

CHAPTER 1

GETTING STARTED

The very first thing that we need to do is get your development system ready. This chapter walks you through the process from the very beginning. We'll start by evaluating your system and work our way to preparing your system for your first repackaged application.

Analyzing Your System

To begin with AdminStudio is a big application and uses lots of system memory. In addition, we'll be using VMWare along with AdminStudio. VMWare also consumes significant disk space and requires additional system memory.

For the operating system, AdminStudio requires Microsoft Windows 2000 or better. This should include Internet Explorer 5.5 or better. If you're interested in using the Pre-deployment Test Wizard or the SMS Web Console, you'll also need Microsoft Internet Information Services 5.0 or better.

As far as hardware requirements, Macrovision recommends at least 500 MHz or better. It should make sense the faster the better. If I would give a recommendation it would be for the fastest processor you (or your company) can afford. This often isn't a choice that you can make so do the best you can. This is the same for system memory. Macrovision prefers that you have at least 512 Mbytes. I'm telling you that 2 Gbytes is ideal—especially if you're also using VMWare. Anything less will cause the utilities (Repackager, Quality Monitor, etc) within AdminStudio to run incredibly slow. This is an important: get as much system memory as you can!

In addition to memory, disk space is also a concern. We'll be saving our work directly on the computer. If disk space is limited, you may want to consider archiving your work remotely. Within VMWare, the VMImages each consume about 8 Gbytes of disk space. If you're only using one image then this shouldn't be a problem. If you want to use multiple VMImages this is something to think about.

In a perfect world, you'll have a wicked fast machine with 2 or more GBytes of system memory and several hundred GBytes of free disk space. Heck, let's even throw in multiple flat screen monitors. And this is just your development system. You may want to consider using a different computer for checking email and miscellaneous work.

Installing AdminStudio

AdminStudio comes in three flavors: Enterprise, Professional, and Standard. Each version offers varying degrees of functionality and varies greatly in price.

The Enterprise edition includes support for a centralized install location and individual logged-in users. This includes the capability for a centralized application catalog and report generation.

For an in-depth look at the differences between the three editions of AdminStudio, the following link to Macrovision provides a comparison table:
http://www.macrovision.com/products/flxnet_adminstudio/adminstudio/features/all.html

The Professional edition is the standalone version and is the version that we'll be using in this book. It has most of the functionality of the Enterprise edition and is intended for small (non-sharing) repackaging environments.

The Standard edition is a stripped down version of the Professional edition. Amongst other things the Standard edition does not include support for the Quality Monitor and some advanced capabilities of

the Application Catalog.

For the most part, installing AdminStudio is straightforward. The install program is good at letting you know if you're missing any requirements and it installs the ones that it can. The default destination directories are good defaults and in our case they are the ones that we'll use.

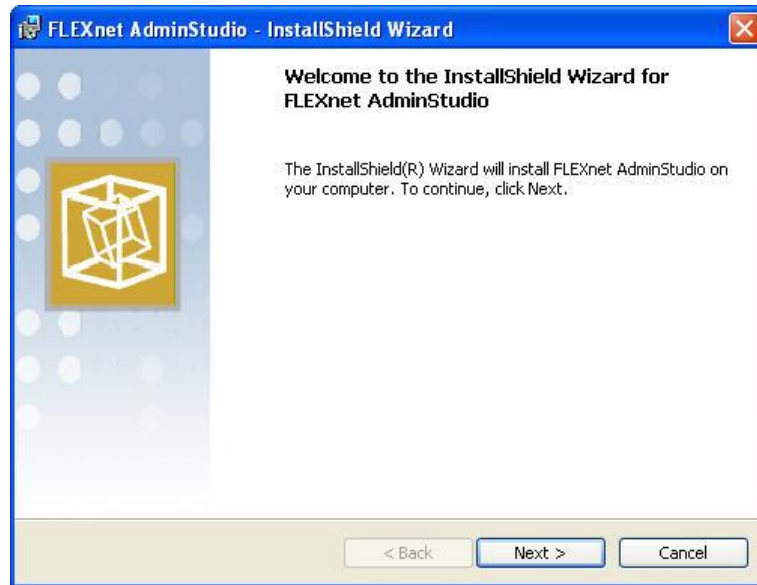


Figure 1. Running the AdminStudio install is a simple process. In most cases, the default values will work well.

Once you're ready to go, run the AdminStudio install. If you've downloaded the image from the Internet you'll be launching a single executable. If you're running from the CD image you'll be double-clicking on setup.exe.

If you don't have a serial number you'll end up with a 30-day evaluation copy. Macrovision has a very stringent copy protection system, so don't think the uninstall/re-install will extend the evaluation period.

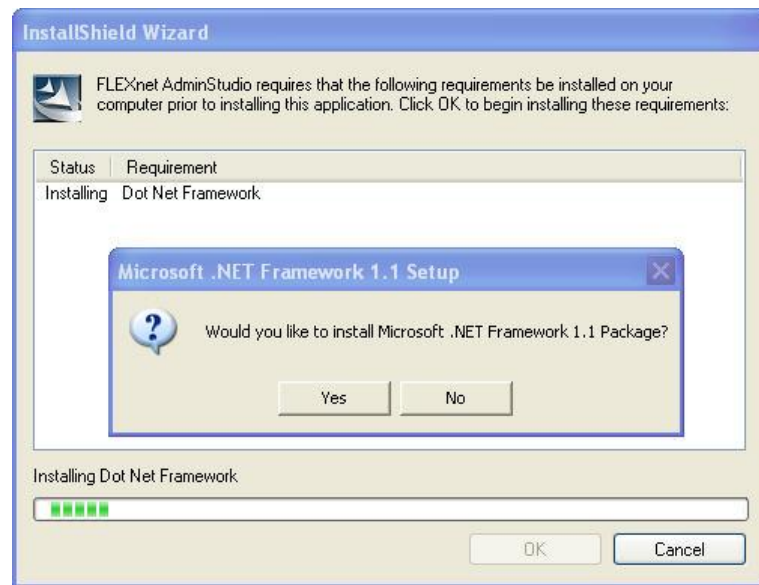


Figure 2. A key component to the installation of AdminStudio is .Net Framework 1.1. If this prerequisite isn't already installed, it will be installed for you..

The first check that's made is for .Net Framework. If you don't currently have version 1.1 or better installed, the AdminStudio install will want to install this version for you. After .Net is installed, the main AdminStudio install will begin.



Figure 3. Of course, the AdminStudio install provides the ubiquitous License Agreement dialog.

The AdminStudio install uses a standard dialog interface. There isn't much that needs to be changed from the default values.

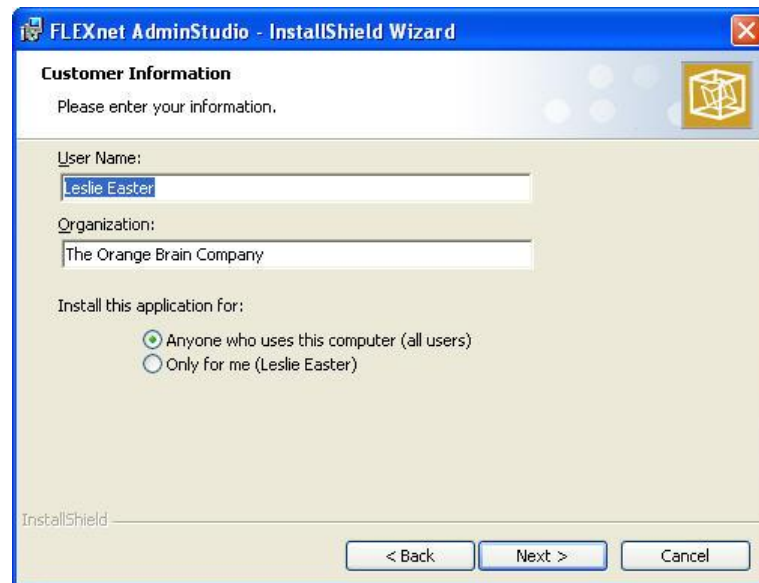


Figure 4. Besides prompting for user name and company name, the Customer Information dialog also requires information about how AdminStudio should be installed—per user or per machine.



Figure 5. From the Destination Folder dialog, you can alter where AdminStudio is installed on your computer.



Figure 6. In a shared environment, this may be one directory path you'll want to change.

One of the destinations is for the AdminStudio Shared Location. For single use, the default value is ideal. For multiple users that desire sharing, this location should be changed to reflect a common share point.

