Chapter 2. Setting Up Your Environment

In this chapter, we will cover setting up the basic environment:

- Hosting OS configuration
- Installing the Workbench
- Discussing VM creation principles
- Discussing general customizations of the share

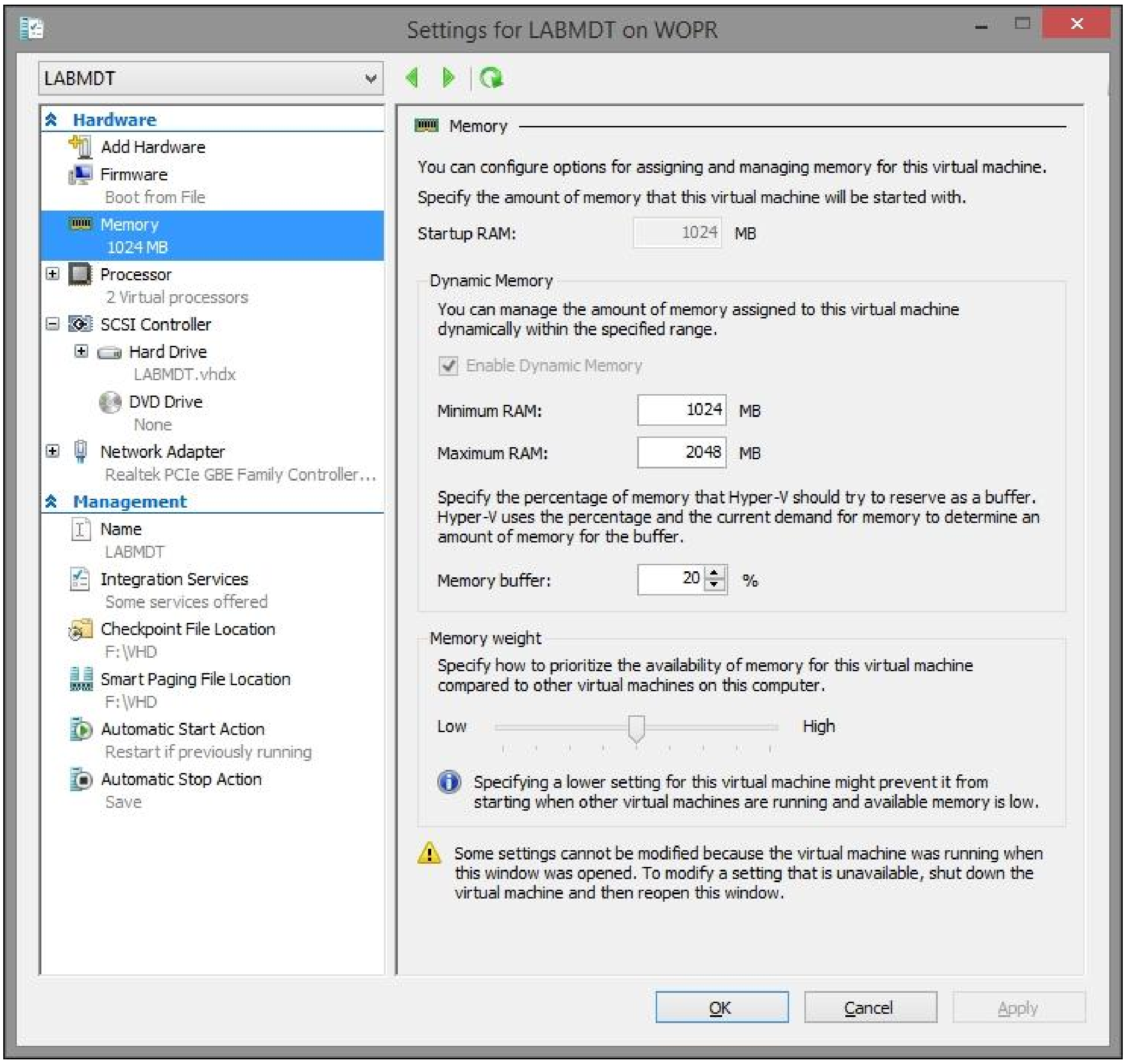
Setting up MDT for the first time

To implement **Microsoft Deployment Toolkit** (**MDT**), one must have a properly provisioned system on which we can lay down the **Windows Assessment and Deployment Kit** (**Windows ADK**) and MDT install files. It must have space, CPUs, some amount of RAM, and most importantly, quick network links to other systems.

All the particulars around these requirements of course depend on what you are using MDT for. Are you building a golden image with which you will deploy via another deployment system? Are you implementing a deployment scenario where you need to deploy to laptops, desktops, medical imaging systems, and point-of-sale devices at multiple sites around the world, all from one infrastructure?

Setting up the virtual machine

So, for the sake of moving things along, we'll start small—with nothing. In this chapter, we'll get started with a single system. In order to illustrate, we will be using Windows Server 2012 R2, so transfers will be faster over the new **Server Message Block** (**SMB**) improvements. It's a virtual machine that is configured as shown in the following screenshot:


Settings for LABMDT on WOPR

LABMDT

Hardware
- Add Hardware
- Firmware
  - Boot from File
- Memory
  - 1024 MB
- Processor
  - 2 Virtual processors
- SCSI Controller
  - Hard Drive
    - LABMDT.vhdx
  - DVD Drive
    - None
- Network Adapter
  - Realtek PCIe GBE Family Controller...

Management
- Name
  - LABMDT
- Integration Services
  - Some services offered
- Checkpoint File Location
  - F:\VHD
- Smart Paging File Location
  - F:\VHD
- Automatic Start Action
  - Restart if previously running
- Automatic Stop Action
  - Save

Memory

You can configure options for assigning and managing memory for this virtual machine.

Specify the amount of memory that this virtual machine will be started with.

Startup RAM: 1024 MB

Dynamic Memory

You can manage the amount of memory assigned to this virtual machine dynamically within the specified range.

[✓] Enable Dynamic Memory

Minimum RAM: 1024 MB

Maximum RAM: 2048 MB

Specify the percentage of memory that Hyper-V should try to reserve as a buffer. Hyper-V uses the percentage and the current demand for memory to determine an amount of memory for the buffer.

Memory buffer: 20 %

Memory weight

Specify how to prioritize the availability of memory for this virtual machine compared to other virtual machines on this computer.

Low — High

Specifying a lower setting for this virtual machine might prevent it from starting when other virtual machines are running and available memory is low.

Some settings cannot be modified because the virtual machine was running when this window was opened. To modify a setting that is unavailable, shut down the virtual machine and then reopen this window.

OK Cancel Apply

For Windows 8(.1) and Windows 10 guest

The specification settings for the virtual machine are described as follows:

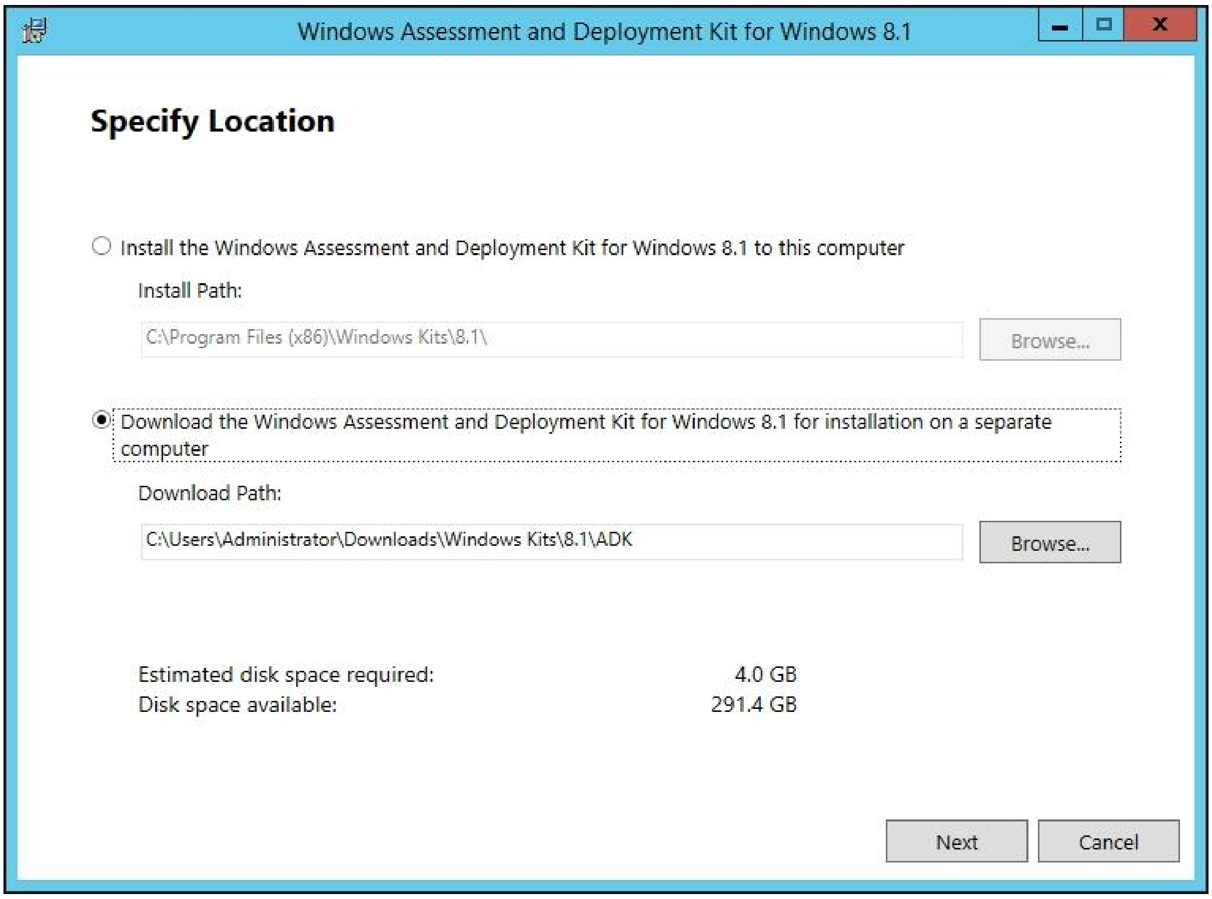
- 1,024 MB of RAM as a floor, autoscaling to 2,048 MB as the OS demands it.
- 2 core—as that's my preference in a virtual machine guest for anything I want to be fairly productive from a performance perspective.
- The only real consideration for an MDT installation is disk space. I've allotted 300 GB in the example, but this number is primarily up to the deployment engineer. The number of images and applications will generally drive the need for space. Using a disk system that allows for easy growth (SAN or NAS environment) or even a RAID array of local disks would work well.

Downloading the MDT installer

I recommend, for the purpose of this book, downloading the MDT installer for 2013. As of this printing, it is located here at https://www.microsoft.com/en-us/download/details.aspx?id=50407 .

Also, you will need the Windows ADK, which is located at http://www.microsoft.com/en-us/download/confirmation.aspx?id=39982 .

The ADK comes as a web installer, `adksetup.exe`, by the way. For serious work, download the complete ISO by first downloading `adksetup.exe` and then selecting the **Download the Windows Assessment and Deployment Kit** option:


Windows Assessment and Deployment Kit for Windows 8.1

Specify Location

( ) Install the Windows Assessment and Deployment Kit for Windows 8.1 to this computer

Install Path:

C:\Program Files (x86)\Windows Kits\8.1\

(•) Download the Windows Assessment and Deployment Kit for Windows 8.1 for installation on a separate computer

Download Path:

C:\Users\Administrator\Downloads\Windows Kits\8.1\ADK

Estimated disk space required:	4.0 GB
Disk space available:	291.4 GB

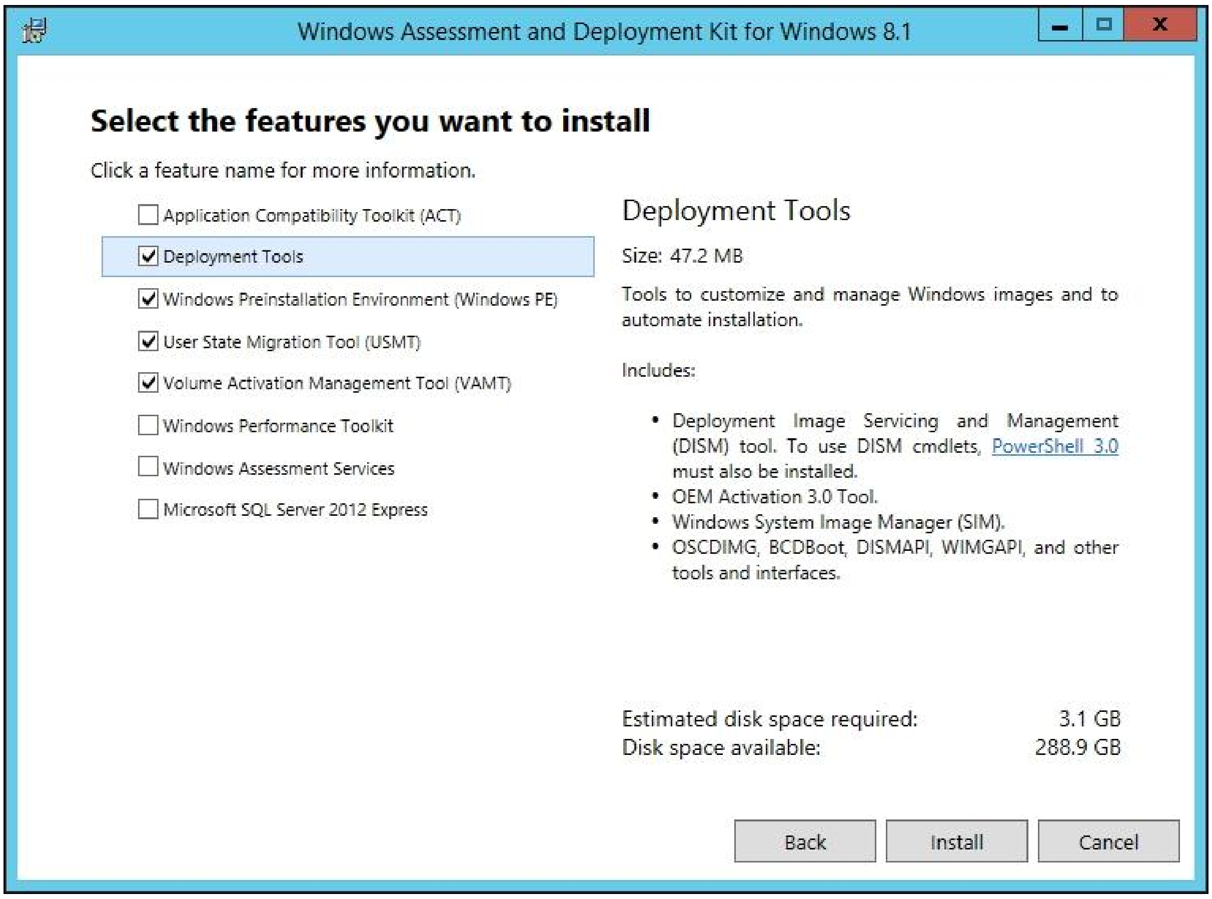
This option stores all the ADK setup files so that the following installations won't require a potentially lengthy download and can even be done on machines that don't have the Internet access going forward.

Installing Windows ADK

Once you have the files, simply run `adksetup.exe`. For the purpose of this work, you'll want to tick the following boxes in the installer:

- **Deployment Tools**
- **Windows Preinstallation Environment (Windows PE)**

- **User State Migration Tool (USMT)**
- **Volume Activation Management Tool (VAMT)**



Select the features you want to install

Click a feature name for more information.

- ☐ Application Compatibility Toolkit (ACT)
- ☑ Deployment Tools
- ☑ Windows Preinstallation Environment (Windows PE)
- ☑ User State Migration Tool (USMT)
- ☑ Volume Activation Management Tool (VAMT)
- ☐ Windows Performance Toolkit
- ☐ Windows Assessment Services
- ☐ Microsoft SQL Server 2012 Express

Deployment Tools

Size: 47.2 MB

Tools to customize and manage Windows images and to automate installation.

