 'underneath' of the directory where the executable is located. As is the case most times with wizards, the default is usually fine---so we'll just click on the Next button here."

I did, and this screen shot appeared.

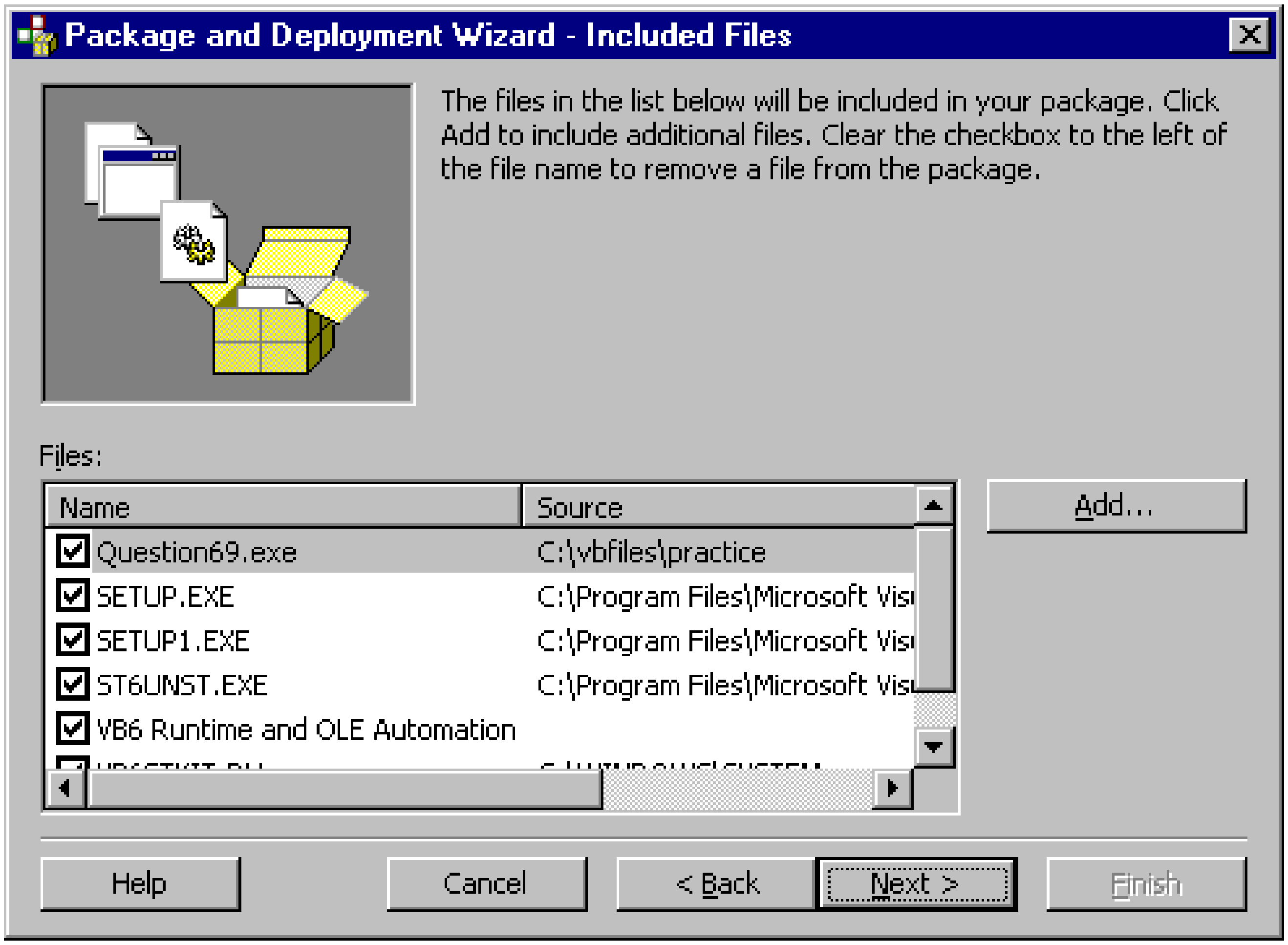


Cannot find 'C:\vbfiles\practice\Package' to use as the package folder. Do you want to create it?

"Uh oh…" I heard Bess say. "That looks like a problem…"

"It's OK," I said. "The Wizard is just telling us that a Package Directory does not yet exist in our Practice directory, and wants to know if it's OK to create one. If we click the 'Yes' button here, the Wizard will create the directory or folder for us."

I clicked on the Yes button, and this screen shot was displayed.



The files in the list below will be included in your package. Click Add to include additional files. Clear the checkbox to the left of the file name to remove a file from the package.

Files:

Name	Source
✓ Question69.exe	C:\vbfiles\practice
✓ SETUP.EXE	C:\Program Files\Microsoft Visı
✓ SETUP1.EXE	C:\Program Files\Microsoft Visı
✓ ST6UNST.EXE	C:\Program Files\Microsoft Visı
✓ VB6 Runtime and OLE Automation	

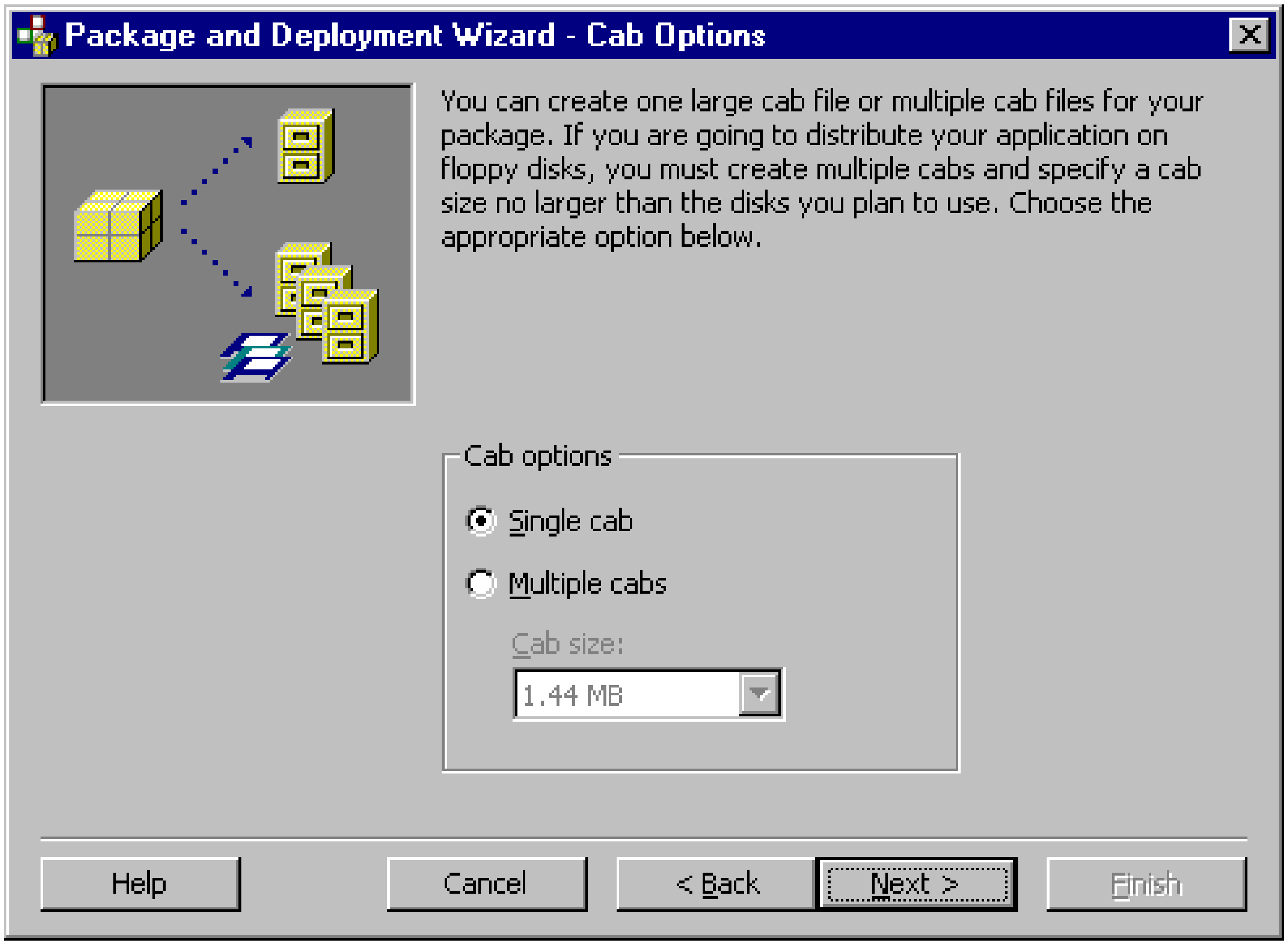
"I'm just guessing," Bess said, "but are these the files that Harry needs to run my executable on his PC?"