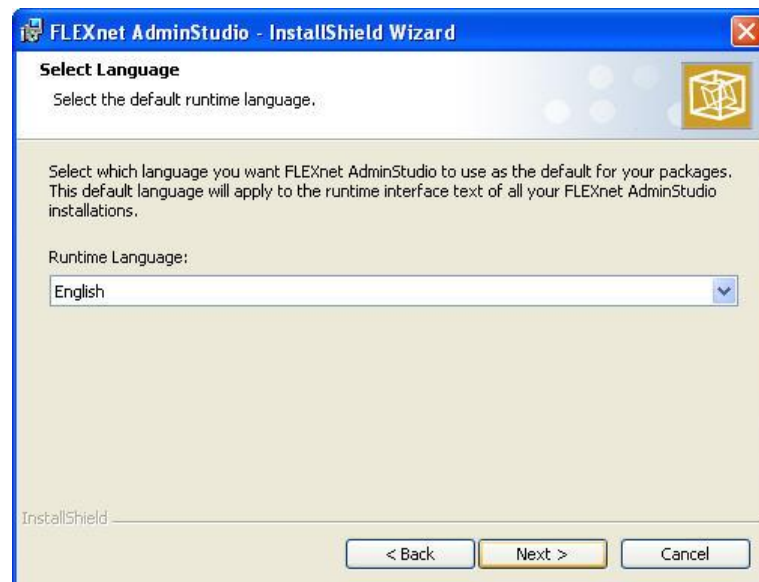


Figure 7. In this book, we're assuming that the English version of AdminStudio is installed. However language selection makes little difference in the actual use of the product.



Figure 8. At this point in the installation, we can choose from the standard setup types. In this book, we'll be installing the complete version of AdminStudio.

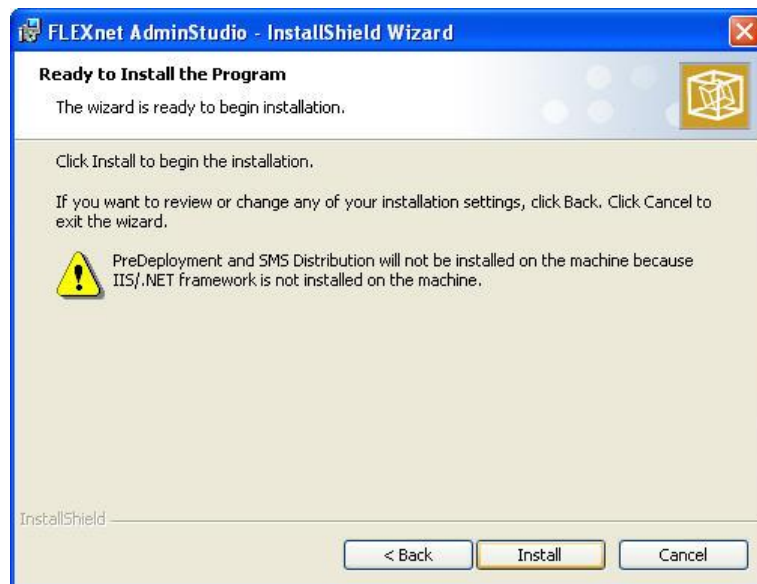


Figure 9. Before installing AdminStudio there is one last opportunity to make changes. The Ready to Install dialog also alerts us if specific functionality will not be available..

Several of the AdminStudio tools require *Internet Information Services* (IIS) to be present. If IIS is not installed on your computer, these tools are left out of the installation process.



Figure 10. The actual file transfer and configuration of AdminStudio takes about five minutes to run—results may vary depending on your machine configuration.



Figure 11. Once the AdminStudio install is finished, close the Finish dialog. Depending on your system configuration you may have to reboot.

After the install of AdminStudio, a variety of folders have been created. It's worth knowing where these folders are and what each of them is for. AdminStudio creates five distinct folder locations:

- AdminStudio Shared
- AdminStudio Common Files
- Merge Modules
- AdminStudio Application
- MSBuild

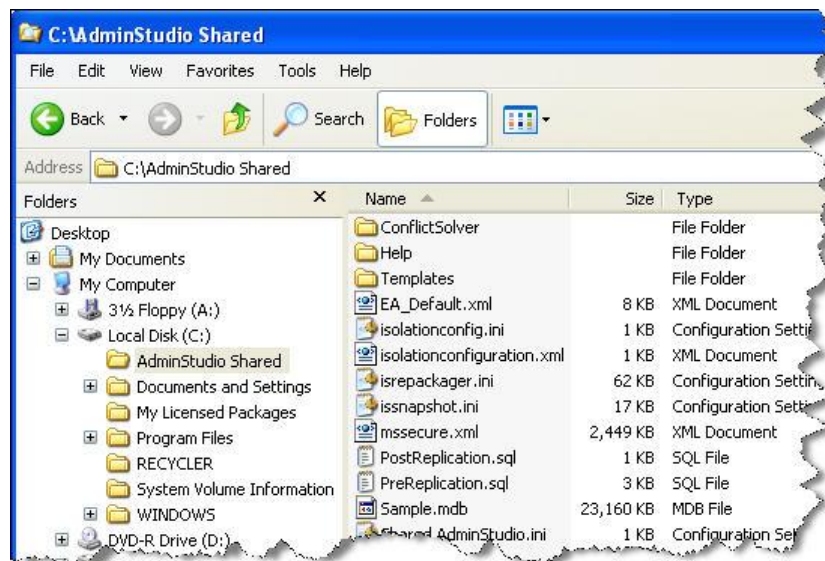


Figure 12. The AdminStudio Shared folder contains common application configuration and data base files.

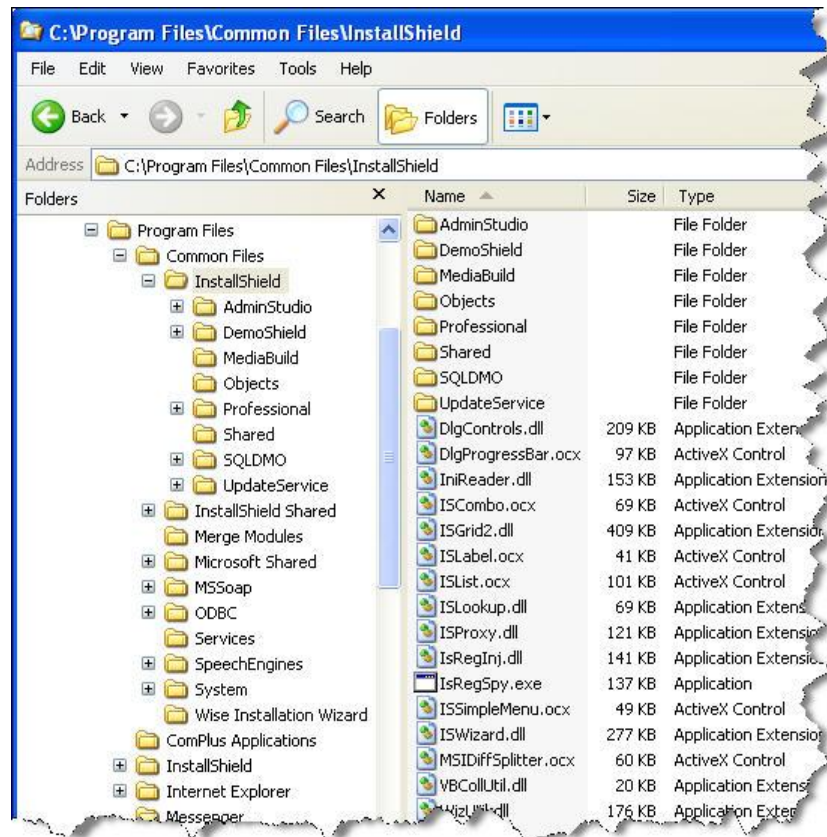


Figure 13. The AdminStudio Common folder contains files that are shared across the AdminStudio tools and utilities.

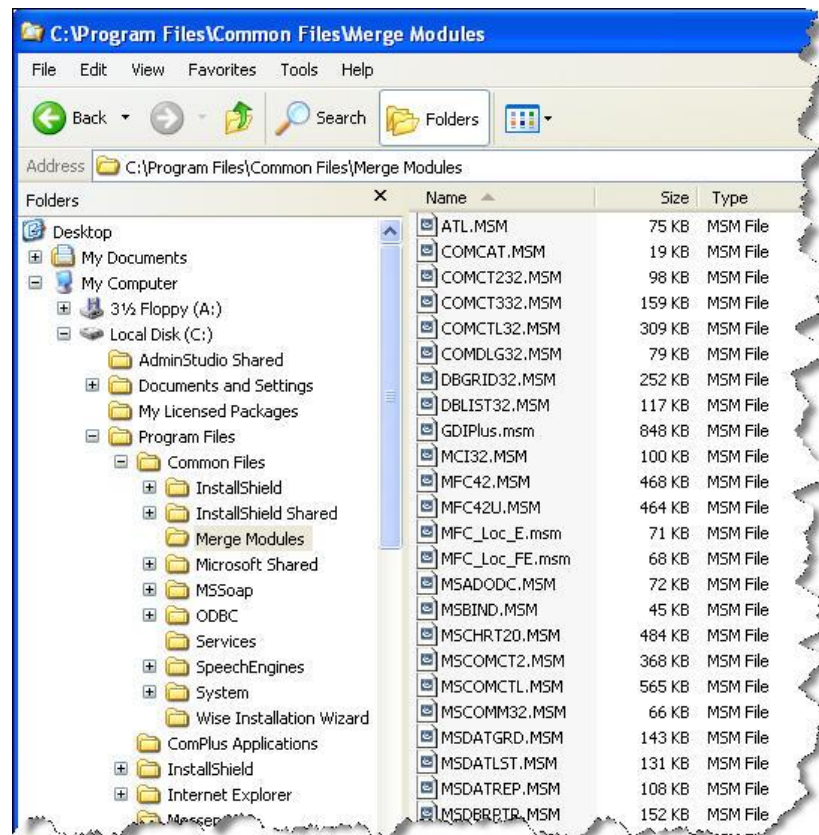


Figure 14. The AdminStudio Merge Modules folder contains the merge module files that are used by the AdminStudio Editor.