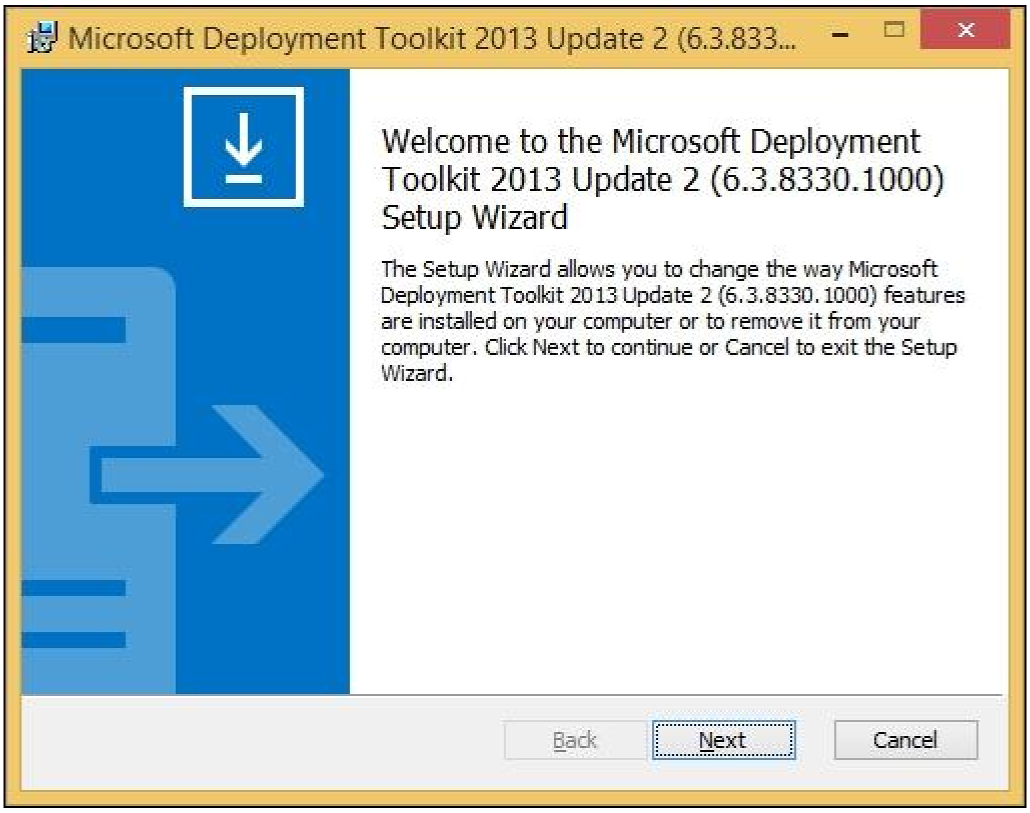
Includes:

- Deployment Image Servicing and Management (DISM) tool. To use DISM cmdlets, PowerShell 3.0 must also be installed.
- OEM Activation 3.0 Tool.
- Windows System Image Manager (SIM).
- OSCDIMG, BCDBoot, DISMAPI, WIMGAPI, and other tools and interfaces.

Estimated disk space required:	3.1 GB
Disk space available:	288.9 GB

Installing MDT

After the ADK for Windows 8.1 is installed, we can get started with installing the actual MDT. As my virtual machine is Windows Server 2012 R2, it is a 64-bit machine, so I will download and install the `MicrosoftDeploymentToolkit2013_x64.msi` MDT 2013 and run it:


Microsoft Deployment Toolkit 2013 Update 2 (6.3.833...

Welcome to the Microsoft Deployment Toolkit 2013 Update 2 (6.3.8330.1000) Setup Wizard

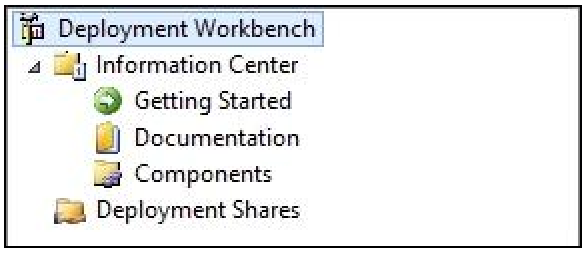
The Setup Wizard allows you to change the way Microsoft Deployment Toolkit 2013 Update 2 (6.3.8330.1000) features are installed on your computer or to remove it from your computer. Click Next to continue or Cancel to exit the Setup Wizard.

So, we're installing the **6.2.5019.0** version of MDT 2013. Install it to `C:\`, along with the ADK as done previously, and then we are ready to set up the reference share and the deployment share.

Setting up reference share and deployment share

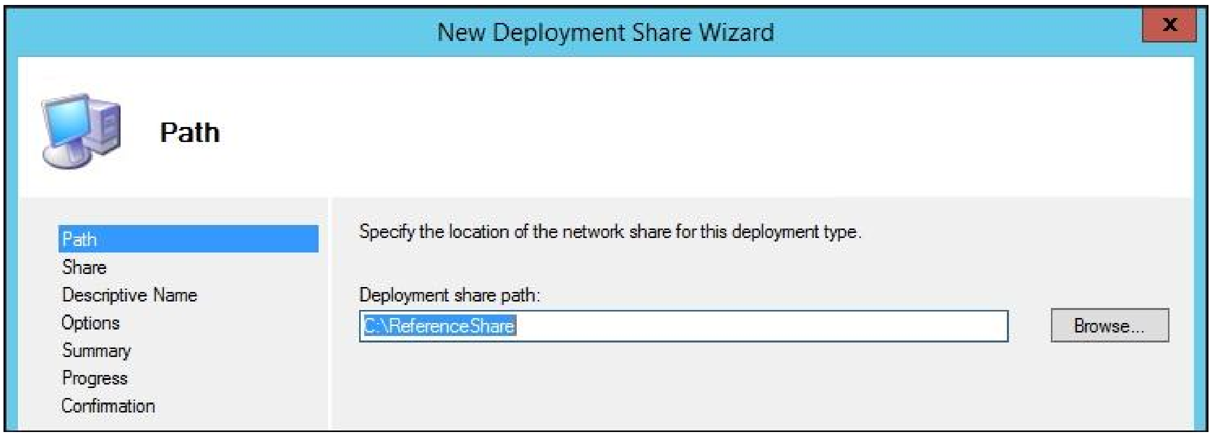
Let's launch the MDT:

We are presented with a console that should look similar to a typical **Microsoft Management Console** (**MMC**):



Feel free to poke at the console. The primary area we are concerned with is the **Deployment Shares** line, which we will select with the mouse, and then right-click to select **New Deployment Share**. Make sure to change the **Deployment share path** to reflect this as a reference share, not a deployment share.

In the next window, as shown in the following image, the wizard puts up a window to create the deployment share. In this case, we will name it `ReferenceShare` and place it on our `C:` drive, which has plenty of space allocated in the virtual machine:


New Deployment Share Wizard

Path

Specify the location of the network share for this deployment type.

Deployment share path: C:\ReferenceShare

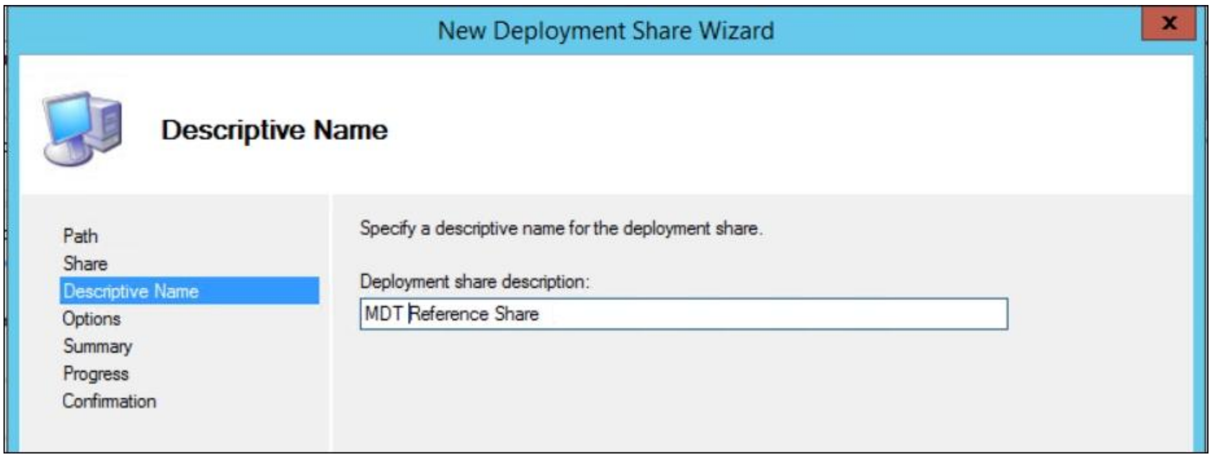
Now, keep in mind, this can be pretty much anywhere. It's a filesystem-based directory/share here. We aren't doing complex DB work like Microsoft Exchange; we are creating a file share at this point, that's it. If you have a volume for this, fine, you can put it here. No problem. Antivirus can cause some issues while hitting the **Next** button though, especially, if the filter driver is set to prevent the creation of `autorun.inf` files.