"Some of them are," I said. "Our executable and those DLL's I mentioned are included in this list of files---if you scroll down far enough, that is. Other files are those that the Setup program that will be placed on the diskette needs to run. I should mention that if we click on the Add button here, the Wizard will provide us with an opportunity to add any other files to our Package that it might be necessary for our program to have at run time."

"Such as?" Bess asked.

"Such as," I said, "graphic files that we might load into the Form or PictureBox using the LoadPicture method, disk files that the program reads at startup, a database that we intend to write records to during the program run, etc …We have no files to add at this time, so all we need to do here is click on the Next button."

I did that, and the following screen shot was displayed.


Package and Deployment Wizard - Cab Options

You can create one large cab file or multiple cab files for your package. If you are going to distribute your application on floppy disks, you must create multiple cabs and specify a cab size no larger than the disks you plan to use. Choose the appropriate option below.

Cab options

[✓ Single cab]

Multiple cabs

Cab size: 1.44 MB

I gave Bess and the viewing audience a chance to read this screen.

"This is a bit confusing," Bess said. "What do we do now? What's a Cab?"

"I agree" I said, "the first time I saw this screen, I spent about 5 minutes reading it and re-reading it. But I can summarize this pretty easily. 'Cab' stands for cabinet, and it forms the basis of the Distribution package in Visual Basic. There are many options for creating and distributing Cabinet files---for example, as a single file over a network, as a writable CD-ROM device, over the Internet, and in our case via floppy diskettes. Since we will be distributing our package via floppy diskettes, we need to specify the Multiple cabs option here."

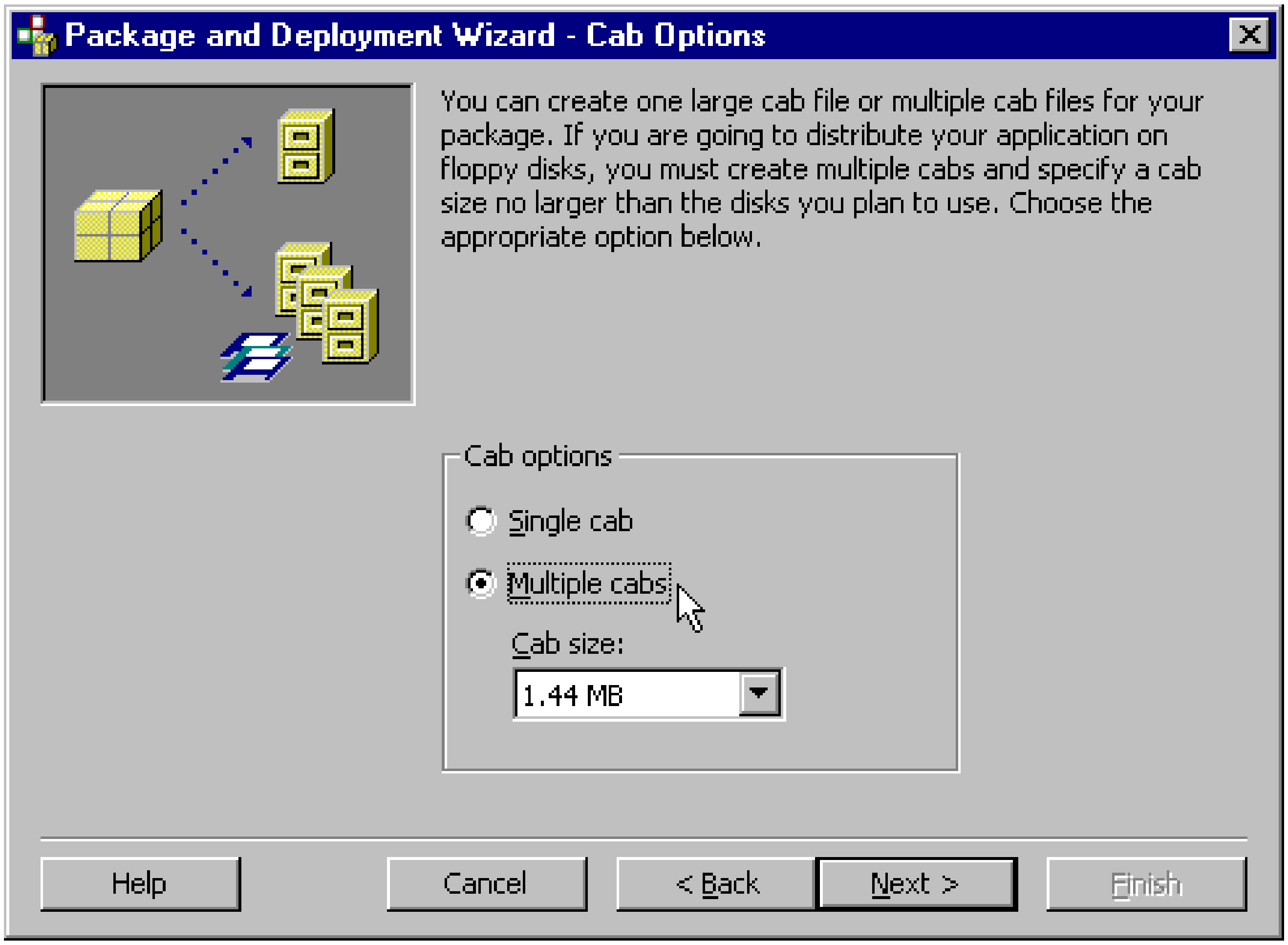
"Why Multiple cabs?" Bess asked.

"That's because," I replied, "the Wizard knows that even the smallest Visual Basic executable requires at least two floppy diskettes for the distribution files."

"Is that right?" Bess inquired. "Those DLL's must be pretty large!"

"The Visual Basic runtime DLL," I said, "which is just one DLL necessary to place on your husband's PC, is over a megabytes and half---which means it takes more than one floppy just to store that file. Let's select Multiple cabs here…"

I did, and also accepted the default Cab size of 1.44 MB---the capacity of my floppy diskette drive.



Package and Deployment Wizard - Cab Options

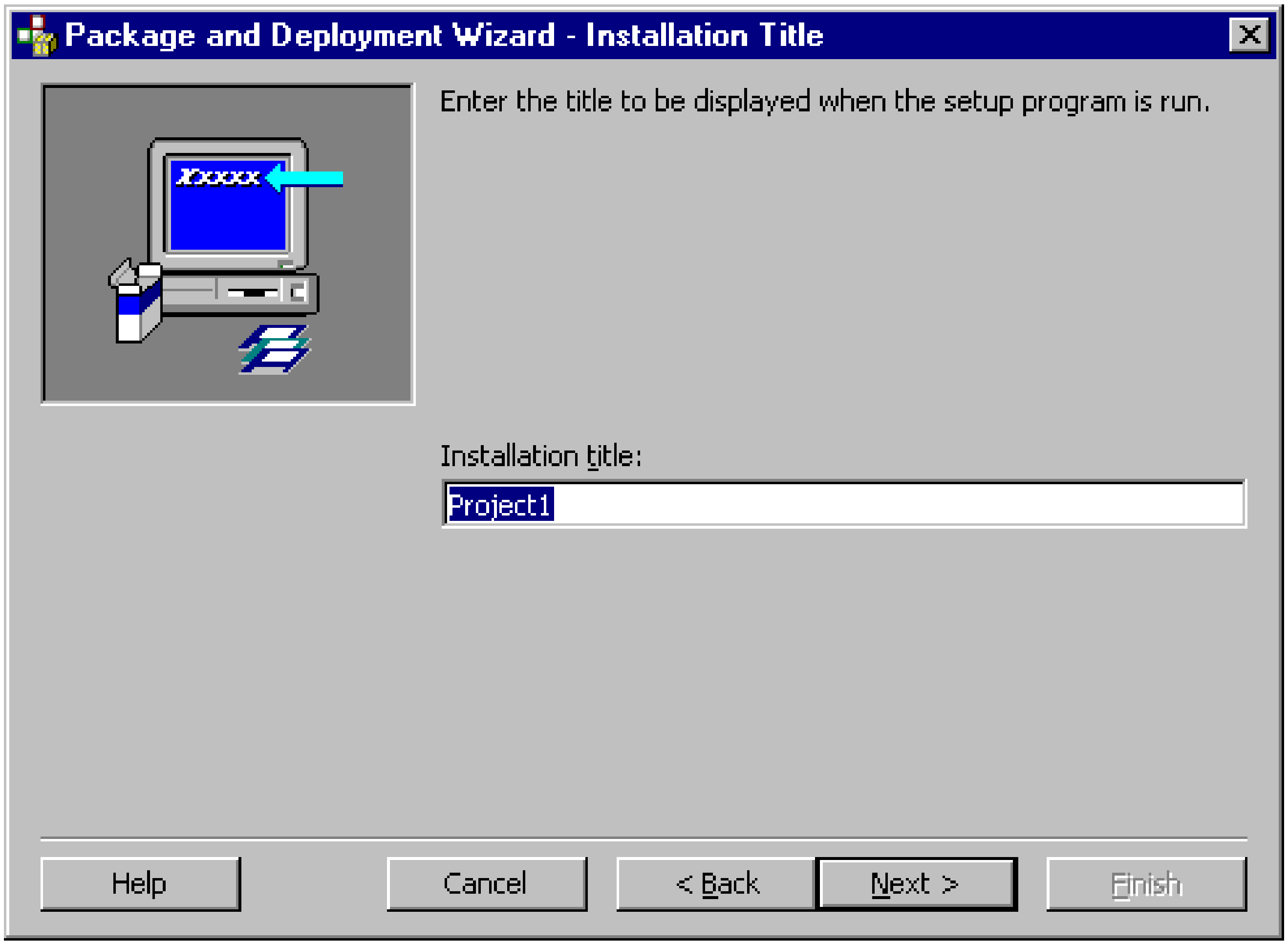
You can create one large cab file or multiple cab files for your package. If you are going to distribute your application on floppy disks, you must create multiple cabs and specify a cab size no larger than the disks you plan to use. Choose the appropriate option below.