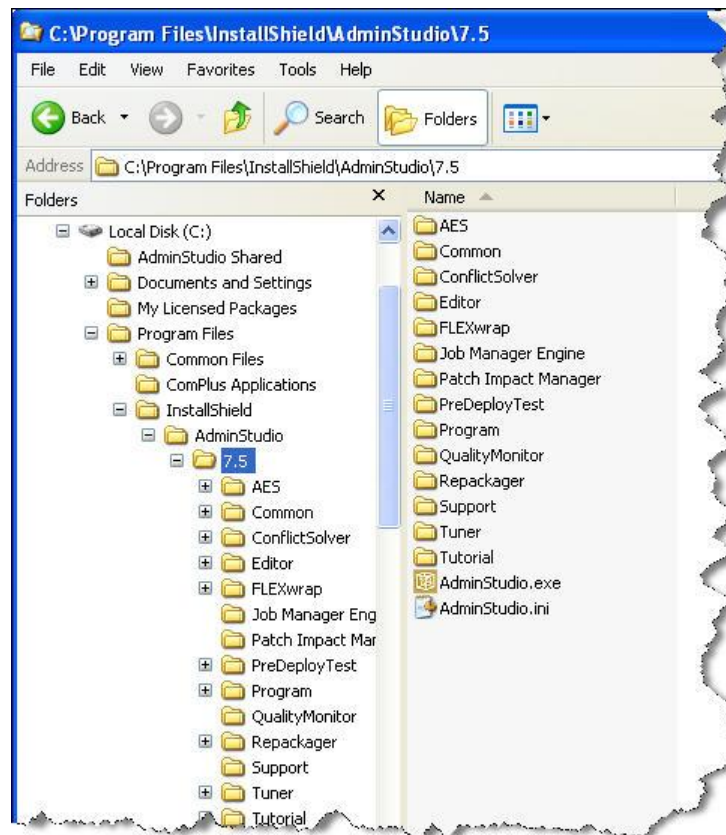


Figure 15. The AdminStudio Application directory is the heart and soul of the AdminStudio application.

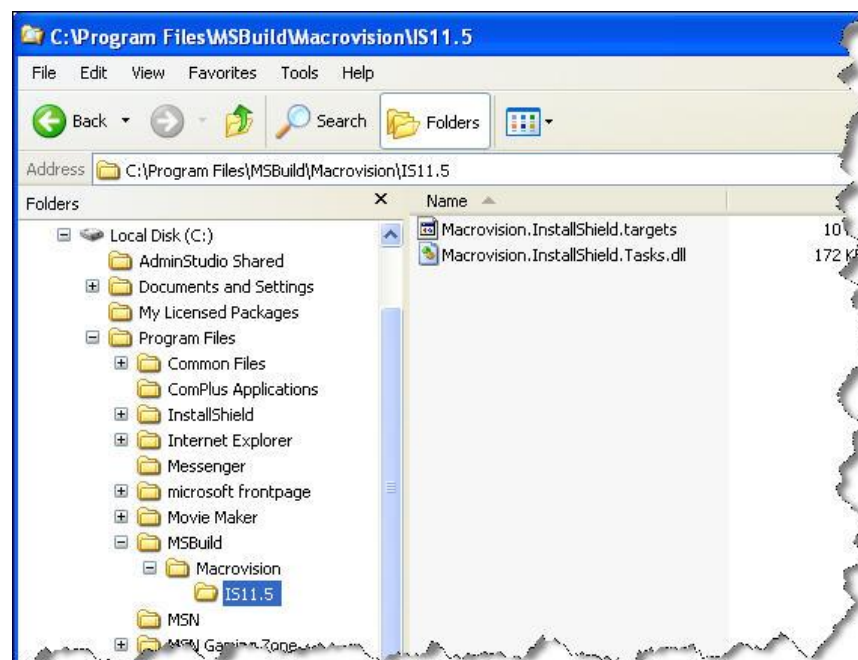


Figure 16. The MSBuild folder added by AdminStudio is extended functionality to support for the Microsoft Build Engine.

Setting up Your AdminStudio Work Space

Once AdminStudio is installed, we'll need to set-up our work space. For the most part, these are just simple conveniences. However, they will make the repacking process much smoother. Throughout this book, I'm going to assume that you've made the changes below. It's up to you to keep things straight if you don't.

Package Folder Structure

The first thing that we're going to do is create some folders. First, create a folder called 'packages' on the root of your C:\ drive. This will be our repository for the packages that we'll be working on through out this book. Next, on the book CD is a folder called 'Package Template.' Copy the Package Template folder from the CD into the C:\Packages folder.

Macrovision recommends a similar directory structure. Take a look at the AdminStudio at the sample chapter (Preparing for Application Migration): http://www.installshield.com/downloads/isas/tm/AS55_SampleChapter.pdf

This folder structure is important to the process flow of repackaging. As you can tell the number folders reflect significant steps involved in the process. In addition, as you continue to repackage and deploy packages, this repository reflects the history of that process. If you're working with a team of developers and each developer is using the same folder structure anyone can quickly determine the status of a package.

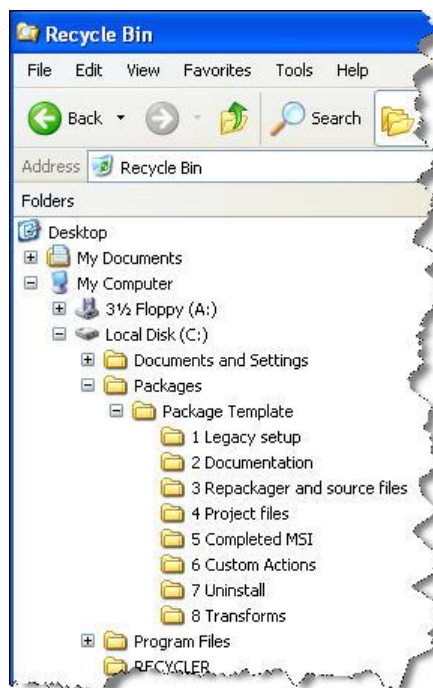


Figure 17. The package directory is the repository for all the work that we'll be doing with AdminStudio. For each package, there is a clearly defined sub-folder structure.

For now, don't worry too much about this folder structure. When we start a new project, we'll simply make a copy of Package Template and rename it after the application. A brief description of each folder follows below. Much of what these folders do will be explained as we move along in this book.

1 Legacy Setup

The Legacy Setup folder contains the original install image for the application. In most cases, we'll refer to this as the legacy image. When doing this we're making the assumption that this is a non-.msi package. Of course, we could be wrong but this is a good starting assumption.

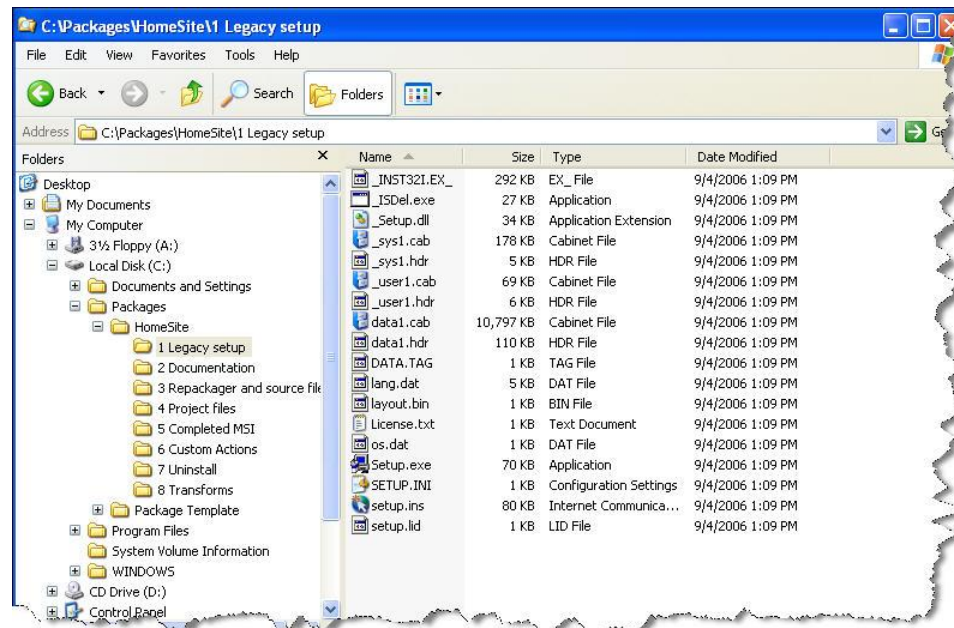


Figure 18. The '1 Legacy Setup' folder contains the original install image for the targeted package.