Specifying a share name

The **Share** screen is to set the actual share for the reference share that we will use to create the reference image.

Specifying a descriptive name

On the **Descriptive Name** screen, specify the reference rather than deployment in the **Deployment share description** box:


New Deployment Share Wizard

Descriptive Name

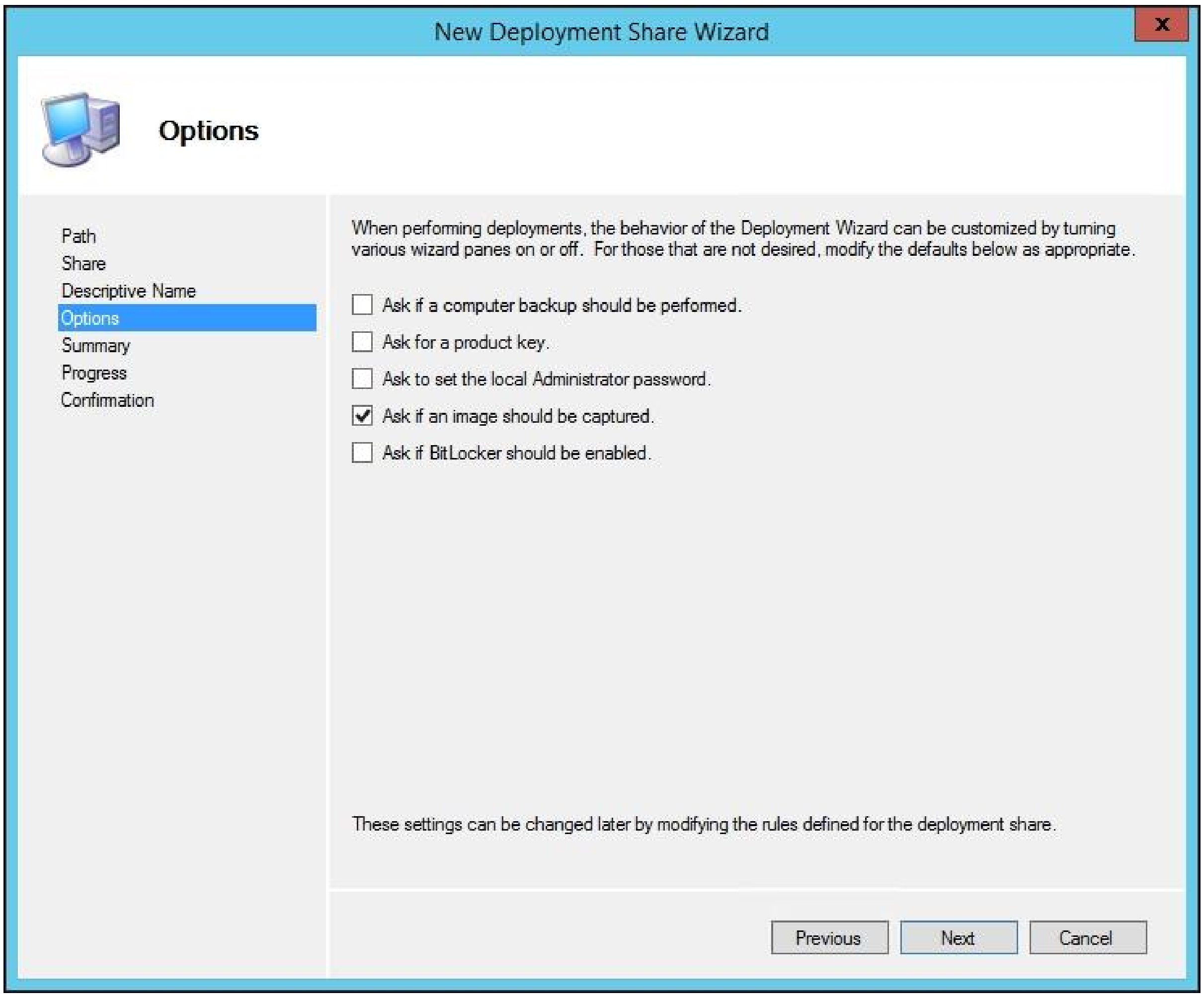
Specify a descriptive name for the deployment share.

Deployment share description: MDT Reference Share

Modifying deployment options

We are going to utilize this reference share as a workshop of sorts; it isn't going to be utilized in a traditional deployment sense. As such, we don't need it to do a lot of default actions that are available on the **Options** screen, except **Ask if an image should be captured**.

What the following checkboxes are doing is setting properties in the `CustomSettings.ini` file, also known as the `Rules` section of the share:



When performing deployments, the behavior of the Deployment Wizard can be customized by turning various wizard panes on or off. For those that are not desired, modify the defaults below as appropriate.

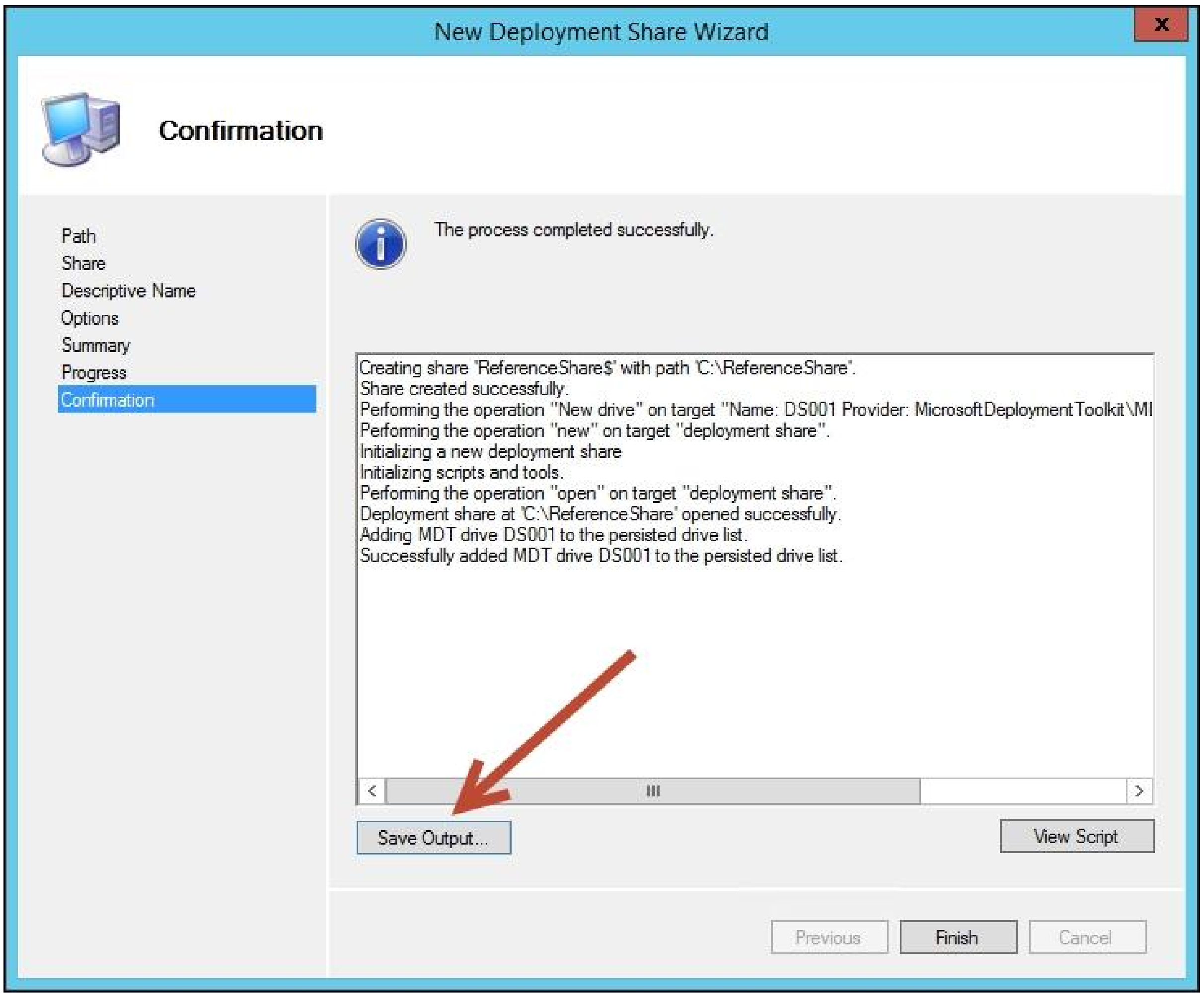
- [ ] Ask if a computer backup should be performed.
- [ ] Ask for a product key.
- [ ] Ask to set the local Administrator password.
- [✓] Ask if an image should be captured.
- [ ] Ask if BitLocker should be enabled.