Cab options

Single cab [✓ Multiple cabs]

Cab size: 1.44 MB

I then clicked on the Next button, and this screen shot was displayed to Bess and the viewers at home.



Enter the title to be displayed when the setup program is run.

Installation title:

Project1

"What's this asking us to do?" Bess said. "I thought we had changed the name of our executable from Project1.exe to Question69.exe?"

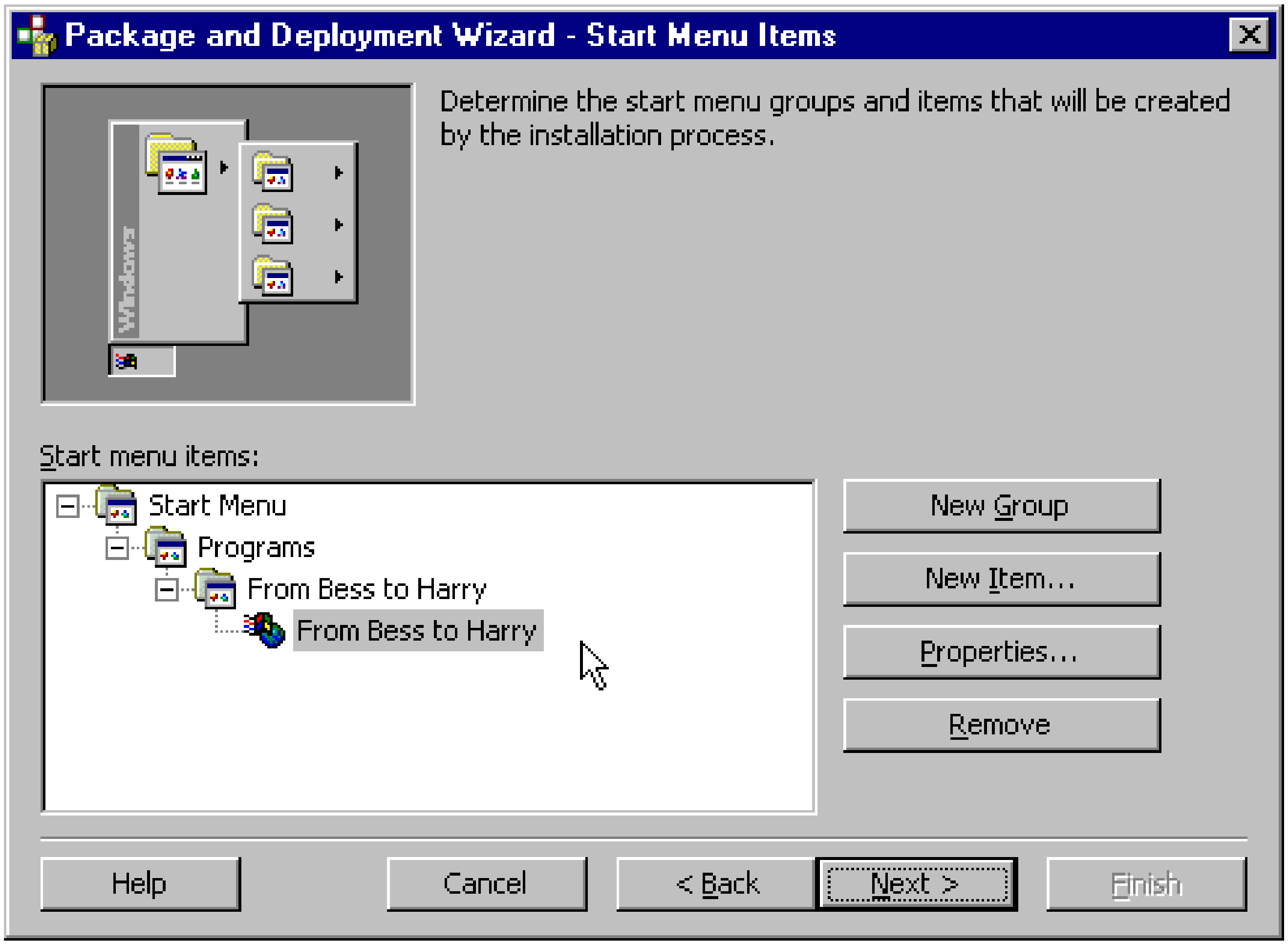
"I realize that I forgot to change the name of our project in Visual Basic," I said. "That's where this default for the Installation title is coming from."

"What's the Installation title?" Bess asked.

"As I mentioned earlier," I replied, "the Wizard will generate a professional quality setup program for us. Part of that is a very spiffy looking splash screen when it fires up---and that's what we're being asked about here---the title for that splash screen. How about if we get fancy Bess, and customize the title to read 'From Bess to Harry' for now."

"That's great," she said, "I know he'll love that."

I entered 'From Bess to Harry' as the Installation title, and clicked on the Next button. The following screen shot was then displayed.



Determine the start menu groups and items that will be created by the installation process.

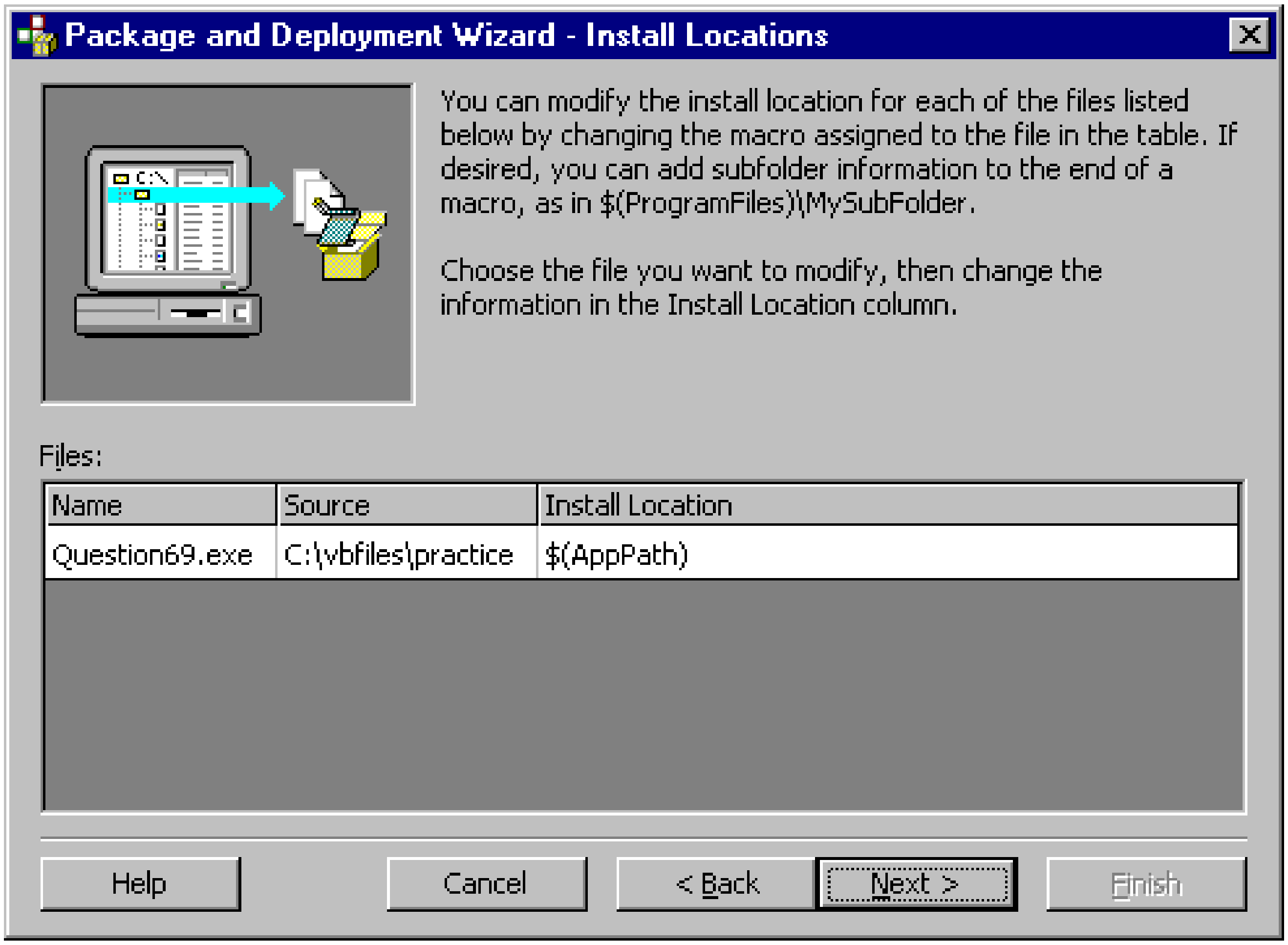
Start menu items:

- Start Menu
  - Programs
    - From Bess to Harry
      - From Bess to Harry

"What's that?" Bess asked.

"This is a neat little feature of the Wizard," I said, "It allows us to specify exactly where the program will be installed on Harry's Start Menu. By default, our program will be installed in the Program group of the user's Start Menu---but we could change that if we wanted. My advice is to go with the defaults here Bess. That means that the Wizard will create a Program Group called 'From Bess To Harry' on Harry's PC."

I accepted the default by clicking on the Next button.


Package and Deployment Wizard - Install Locations

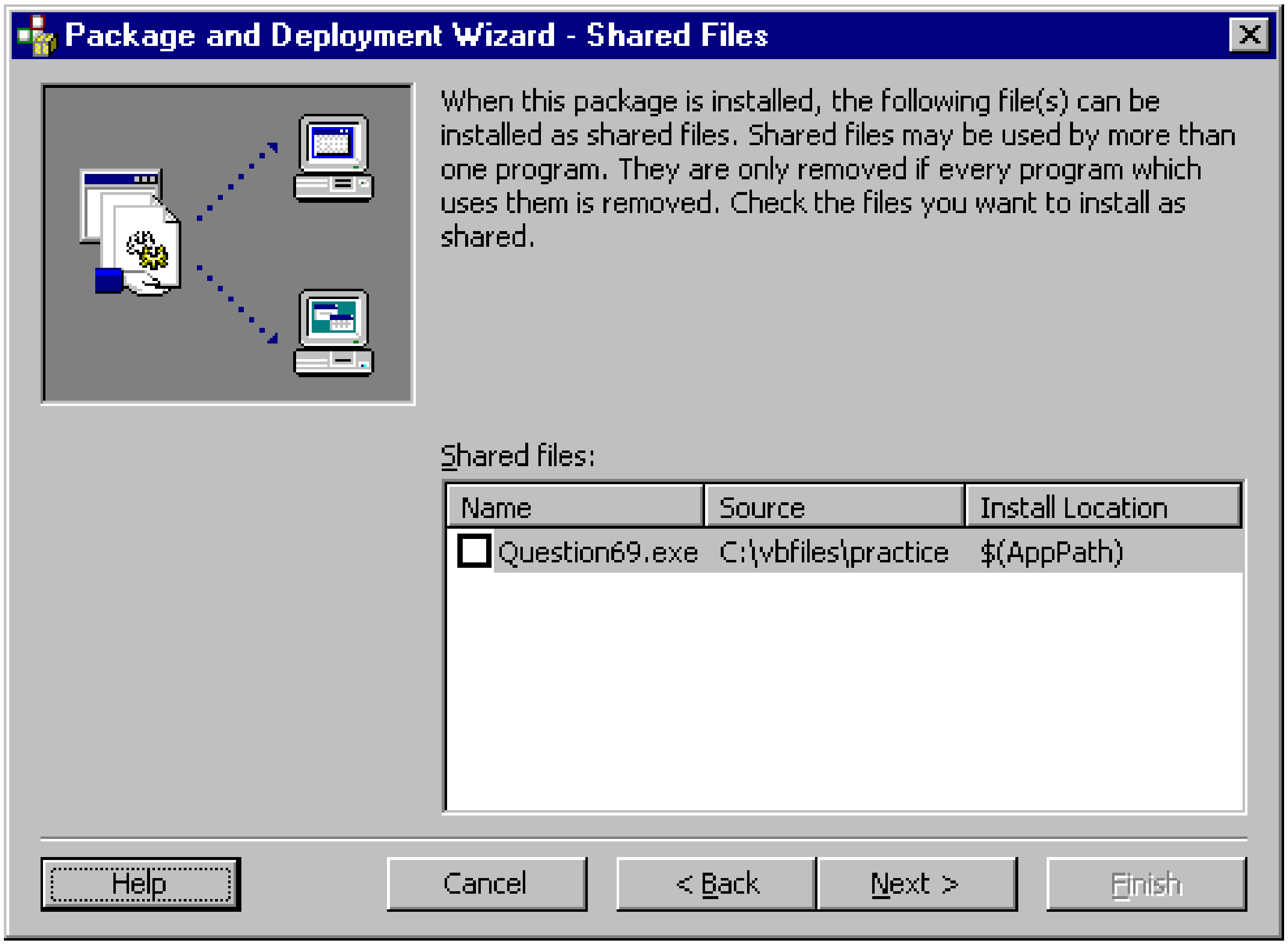
You can modify the install location for each of the files listed below by changing the macro assigned to the file in the table. If desired, you can add subfolder information to the end of a macro, as in $(ProgramFiles)\MySubFolder.

Choose the file you want to modify, then change the information in the Install Location column.

Files:

Name	Source	Install Location
Question69.exe	C:\vbfiles\practice	$(AppPath)

"Whereas the previous screen allowed us to specify where the icon representing our executable would be placed in regards to the user's Start menu," I said. "This screen allows us to specify the actual location of the executable. As you can see, the default location is the same directory as our project is saved in on my studio PC. Again, unless you have a good reason to change this location, you should probably accept the default. Let's do that now by clicking on the Next button."



Package and Deployment Wizard - Shared Files

When this package is installed, the following file(s) can be installed as shared files. Shared files may be used by more than one program. They are only removed if every program which uses them is removed. Check the files you want to install as shared.