In some cases, the legacy image could be a single self-extracting executable. In others, it could be a copy of the CD image. This will be our starting point for whatever work needs to be done. Preserving the original install image ensures that we have a well-defined starting point.

2 Documentation

The Documentation folder is critical to the success of our repackaging effort. This folder should contain any documents, text files, emails, and comments about the application. This includes key information such as:

- Contact information (names, email addresses, phone numbers of the people knowledgeable about the application)
- Install related information (dialog selections, passwords, serial numbers, etc.)
- Testing information (how to verify the application is installed correctly)
- Corporate-specific information (anything specific to your company)

With this information you can see why this folder is so important. This information tells us how to run the legacy application. It tells us how to test the application once it's been repackaged. It also tells us who to contact in case things go wrong.

3 Repackager and Source Files

The Repackager and Source Files folder is used by the Repackager. This is where we'll store the results of the snapshot process. Normally we won't need to do much with anything in this folder.

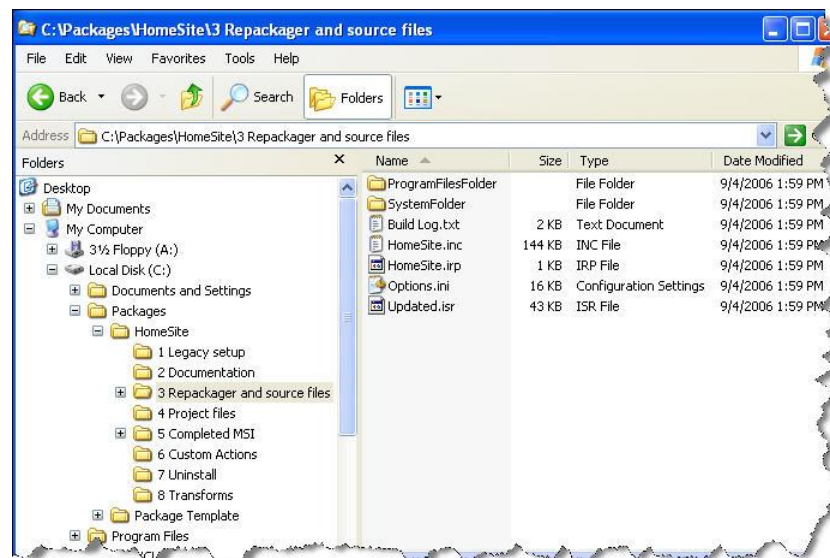


Figure 19. The '3 Repackager and source files' folder is used by the repack tool as a storage for the targeted application files.

Whenever we build from our Editor project file, the files in this folder are used to create the resultant .msi package.

4 Project Files

The Project Files folder contains the Editor project file. The Editor project file has a .ism extension which is registered to the FLEXnet Editor application which means we can double-click the project file to launch the Editor. The project file is created by the Repackager during the repackaging process.

This project file is the basis for all changes, edits, and any other modifications to the final .msi package. Normally during the refining process we'll return to this project file to make changes.

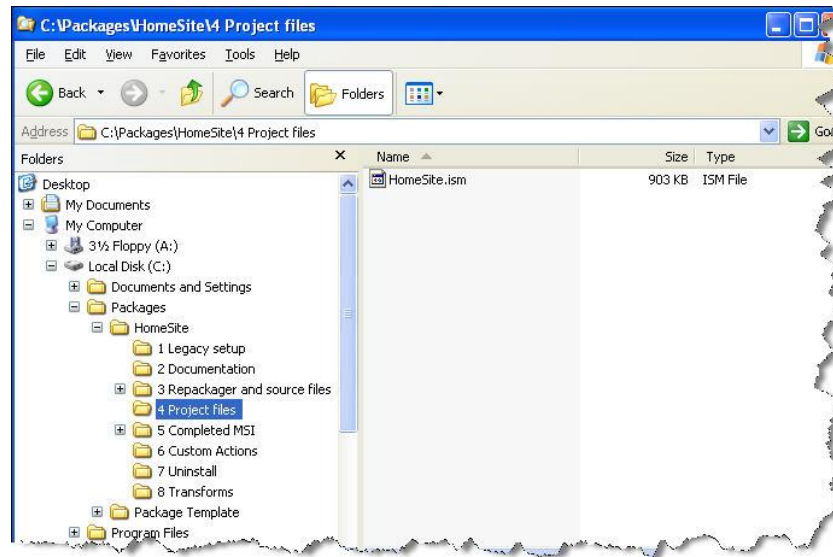


Figure 20. The '4 Project files' folder contains the AdminStudio Editor project file. This file is created by the Repackager from the contents of '3 Repackager and source files.'.

Once this file is created by the Repackager we'll rarely need to go back to either the Repackager or the original legacy image.

Keep in mind that if the original legacy image is already an .msi package, there will not be a project file. Under these circumstances, this folder remains empty.

5 Completed MSI

The Completed MSI folder contains the .msi package built by the project located in 4 Project Files. In most cases, building from the project file (.ism) replaces the contents of this folder.

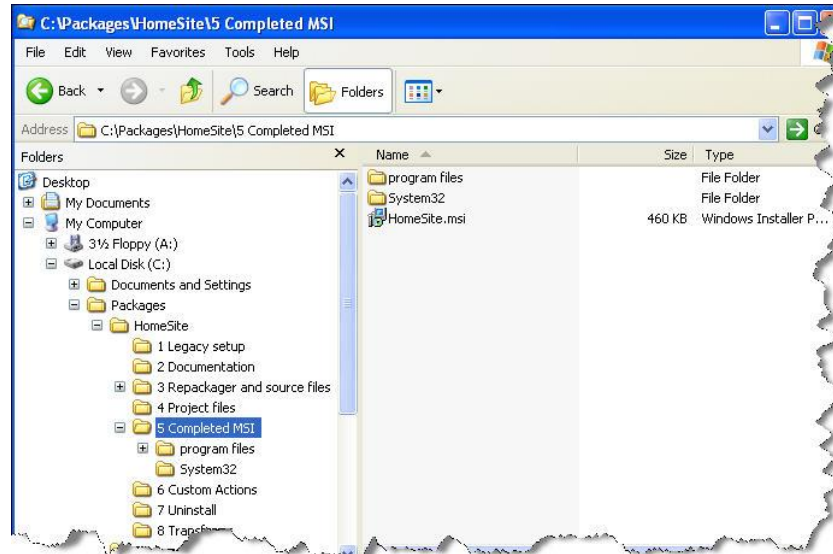


Figure 21. The '5 Completed MSI' folder contains the final Windows Installer image. From this folder that install image can be deployed.

After testing and resolving conflicts, this folder contains the .msi package that you'll deploy out your end-users.

6 Custom Actions

The Custom Actions folder contains all of the custom actions that are unique to your install. These custom actions are used by the project file located in 4 Project Files.

Keeping a separate folder for custom actions allows us to make application-specific changes to our custom actions without worrying about a rippling effect across multiple projects. In this case, each project has its own unique custom actions.

7 Uninstall

The Uninstall folder contains the testing results of our .msi package. Specifically, we'll store the results of the application uninstall test in this folder. Storing this information here allows us to review the changes left on the system after our application is uninstalled. In most cases, we'll create custom actions to clean-up the uninstall.

8 Transforms

The Transforms folder contains the transform projects for our .msi package. In most cases the transforms we'll create will be for vendor-provided .msi packages. That is if the application is already in an .msi format we can still make changes to the application through a transform.

Creating transforms within a sub-folder allows us to create unique transforms that are specific to an .msi package.

Windows Installer SDK Help File

The next thing we'll do is create a desktop shortcut. We'll be using the Windows Installer Help file throughout this book. It's a great reference to any and all things related to Windows Installer. To make things easy, let's create a link to this help file on the desktop. The file is called "msi.chm" and can be found at either of the following locations:

```
C:\Program Files\InstallShield\AdminStudio\7.5\
Editor\Program\0409
```

Or

```
C:\Program Files\Common Files\InstallShield\
AdminStudio\Help\0409
```

Right-click the file and drag it to the desktop. Drop the file and select "Create Shortcuts Here" to create the shortcut. You may want to rename the shortcut "Windows Installer Help" to make it easier to identify. We'll come back to this shortcut later.

AdminStudio Settings

Finally, there are two things that we need to change within the AdminStudio Editor. We'll talk about these later, but for now it helps to make these changes along with everything else. Launch the AdminStudio Editor through Start→All Programs→Macrovision→FLEXnet AdminStudio 7.5 Tools→Editor.