These settings can be changed later by modifying the rules defined for the deployment share.

Summary and confirmation

The Summary screen shows the options that have been previously selected for confirmation.

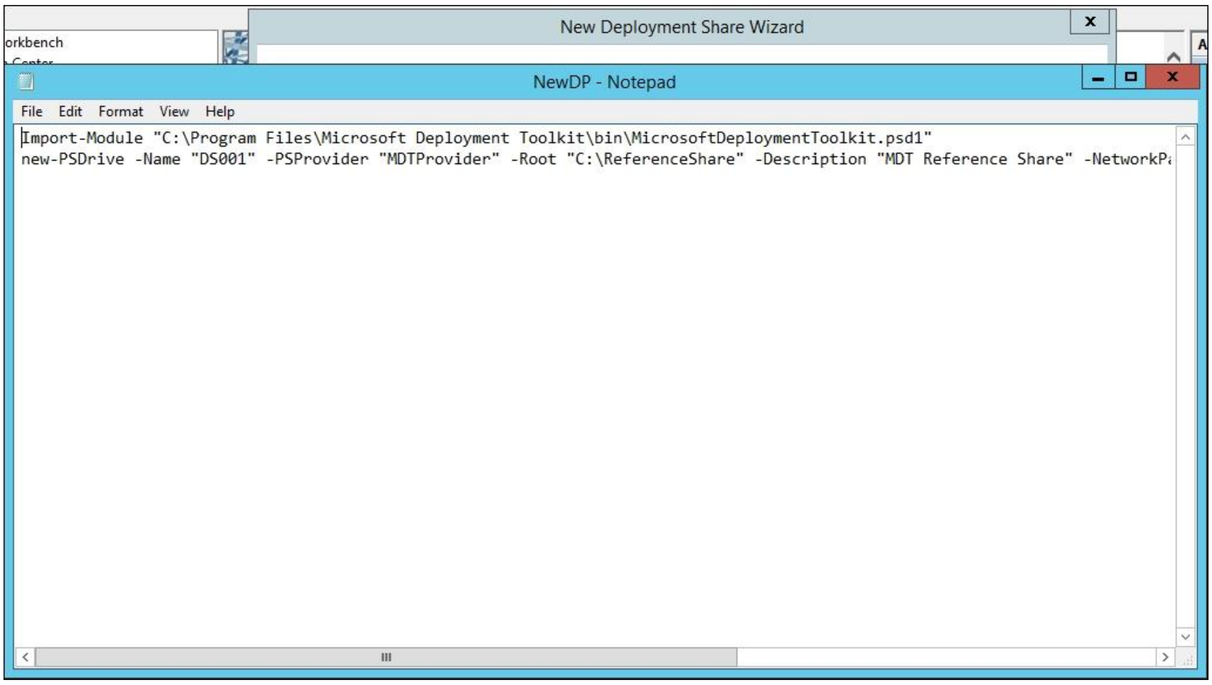
After you click on **Next**, you are provided with a **Confirmation** screen that shows a few interesting options. The first is the **Save Output...** box, which will save the output of what was just done for logging purposes:



```
Creating share 'ReferenceShare$' with path 'C:\ReferenceShare'.
Share created successfully.
Performing the operation "New drive" on target "Name: DS001 Provider: MicrosoftDeploymentToolkit\MI
Performing the operation "new" on target "deployment share".
Initializing a new deployment share
Initializing scripts and tools.
Performing the operation "open" on target "deployment share".
Deployment share at 'C:\ReferenceShare' opened successfully.
Adding MDT drive DS001 to the persisted drive list.
Successfully added MDT drive DS001 to the persisted drive list.
```

The next is the **View Script** button that does just that, reveals the PowerShell script that was run with the variables you provided. This is handy if you need to document what was done and be able to reproduce it at will for change control, versioning, disaster recovery, or any other reason.

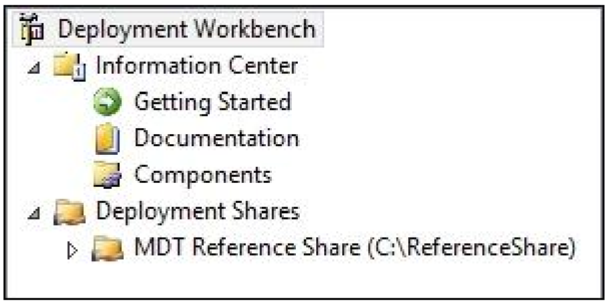
So, once you finish running the wizard, click on the **View Script** button. It will open **Notepad**, where the script is present in raw text, as shown in the following screenshot:



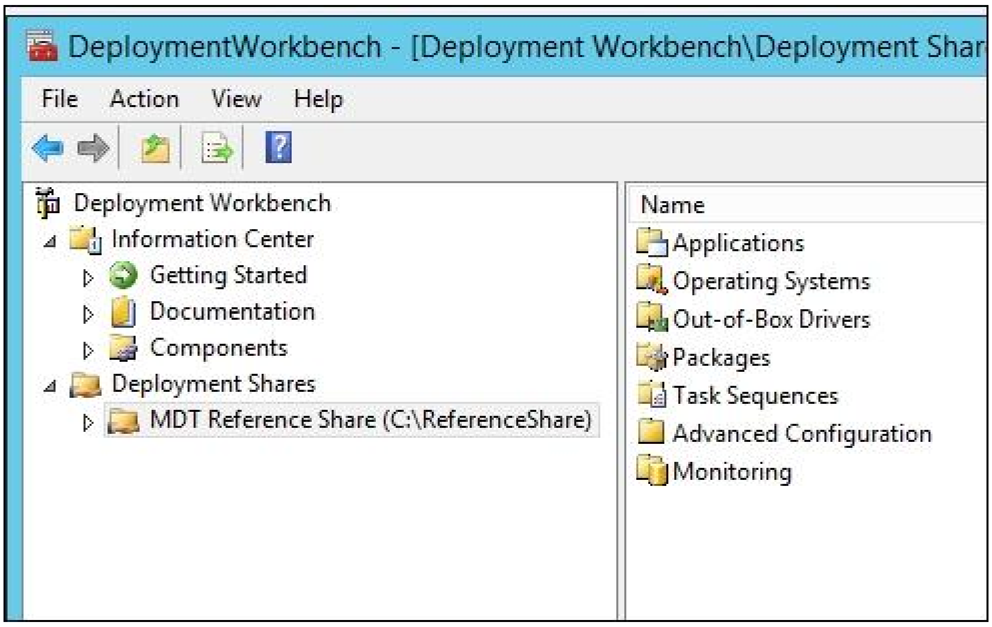
```
Import-Module "C:\Program Files\Microsoft Deployment Toolkit\bin\MicrosoftDeploymentToolkit.psd1"
new-PSDrive -Name "DS001" -PSProvider "MDTProvider" -Root "C:\ReferenceShare" -Description "MDT Reference Share" -NetworkPa
```