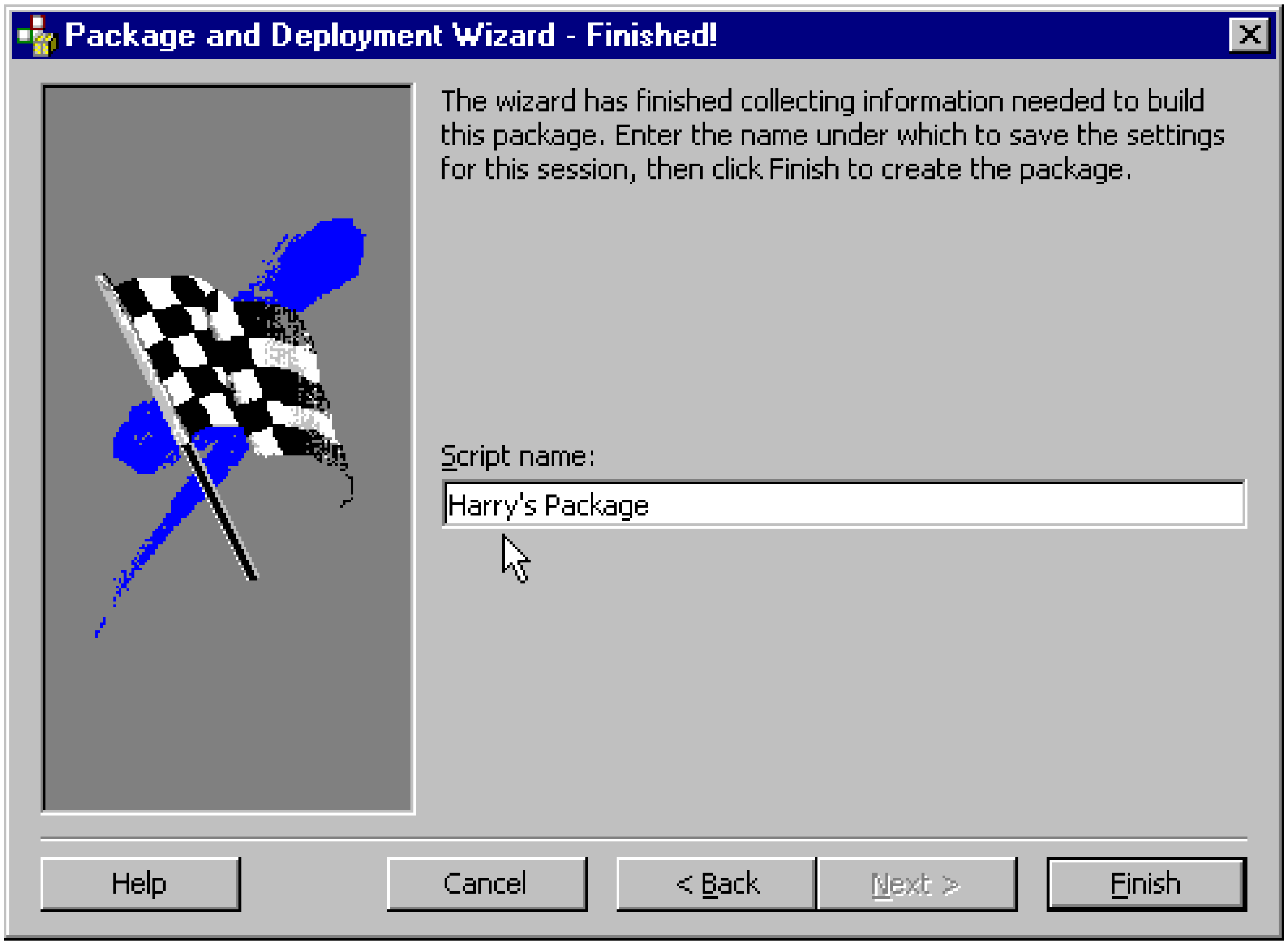
Shared files:

Name	Source	Install Location
☐ Question69.exe	C:\vbfiles\practice	$(AppPath)

"More confusion," I heard Bess say. "What's a shared file?"

"Try not to worry Bess," I replied. "You should be getting the feeling that if you continue to accept the defaults in the Wizard, everything will work out fine. The defaults are really set for a stand alone executable, such as the one you want to give Harry. This window is asking us if we want to designate our executable as a shared file---something we would do only if more than one application needed to use it. In our case, our executable is definitely 'stand alone', so we just need to click on the Next button here."

I did and the following screen shot was displayed.

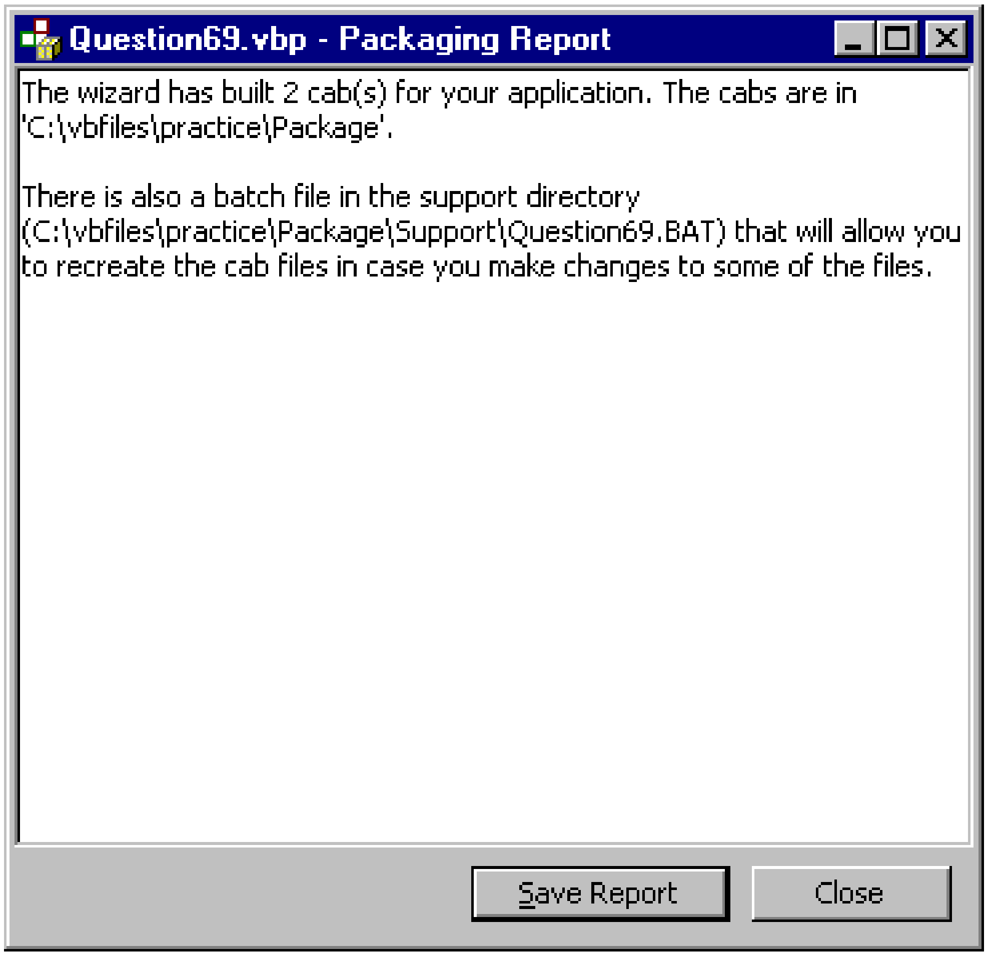


The wizard has finished collecting information needed to build this package. Enter the name under which to save the settings for this session, then click Finish to create the package.

Script name: Harry's Package

"Ordinarily," I said, "I like to accept defaults in the Wizard---but when it comes to a Script name, I like to customize them. Let's change the name of the Script to Harry's Package, and click on the Finish button."

I did so, and the following screen shot was displayed.



The wizard has built 2 cab(s) for your application. The cabs are in 'C:\vbfiles\practice\Package'.

There is also a batch file in the support directory (C:\vbfiles\practice\Package\Support\Question69.BAT) that will allow you to recreate the cab files in case you make changes to some of the files.

"This is a Packaging Report," I said. "It provides us with a little information about our package---the number of cabs in it, and its location. At this point, you can either save the report, or just close the window."