From the application menu, select Tools→Options. This will bring up a multi-tabbed dialog. On the Merge Modules tab is an edit box labeled “Merge Module Locations (Current User).” This edit box contains a bunch of comma-delimited paths. We want to get rid of the first path and move the last path to the beginning.

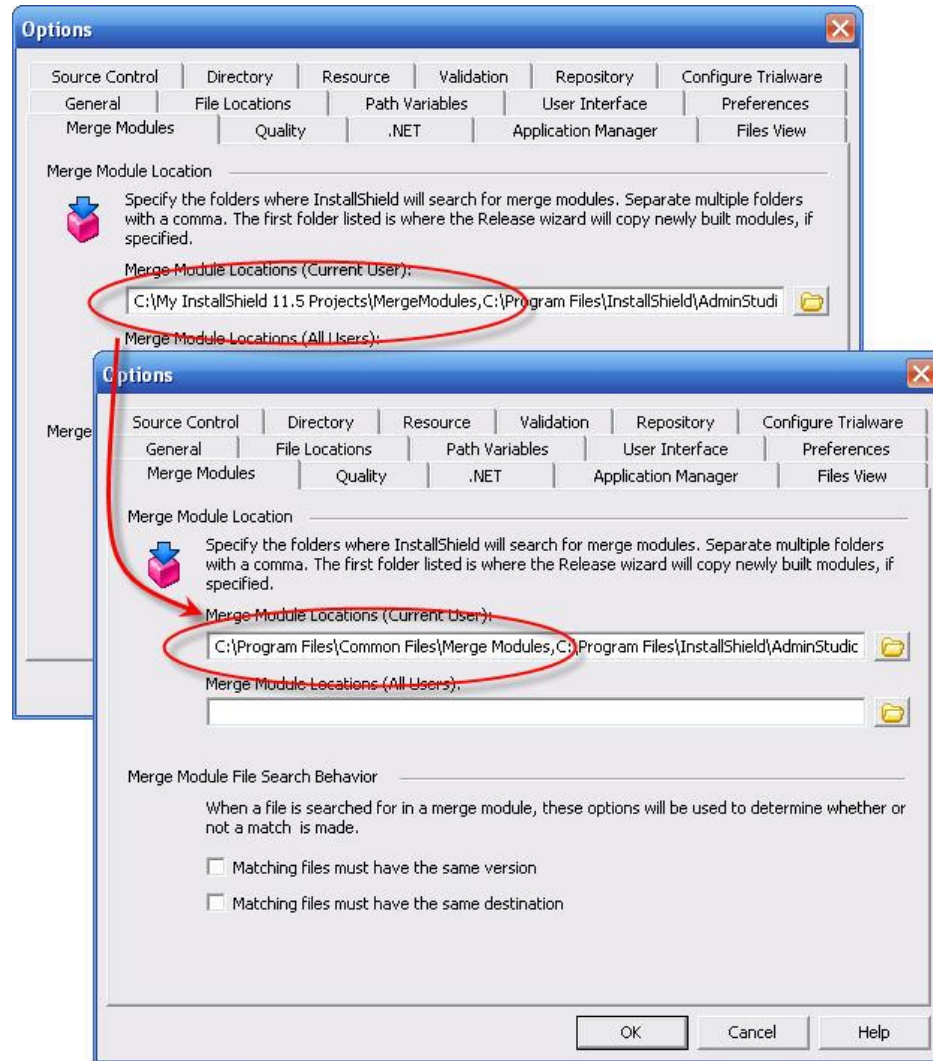


Figure 22. To make things a little bit easier within the AdminStudio Editor environment, let's change the default location of merge modules. We'll use the common path the AdminStudio uses.

The first path should read:

```
C:\My InstallShield 11.5 Projects\MergeModules
```

Delete that entry.

The last path should read:

```
C:\Program Files\Common Files\Merge Modules
```

Cut this entry and paste it to the beginning—include a trailing comma. Afterwards, make sure you remove the trailing comma at the very end of this line. We'll talk about merge modules and what this change means later in the book. Until then, trust me. I know what I'm doing.

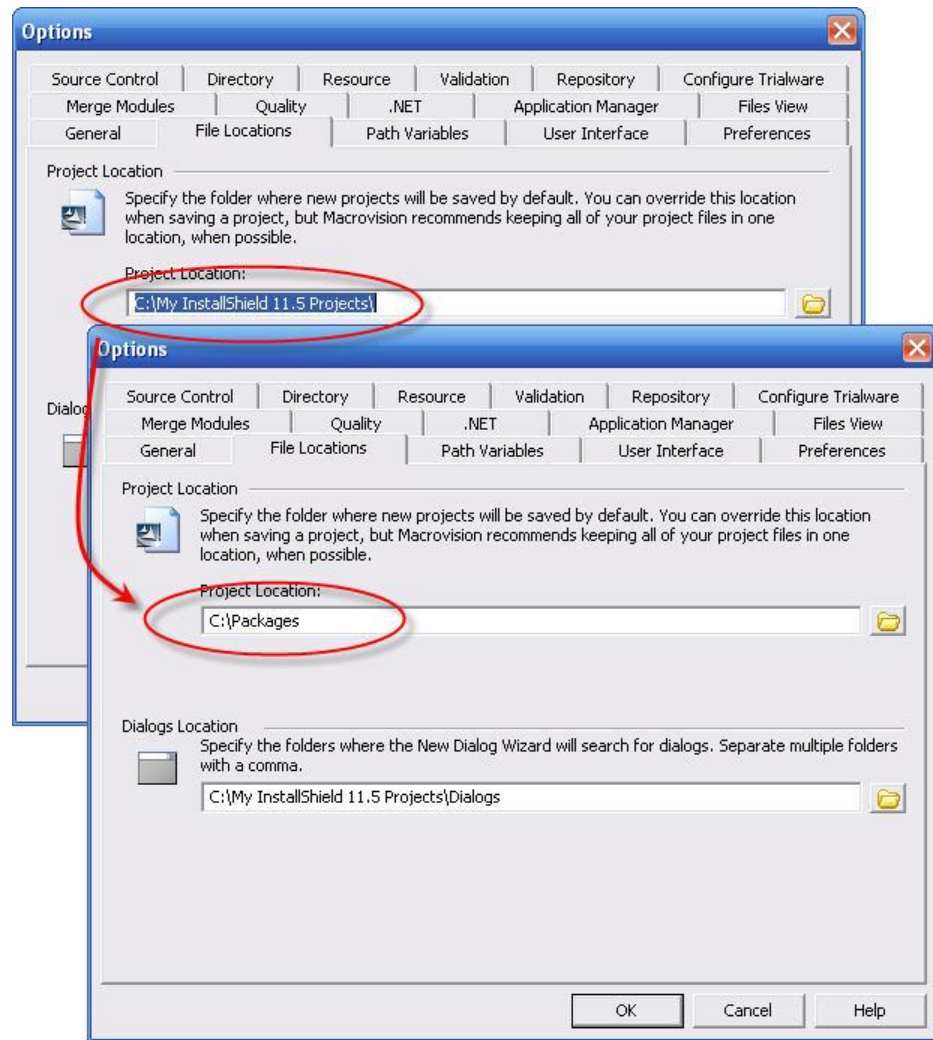


Figure 23. The default project location is another destination that we'll change. In our case, all of our projects will be within the C:\Packages folder.

Next, select the "File Locations" tab. The first edit box is the default location for any Editor project files. Since we'll be repackaging from within our C:\Packages folder, it makes sense to use this path as our default. So delete the current entry (C:\My InstallShield 11.5 Projects\) and replace it with C:\Packages.

If the C:\My InstallShield 11.5 Projects\ folder already exists and is empty, feel free to delete it.

Installing VMWare

Another piece of software that you're absolutely going to need is VMWare Workstation. This is an operating system virtualization software package that allows us to repackage and test in a sterile environment. With this software we can physically separate our development process from running and testing the repackaged application.

There are other software applications that do the same thing. Microsoft Virtual PC is one such application. Feel free to use that instead of VMWare Workstation if you prefer. Use the directions below to set-up a similar environment.

Be aware that there are other VMWare products available. None are as focused or ideal as VMWare Workstation for what we're going to do. However with that said feel free to experiment on your own. My suggestion is to use VMWare Workstation initially just to make sure that you're comfortable with the process. Once you have the process down, you may be in a better position to make changes.

Once you've got the VMWare software, the install process is not painful at all. In fact, we'll go with the default values for all of the install options.

VMWare Workstation can be downloaded and evaluated for 30 days. This should be more than enough time to get you through this book. It's fairly inexpensive and is easily worth the cost. For more information go to the VMWare website at:
<http://www.vmware.com/products/ws/>



Figure 24. Once you start the VMWare installation, select Next on the Welcome dialog.

VMWare takes a little less than 100 Mbytes to install. Additional space will be required for each VMImage that we create. Each image consumes anywhere from 4 Gbytes and up. For multiple VMImages you'll want to make sure that you have plenty of free disk space.