Exploring the completed reference share

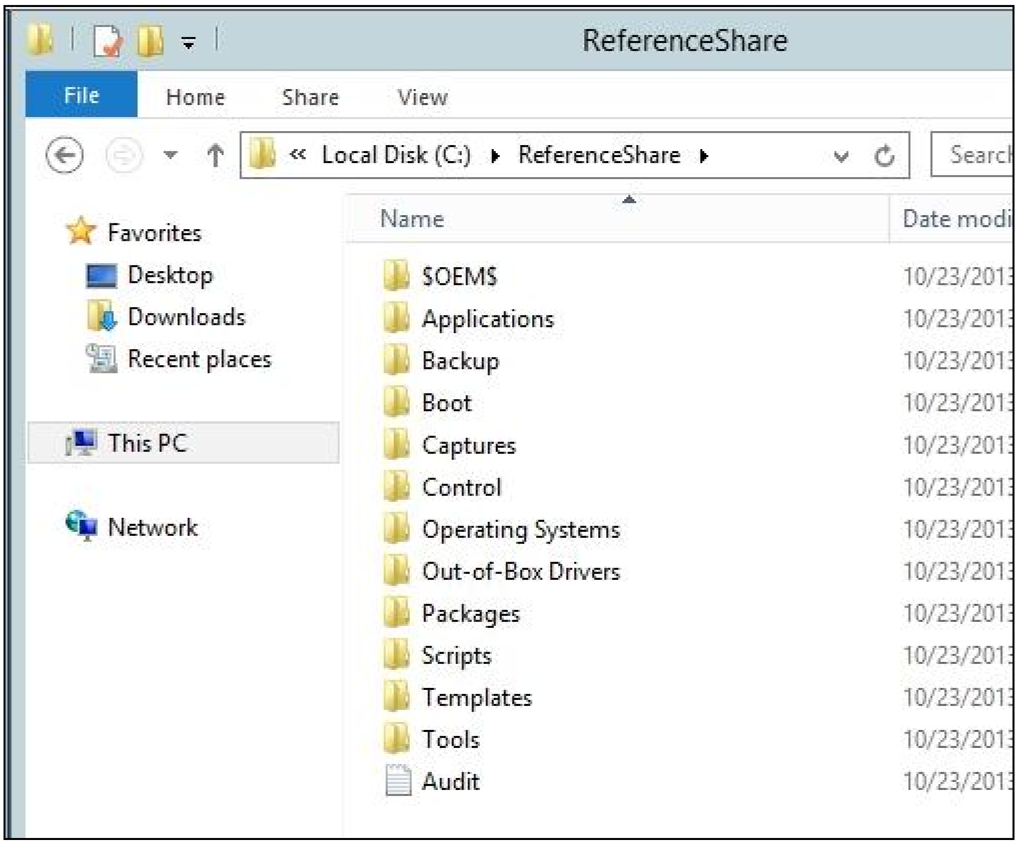
If we've completed the previous steps, we'll be rewarded with a reference share as shown in the following image, under **Deployment Shares**:



Clicking on **MDT Reference Share** will give us a view of some folders. These mirror, to some extent, is the flat filesystem that we have created:



As opposed to the filesystem shown in the following screenshot:



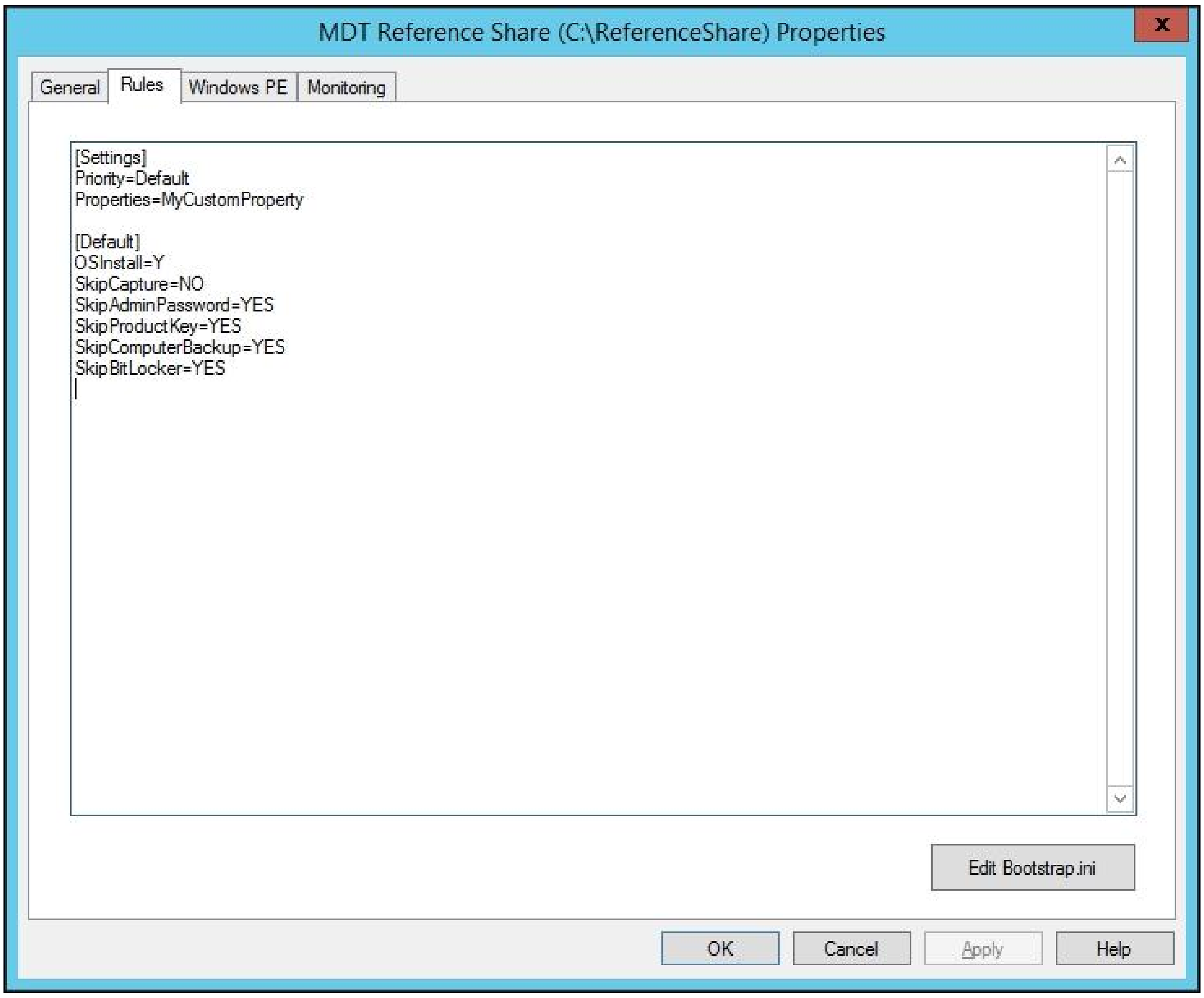
Note the presence of or absence of an attribute, depending on your scenario requirements, matching folders. MDT is keeping some records out of the **user interface** (**UI**) for us. This is quite intentional.

TIP

For typical best practice, one does not have to go into the actual filesystem for anything. In fact, all the times I've had to modify the filesystem rather than the UI, I was doing something I either shouldn't have been doing (we'll call it experimenting) or trying to fix something someone else had done (by their experimenting).

Setting reference properties

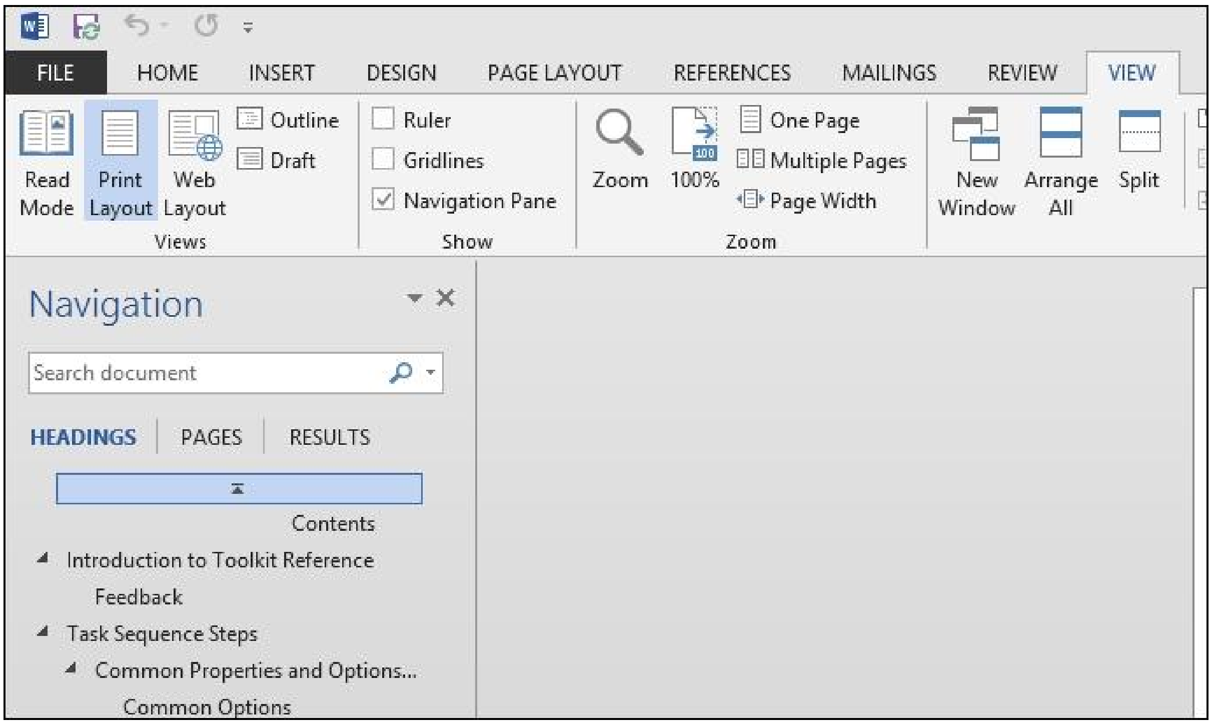
The first thing to do with our reference share is to open the properties and set some items. Getting into the share properties is quite easy; right-click on **MDT Reference Share** and select **Properties**:



```
[Settings]
Priority=Default
Properties=MyCustomProperty

[Default]
OSInstall=Y
SkipCapture=NO
SkipAdminPassword=YES
SkipProductKey=YES
SkipComputerBackup=YES
SkipBitLocker=YES
```