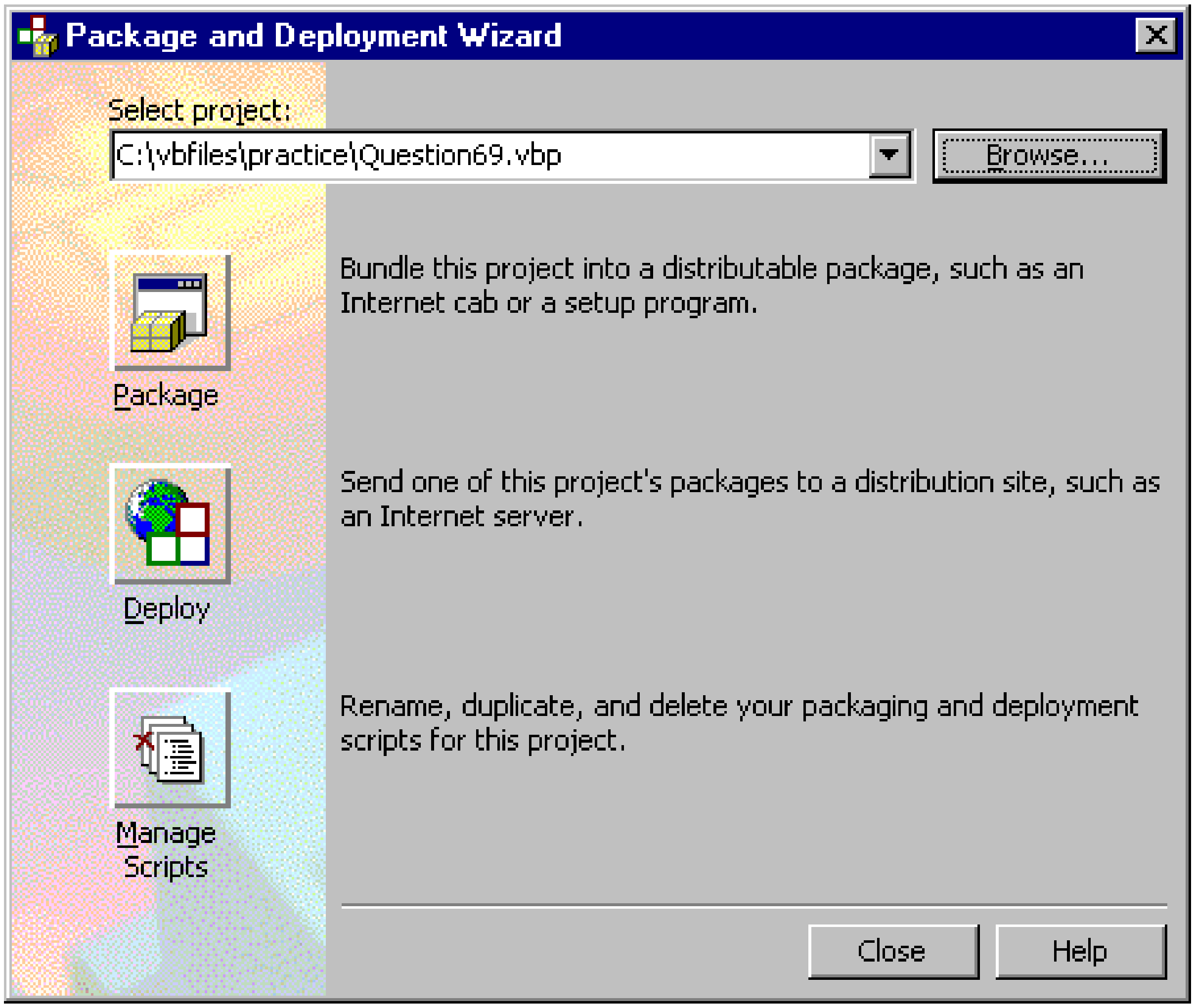
"Great, we're finished," I heard Bess say. "I guess that wasn't too bad after all. But I must have missed something somewhere. I don't remember you inserting any diskettes."

"You’re right Bess," I said. "I didn't insert any diskettes. The first time I ran this myself, I figured I was done at this point, and asked myself the same question. I didn't realize that now that I had the Package, I then needed to deploy it---in our case, to floppy diskettes."

"So this is really a two phase operation," Bess said.

"That's right Bess," I said. "First we create the package, then we deploy it."

I closed the Packaging Report Window, and we were back to the initial screen of the Deployment Wizard.


Package and Deployment Wizard
Select project:
C:\vbfiles\practice\Question69.vbp
Browse...
Package
Deploy
Manage Scripts
Close
Help

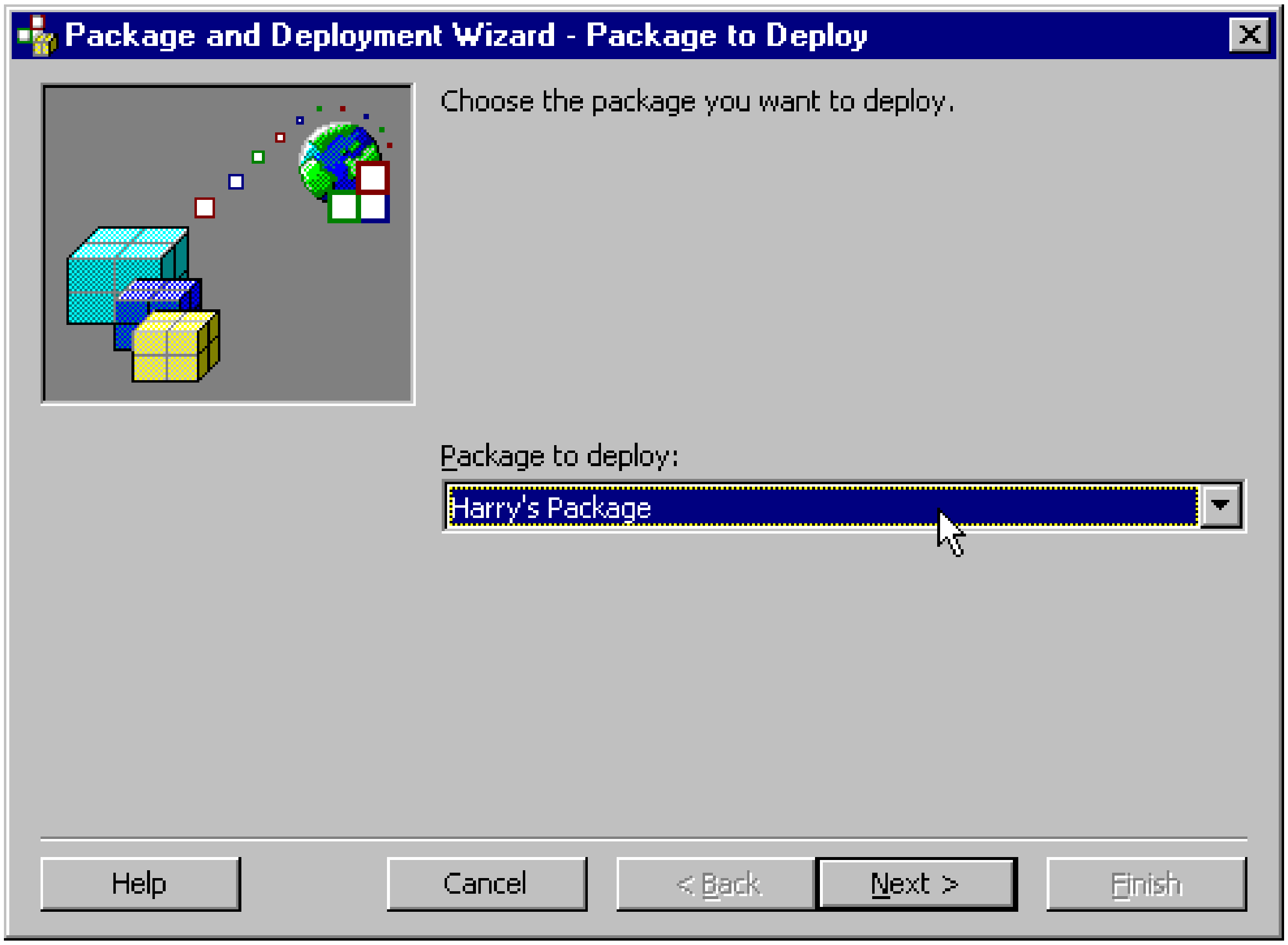
Bundle this project into a distributable package, such as an Internet cab or a setup program.

Send one of this project's packages to a distribution site, such as an Internet server.

Rename, duplicate, and delete your packaging and deployment scripts for this project.

"It won't be much longer now Bess," I said. "From this main screen of the Wizard, we now need to select Deploy."

I did, and the following screen shot was displayed.