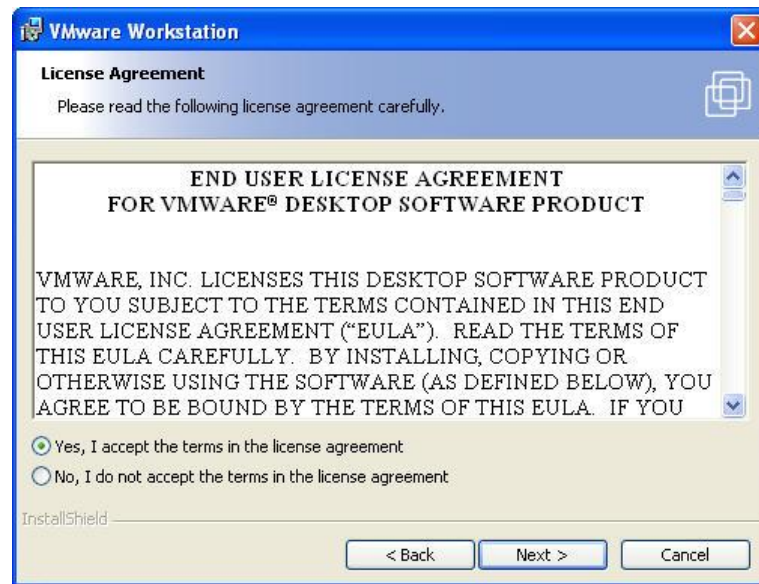


Figure 25. On the license agreement dialog, you'll need to select the Yes radio button to continue..

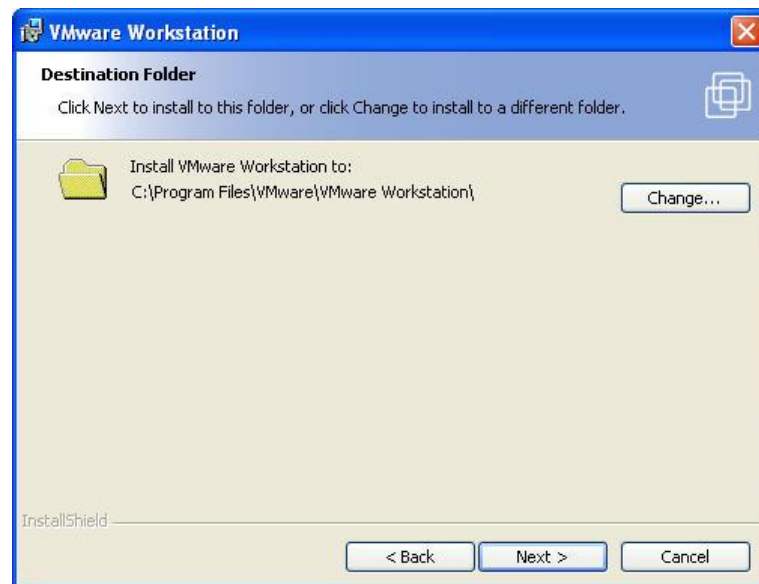


Figure 26. Choose the default destination for the VMWare application. The default is usually a good place to start. Select Next to continue.

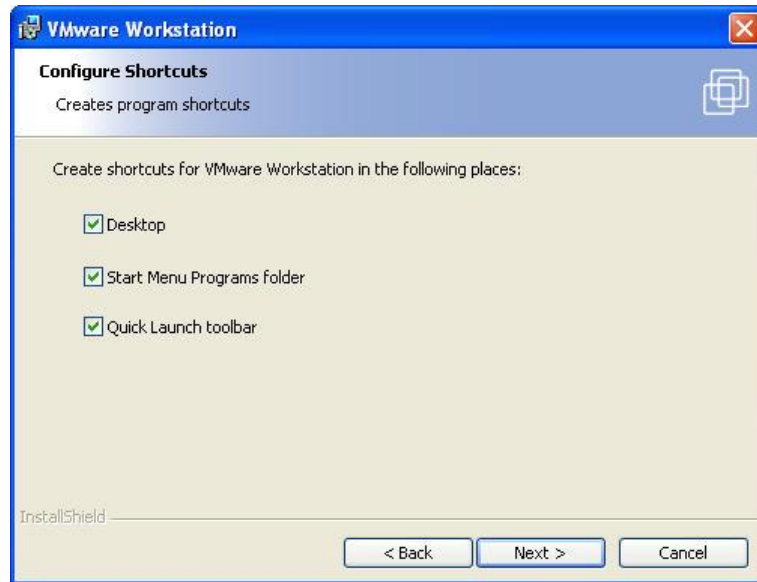


Figure 27. There are a variety of shortcut options. Make sure the shortcuts that you want are checked and select Next to continue.

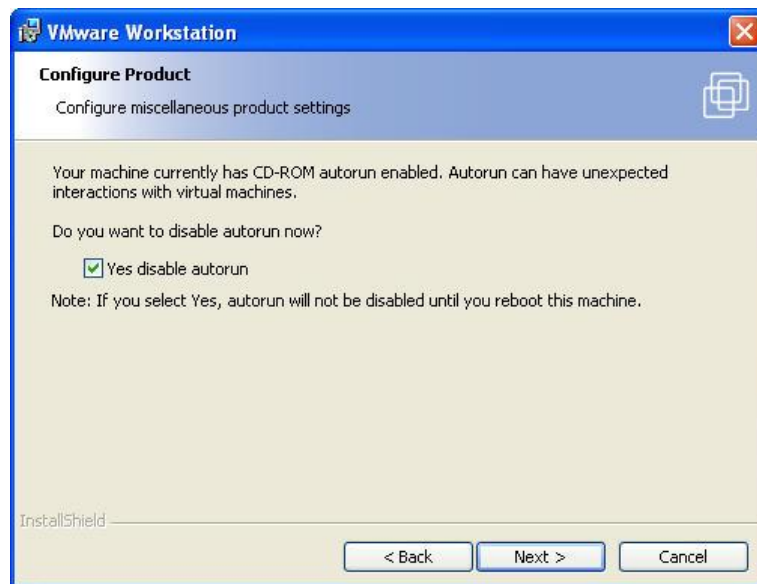


Figure 28. At this point you can disable the CD Autorun support within the VMImage. Feel free to make any changes and select Next to continue.

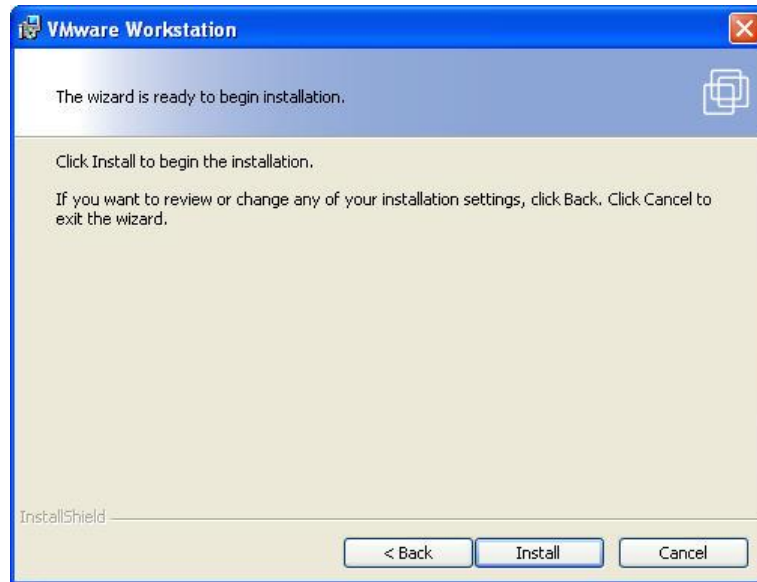


Figure 29. At this point the VMWare install is ready to install and configure the application. Select Next to continue.

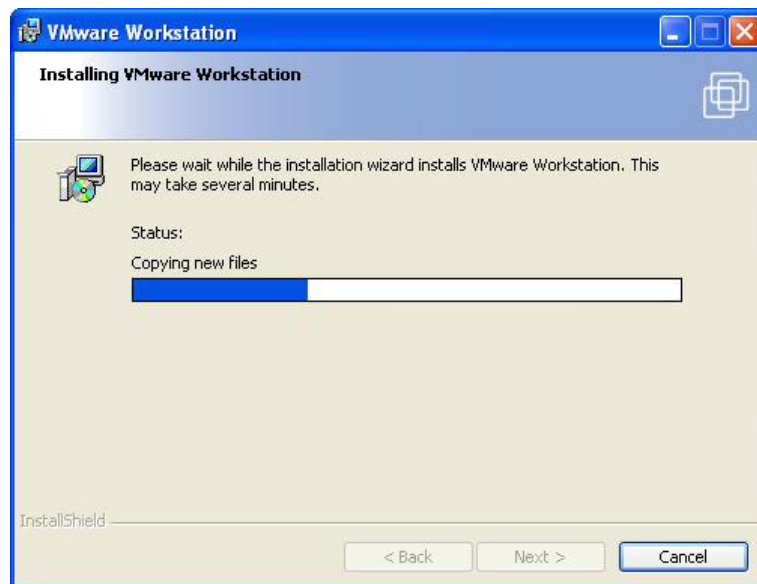


Figure 30. The file transfer and configuration of the application shouldn't take more than five minutes. Again, your results may vary depending on system configuration.