This will bring up the **MDT Reference Share Properties** window. Go to the **Rules** tab, where there are several `Skip` properties. These are used to skip past the MDT wizard that runs during a task sequence.

Think practically; what would you want to skip to make this an automated process, and what are your options? The MDT Print Ready Documentation Pack has a list of all the Skips. You will download the pack at the same location used for MDT 2013. Open the `Toolkit Reference.docx` document and use the **Navigation** pane view to search for Skips in order to find all the Skips headings in the document-simply type `Skip` in the **Search document** field:



NOTE

The Toolkit Reference document can also be directly accessed at https://technet.microsoft.com/en-us/library/dn781091.aspx

Setting up our reference share task sequence

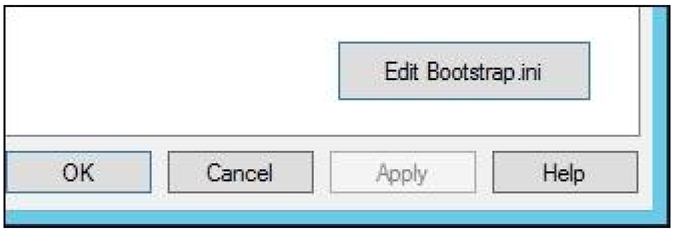
We will want to skip things for a reference share that are different than things skipped for a deployment share. The reason for this is that in a reference share task sequence, we will typically want this to run in a fully automated fashion (Zero Touch-Lite Touch, if you will). So when we need to rerun the task sequence in the future, we can simply apply a change, fire up the virtual machine targeted to be run, run the task sequence, and off we go.

Therefore, the `Rules` section can look similar to the following code. This is properly formatted and skips many steps, which in a reference image task sequence, would be considered superfluous. Note that `UserID`, `UserDomain`, and `UserPassword` will be dependent on your configuration. In the following example, I am using a domain user account with rights to the share (read and write, as the reference image has to be written as a WIM file into this share using the ID):

```
[Settings]
Priority=Default
```

```
Properties=MyCustomProperty

[Default]
OSInstall=Y
SkipAppsOnUpgrade=YES
SkipCapture=NO
SkipAdminPassword=YES
SkipProductKey=YES
_SMSTSOrgName=MDT Reference Task Sequence
SkipBitLocker=YES
SkipDomainMembership=YES
JoinWorkgroup=Workgroup
SkipFinalSummary=YES
SkipLocaleSelection=YES
SkipSummary=YES
SkipTimeZone=YES
SkipUserData=YES
TimeZoneName=Eastern Standard Time
UserID=<account with rights to the MDT share>
UserDomain=<domain>
UserPassword=<password>
FinishAction=SHUTDOWN
```

Next, you will click on **Apply** and then click on the **Edit Bootstrap.ini** button shown in the following screenshot, as we have more customization to do:



I will add some items here to speed the task sequence along. Your `UserID`, `UserDomain`, and `UserPassword` are all similar to the ones in the `Rules` section:

```
[Settings]
Priority=Default

[Default]
DeployRoot=\\mdt-share\Reference Share
UserID=< >
UserDomain=< >
UserPassword=< >
SkipBDDWelcome=YES
```

The reason for these `Skip` commands is to skip parts of the MDT wizard. As we're setting up a reference image, we don't need to be prompted for things such as time zone or fluffy welcome screens.

Increasing the scratch space

Some driver installations have a problem with low memory availability. Increasing scratch space is the best way to avoid these issues. Click on the **Windows PE** tab and then increase the scratch space to `128`.

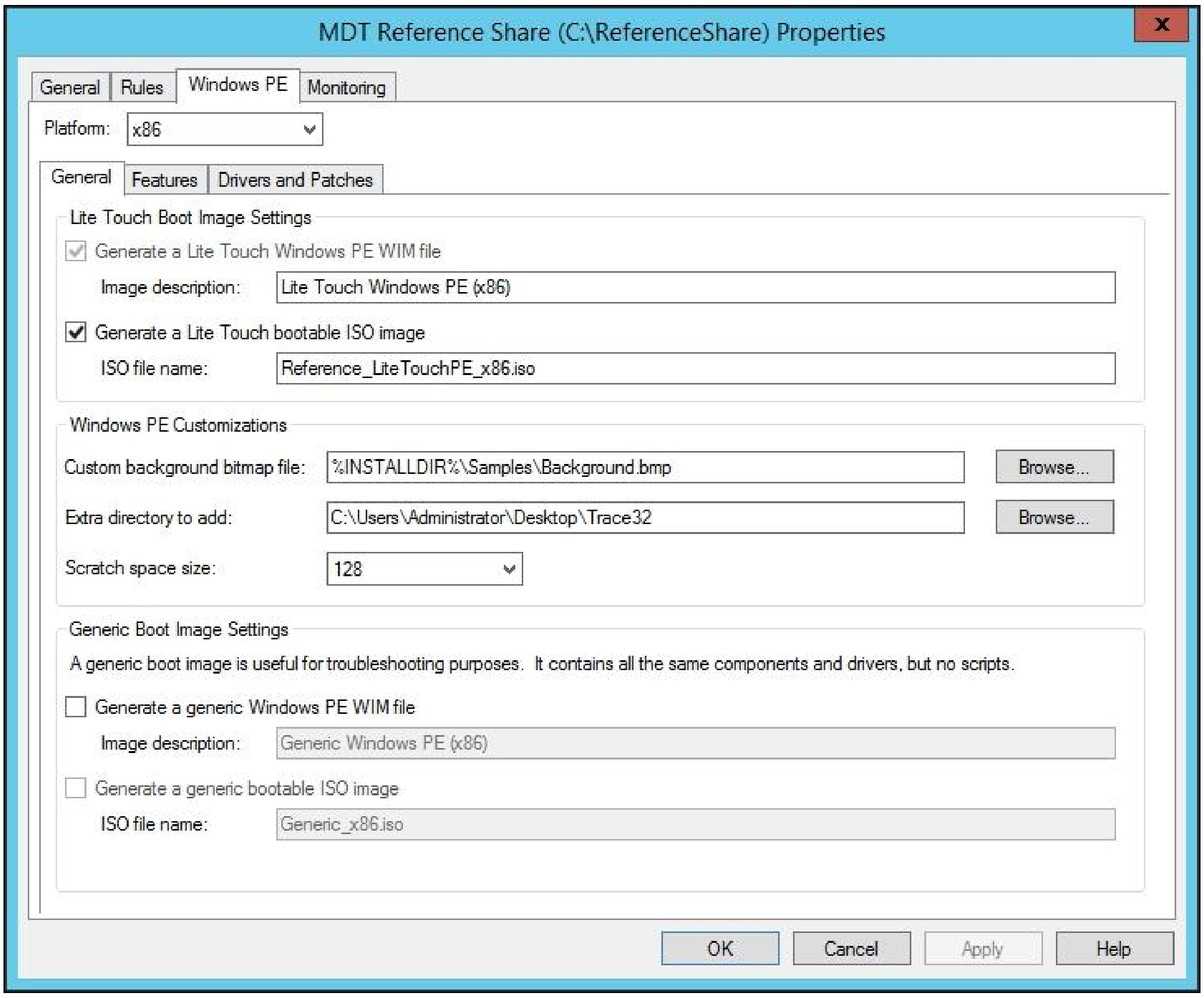
Including trace files in the boot WIM

A general troubleshooting best practice is to include `trace32.exe` in the boot WIM. This tool is part of the **System Center Configuration Manager** (**SCCM**) tools and can be used to interpret the `.log` files that are produced by the task sequence engine in a much more legible manner.