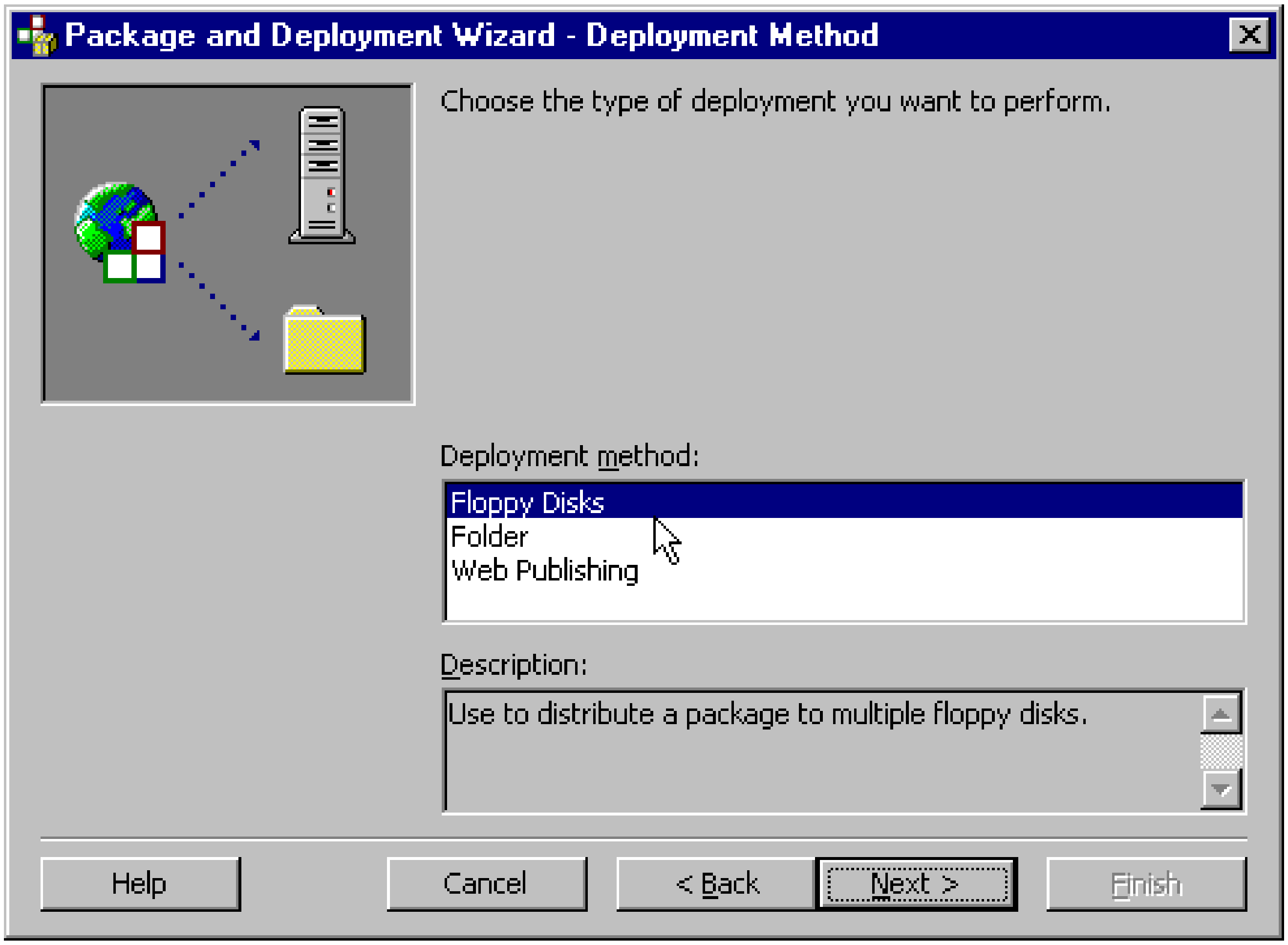
Choose the package you want to deploy.

Package to deploy:

Harry's Package

"This screen is just asking us to select a package for deployment," I said. "A few minutes earlier, we named our Package 'Harry's Package'---that makes it easier to pick out of a list here. Once the Package is selected, we just need to select the Next button."

I did, and the following screen shot was displayed.



Choose the type of deployment you want to perform.

Deployment method:

Floppy Disks
Folder
Web Publishing

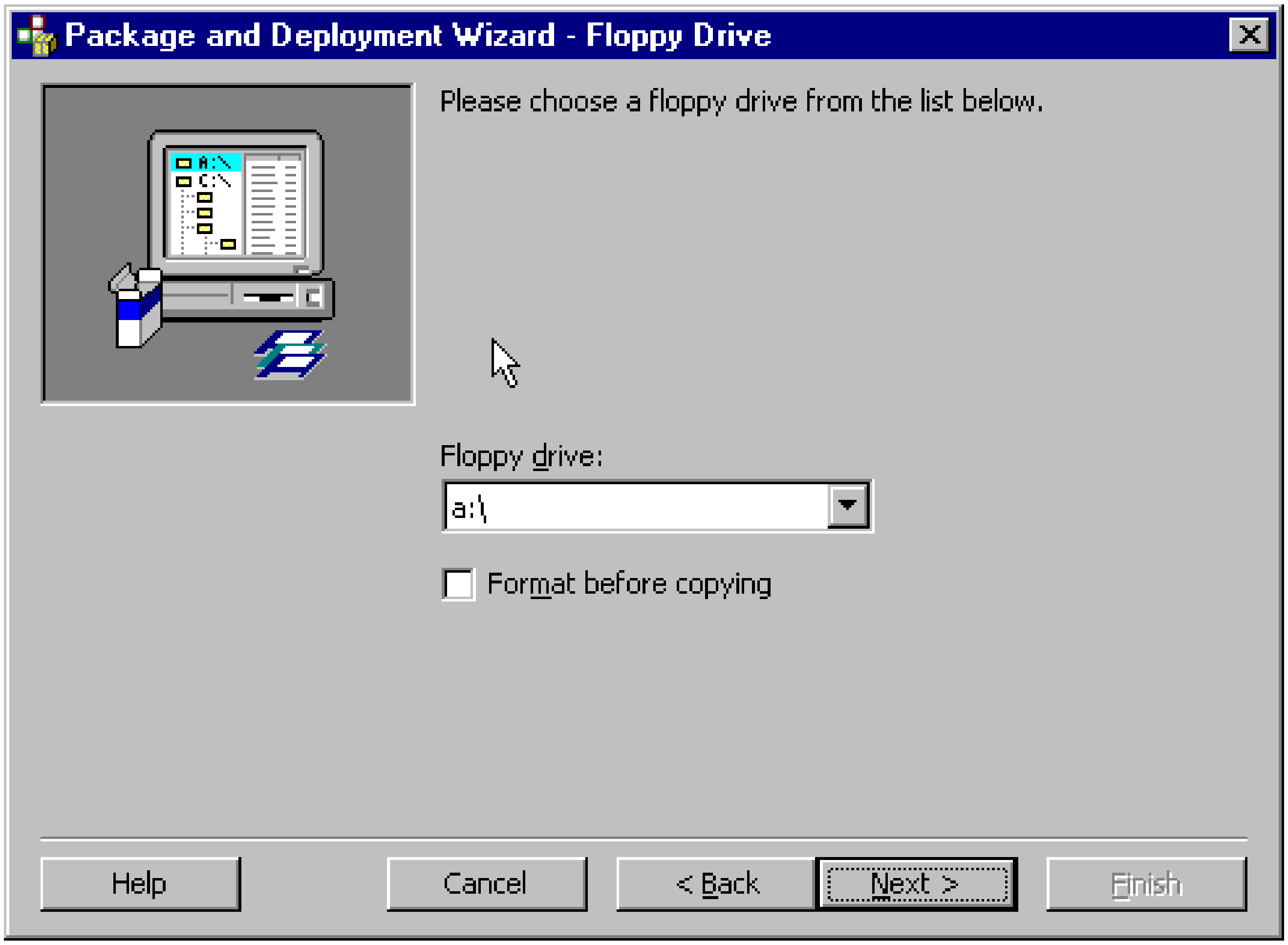
Description:

Use to distribute a package to multiple floppy disks.

"The Wizard wants to know about our Deployment method," I said. "In our case, we need to select Floppy Disks, and then click on the Next button."

"Didn't we tell the Wizard Floppy Disks a while back?" Bess asked. "In the Package Creation portion of the wizard."

"When we specified Multiple Cabs," I said, "we needed to tell the Wizard the size of our distribution media--which was 1.44 MB. Despite that, this is really the first time we've told the Wizard that we want to place our Setup.exe on floppies."



Please choose a floppy drive from the list below.