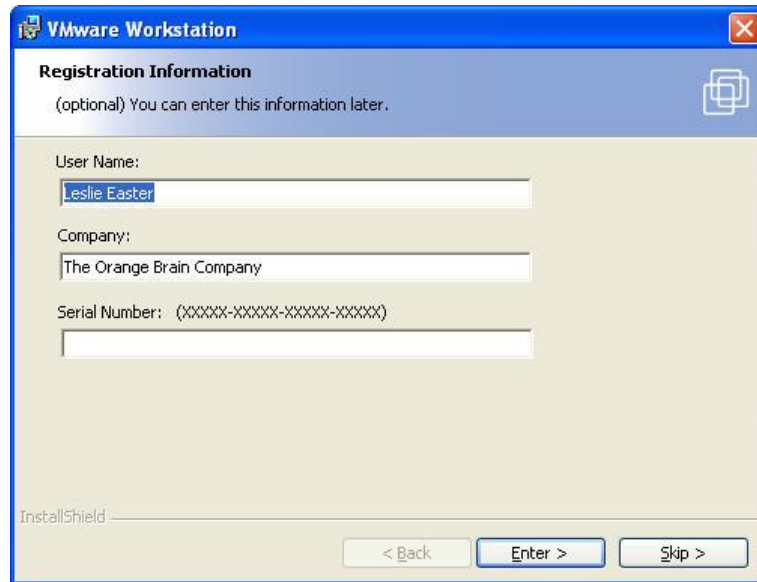


Figure 31. If you have a serial number, enter it here. Selecting 'Skip' will enable a 30-day evaluation.



Figure 32. At last the VMWare installation is finished. Select Finish to exit. You may be required to reboot your system afterwards.

Creating a VMImage

Once VMWare is installed the next thing we'll need to do is create a VMImage. For this we'll need the original operating system install media. My recommendation is to use the operating system most likely to match your end-users. Once you've decided on the operating system to use, we'll need to install that operating system into VMWare.



Figure 33. The process of creating a VMImage is easily initiated through the Virtual Machine Wizard.

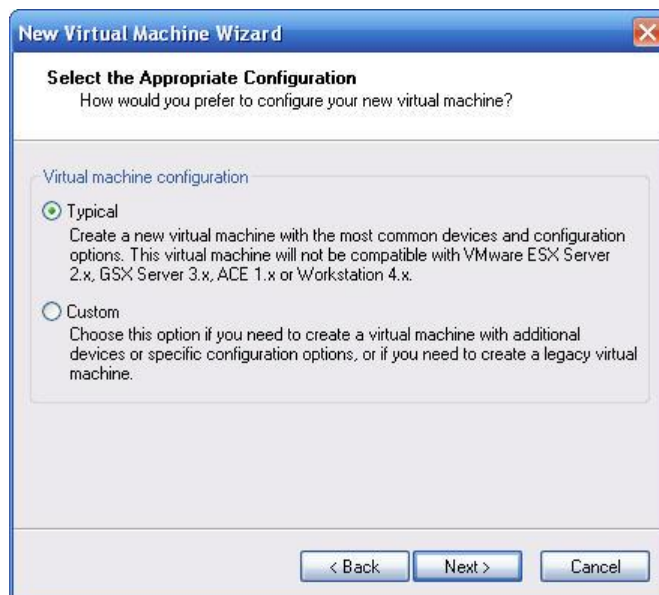


Figure 34. The first time through, you may want to go with the Typical configuration. If you have unusual hardware or software requirements use the Custom option.



Figure 35. In order to prepare the VMImage requirements, select the operating system and version from this dialog.

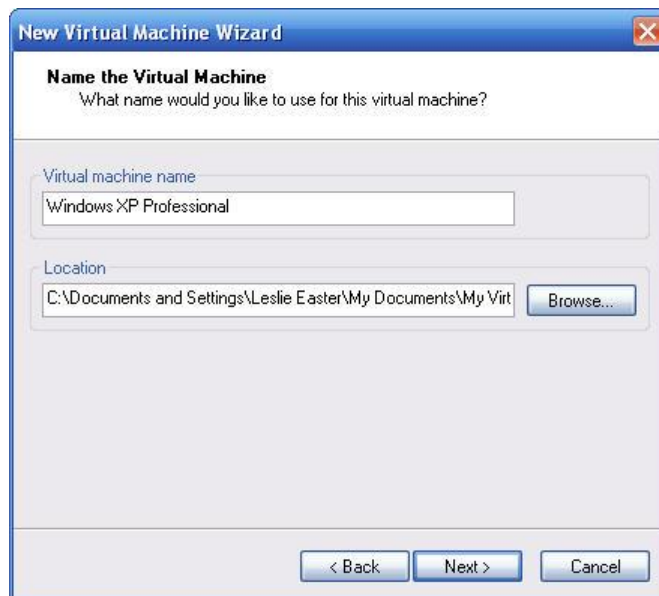


Figure 36. On this dialog, we'll give our virtual machine a name and identify a physical location on our development system to store the VMImage.



Figure 37. The Network Type option may take some experimentation. In most cases, a bridged network option works best. Again, in your environment a different option may be required.

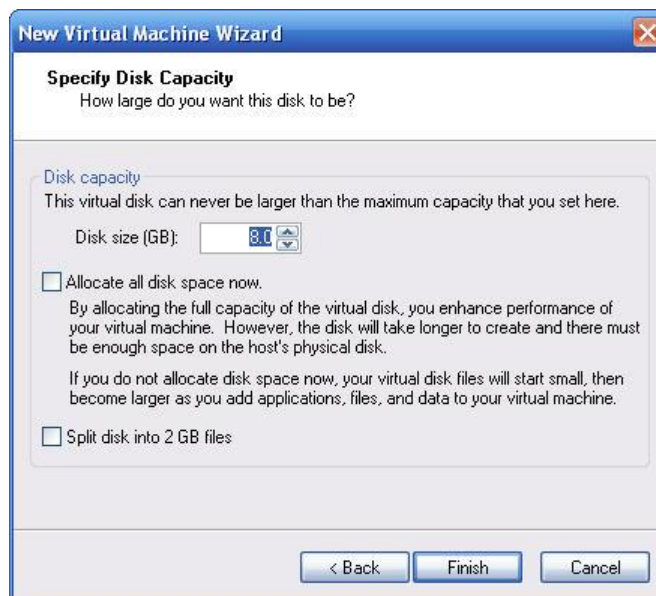


Figure 38. Unfortunately the VMDisk cannot grow past the amount of disk space that is allocated at this point.

Multiple operating systems can be installed into VMWare. Each new operating system will create a new tab entry within VMWare. It's not uncommon to see developers with a half dozen or so different VMDisk. In fact, multiple VMDisks can be started at the same time. Doing this requires a significant amount of system memory. This is particular handy if you want to start one image up as a server and another as a client.

Using VMWare

After you install the target operating system into VMWare, the new operating system can be started by selecting the green “Play” button on the icon toolbar. Making this selection starts the selected operating system.



Figure 39. You'll be very familiar with VMWare toolbar. From here we can revert, start, or capture snapshots.

If there is no operating system tab, use File→Open Virtual Machine to browse to the VMImage directory and select the .vmx file.

Once the operating system is started, it behaves just like a separate operating system—even though it's running in the context of your very much real hardware! Once your done playing with the VMImage it can be shut down by simply selecting the red “Stop” button or by closing the VMWare application. You can also shut down the operating system directly (Start→Turn Off Computer).

The most common use of a VMImage is to save the current state of the operating system. This is done by preparing the operating system the way that you want and selecting the Snapshot button. Depending on the version of VMWare that you're using this will either overwrite the original image or, in the newer versions, preserve a new state.

To return to the saved state select VM→Snapshot→Revert or select the Revert button from the icon toolbar.

Setting up VMWare Work Space

Before we can begin the fun of using AdminStudio, there are still a few things that we need to do. By default the development system and our VM system behave as two separate systems. We'll need to make a connection between the two to save any work that is done within the VMImage. Remember the goal is to keep the VMImage as clean and as untampered with as possible.

In order to set-up the sharing, we'll need to create a couple of VM Shared Folders. This is done by launching VMWare and starting up our VMImage. Once the VMImage is up and running, use VM→Settings to bring up the VMWare tabbed dialog.



Figure 40. The first of two VMWare Shared Folders is the Packages folder. This creates a map between our VMImage and our development system.

Select the Options tab and select Shared Folders from the pane on the left. On the lower right is the “Add...” button. Select this button to start the Add Shared Folder Wizard. Select Next from the Welcome dialog. On the “Name the Shared Folder” dialog, type in “Packages” in the first edit field and Browse to the C:\Packages path in the Host folder edit field. This creates a mapping between the path on our development machine and the VMImage.



Figure 41. The second VM Shared Folder is to the Repackager folder. From this folder we can launch several of the AdminStudio tools within our VMImage.

Once you've created the Packages shared folder, do the same for the Repackager folder. This folder will map to the Repackager folder on our development machine. The path is:

```
C:\Program Files\InstallShield\AdminStudio\
7.5\Repackager
```

Next we'll need to make these shared folders easy to access. To do this we'll use a mapped drive. Launch Windows Explorer within the VMImage and select Tools→Map Network Drive.... To keep things consistent, let's map the R:\ drive to our shared folders. From the Folder control, select Browse. Within the Mapped Drive dialog map the R: drive to \\.\host\Shared Folders. This will show up within the VMWare Network of Network Places.

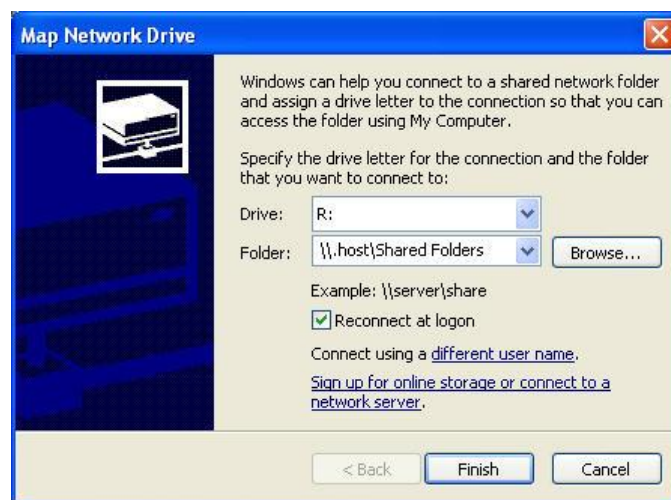


Figure 42. To finalize our work with the VM Shared Folders we'll need to create a mapped drive.

Once the mapped drive is created, let's make life easy for ourselves by creating a few desktop shortcuts. We'll be using this VMImage extensively for repackaging and testing. For repackaging, we'll be using the AdminStudio Repack tool. For testing, we'll be using several Windows Installer service command line values. Since we'll be doing a lot of each, creating these shortcuts makes sense.

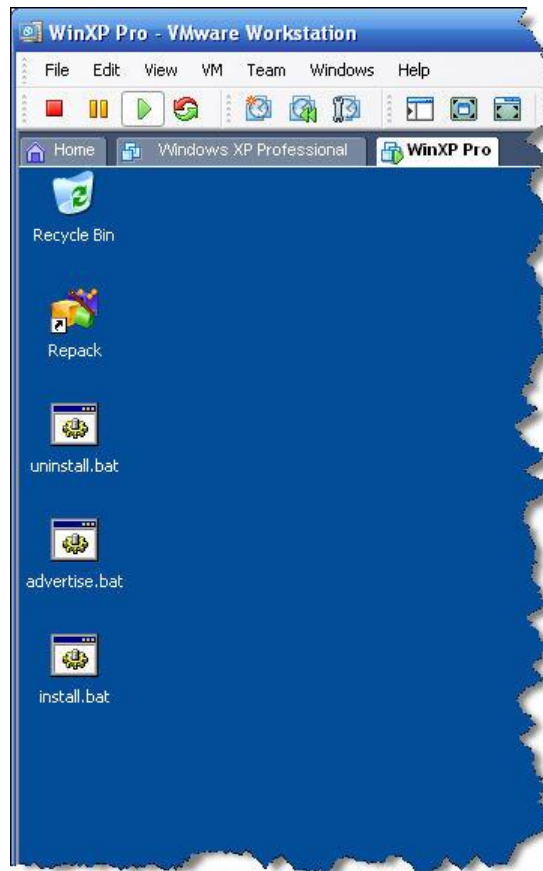


Figure 43. The final view of our VMImage should contain four new shortcuts: Repack, uninstall.bat, advertise.bat, and install.bat.

The first shortcut that we'll create is to the Repack utility. This utility is found in the R:\Repackager folder and is called "repack.exe." Go to the R:\ drive, open the Repackager folder, right-click on repack.exe and drag to the VMImage desktop. When you release the right-click, select "Create Shortcuts Here." Let's give this shortcut the name: "Repack."

The next shortcuts are connected to a few batch files. These batch files are located on the course CD in the "VMImage Shortcuts" folder. You should see the following files:

- Install.bat,
- Advertise.bat, and
- Uninstall.bat.

Each of these batch files contains the common command line parameters that we'll be using to test our .msi packages. We'll go into more of this a little bit later. For now, let's drag-and-drop these batch files onto the Desktop of the VMImage.

The last thing we need to do is check the Internet connection. Launch Internet Explorer and see if you can get to the Internet. If you can then everything is good. If you can't we may need to make a change or two.

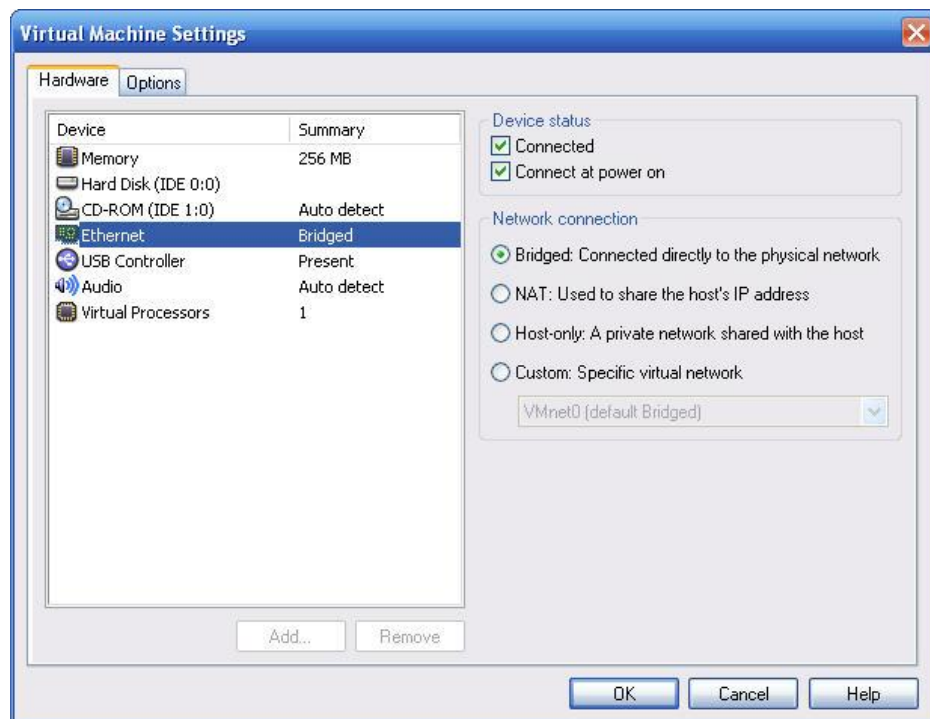


Figure 44. The VMImage Ethernet setting depends on your development system. You may need to experiment with this setting, but Bridged is a good place to start. Odds are good that you'll need an Internet connection within your VMImage.

In most situations, changing how the VMImage network is configured will do the trick. On the application menu, go to VM→Settings. From the tabbed dialog make sure the Hardware tab is selected. Select the “NIC 1” or “Internet” option—which one is available depends on your version of VMWare. When you make this selection, the network options are shown to the right. I would suggest selecting the “Bridged” connection. This one seems to be the most reliable on the majority of systems.

After you make this selection, you may need to restart the operating system. Alternatively you can use “ipconfig /release” followed by “ipconfig /renew” on the command line. If this selection doesn't work you may need to play with the options a bit more. If all else fails, track down a networking specialist to help you out.

Once everything is working, we'll need to save our work. Within VMWare this is done by taking a snapshot of the operating system. Taking a snapshot is easy enough—simply select the Snapshot toolbar button. You may be prompted to overwrite the existing snapshot (older versions of VMWare) or give the snapshot a name (newer versions of VMWare). I would name this one “Clean” because that's exactly what it is.

To test our work thus far, delete the Repack, install.bat, advertise.bat, and uninstall.bat shortcuts. Then select the “Revert” button from the Icon toolbar. You should see that when the revert process is complete our shortcuts are right back where they belong. We’ve reverted back to the initial state of our saved operating system image.

Final Thoughts on Getting Started

We’ll talk about many of these changes as the book progresses. Once you understand the repackaging process and get the feel for how AdminStudio works, you may want to come back and re-visit these settings. While they are good defaults, your corporate environment and personal preferences may yield changes.