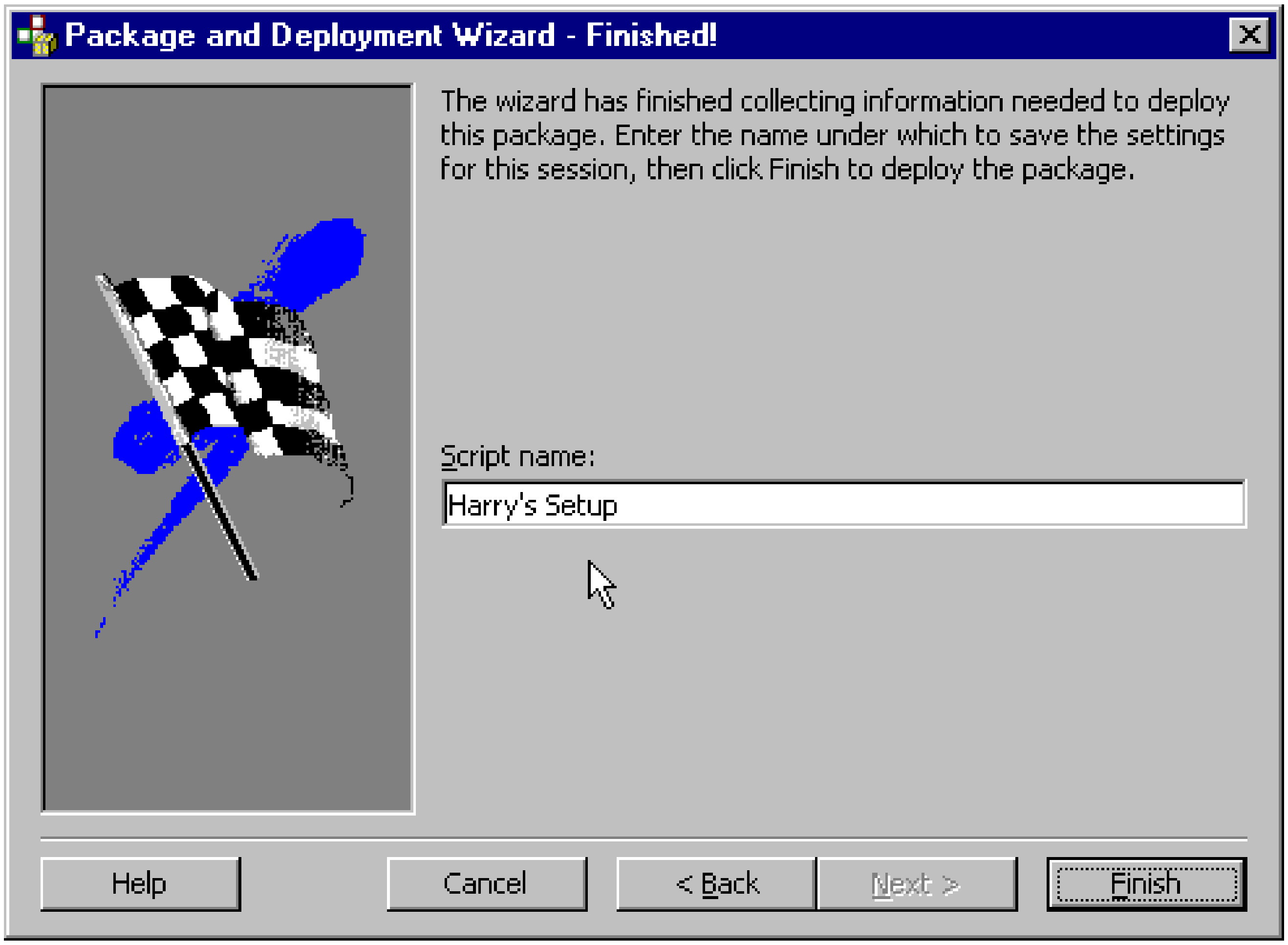
Floppy drive: a:\

Format before copying

"This one's easy," Bess said. "We just need to specify the drive letter for our floppy drive."

"That's right," I agreed. "Do you see that there's an option here to Format the diskette before the Wizard begins copying. It's a good idea to check that 'on'---the Wizard requires 'empty' formatted diskettes anyway. If you format before copying, you'll verify the data integrity of the diskette, and ensure that it's empty prior to copying the setup files to it."

I checked 'on' the option to Format before copying, and then clicked on the Next button. The following screen shot was displayed.



The wizard has finished collecting information needed to deploy this package. Enter the name under which to save the settings for this session, then click Finish to deploy the package.

Script name: Harry's Setup

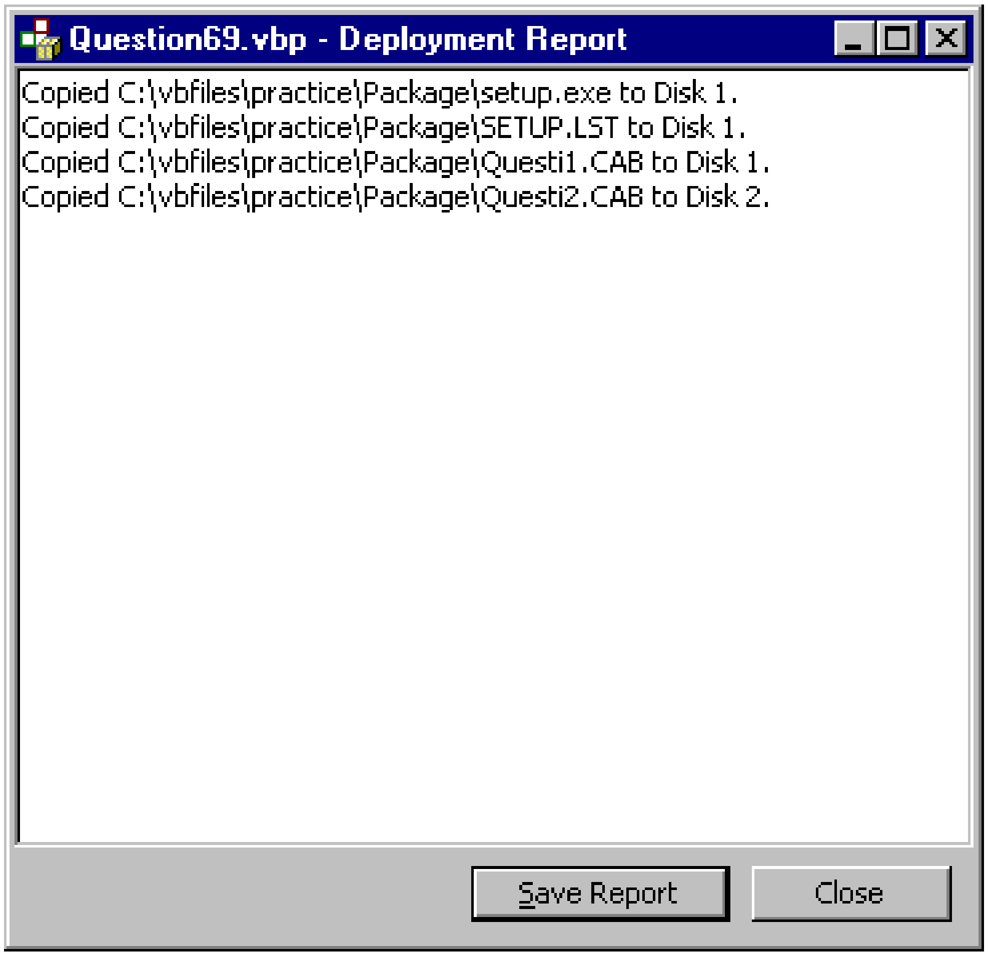
"The Wizard wants another name?" Bess asked. "I though we already named this script."

"That was the Package Script," I said. "Now the Wizard wants to save the settings for this Deployment in a Script name. We need to take care not to name the Script the same name as the Package Script. Let's call this 'Harry's Setup."

I did so, and clicked on the Finish button.

"Now we're getting close, Bess" I said.

We were then prompted to insert the first diskette into our floppy diskette drive, and after a minute or so, a prompt for the second diskette appeared. After a few seconds of copying files to the second diskette, the following screen shot appeared.



```
Copied C:\vbfiles\practice\Package\setup.exe to Disk 1.
Copied C:\vbfiles\practice\Package\SETUP.LST to Disk 1.
Copied C:\vbfiles\practice\Package\Questi1.CAB to Disk 1.
Copied C:\vbfiles\practice\Package\Questi2.CAB to Disk 2.
```

***69-23

"Another report?" I heard Bess say.

"Yes, another report, " I replied, "but this is the final one---the Wizard process is complete. This report is giving us a list of the files that it has copied to our diskettes."

"Does this mean I can just send those files off to Harry now?" she asked. "Where are the DLL's you mentioned. And what about our executable?"

"Our setup program," I said, "much like the others in use today, has compressed the files that it placed on the diskettes. The .cab files you see are the compressed files, and they will be uncompressed when Harry runs the Setup.exe program that you see listed in the report."

"So all I need to do is send Harry the diskettes that the Wizard produces?" she asked.

"That's right," I said, "Just tell Harry to insert the first diskette into his diskette drive, open up Windows Explorer, and double click on the Setup.exe file---he'll be prompted for the rest. I guarantee you he'll be wildly impressed with your setup program, and in no time at all, he'll be running the program you wrote for him."

"I want to thank you for taking the time to show me how to do this," Bess said. "I'm sure there are others in the viewing audience who had the same problem."

"You're quite welcome Bess," I said. "and thanks for calling. By the way, I mentioned earlier that versions of Visual Basic prior to Version 6 have a different version of the wizard than the one I demonstrated today. It's called the Application Setup Wizard. It's similar in many ways to the one we used here today--- but there are some obvious differences---the main one being that the Visual Basic 5 Wizard doesn't first create a package on your hard drive---it starts immediately with the creation of the Setup.exe to your floppy diskette. And now onto our final caller of the day."