To do this, you will install Trace32 from the SCCM tools and then copy `trace32.exe` into a directory. In this case, I have placed it in a directory named `Trace32` on the `Desktop`. Then just specify this directory for inclusion in the WinPE WIM generation.

Naming ISO-generated files

In the **Windows PE** tab, name your ISO-generated files something logical. For the purpose of this example, we'll be naming them `Reference_LiteTouchPE_x86.iso`. Right now, your **Windows PE** tab should look similar to the following image:


MDT Reference Share (C:\ReferenceShare) Properties

General | Rules | Windows PE | Monitoring

Platform: x86

General | Features | Drivers and Patches

Lite Touch Boot Image Settings

[✓] Generate a Lite Touch Windows PE WIM file

Image description: Lite Touch Windows PE (x86)

[✓] Generate a Lite Touch bootable ISO image

ISO file name: Reference_LiteTouchPE_x86.iso

Windows PE Customizations

Custom background bitmap file: %INSTALLDIR%\Samples\Background.bmp

Extra directory to add: C:\Users\Administrator\Desktop\Trace32

Scratch space size: 128

Generic Boot Image Settings

A generic boot image is useful for troubleshooting purposes. It contains all the same components and drivers, but no scripts.

[ ] Generate a generic Windows PE WIM file

Image description: Generic Windows PE (x86)

[ ] Generate a generic bootable ISO image

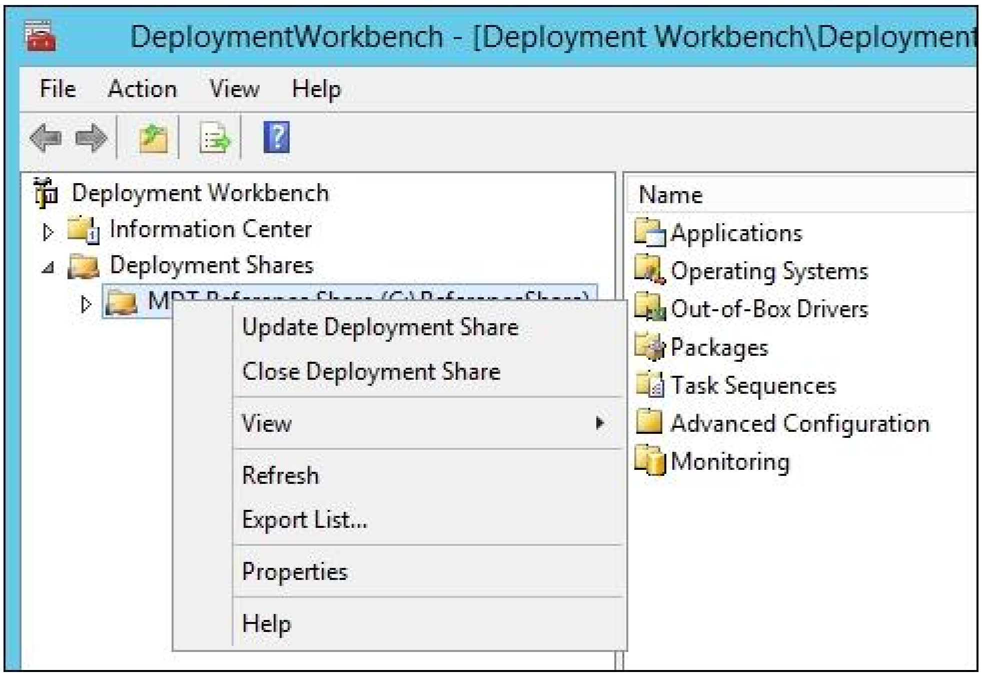
ISO file name: Generic_x86.iso

TIP

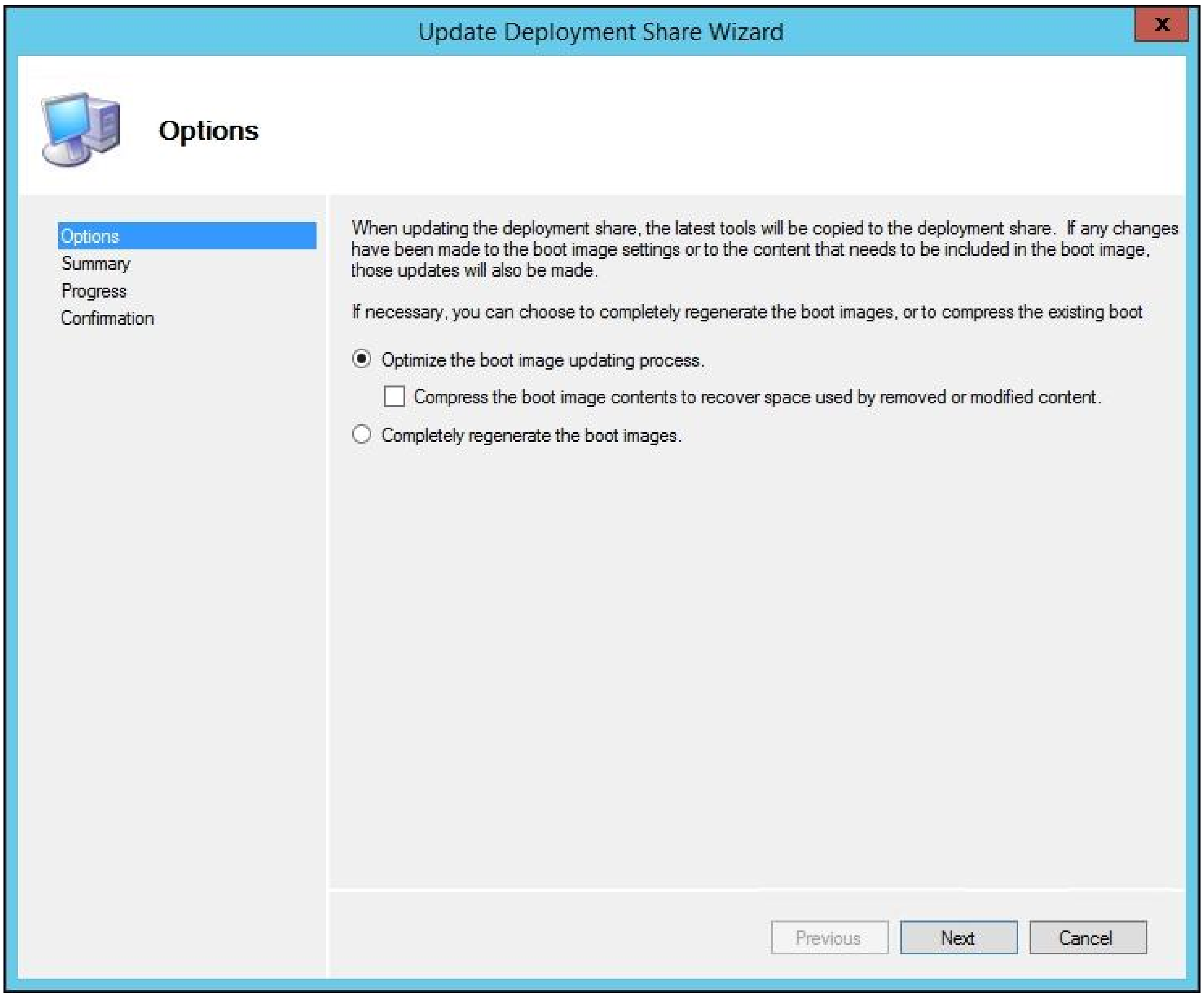
Don't forget that you must modify the x86 and x64 media separately!

Updating up the deployment share

Next we're going to walk through updating the deployment share. Right-click on the reference share that we've just set up, and select **Update Deployment Share**:



The **Update Deployment Share Wizard** dialog box will appear, as shown in the following image:



When updating the deployment share, the latest tools will be copied to the deployment share. If any changes have been made to the boot image settings or to the content that needs to be included in the boot image, those updates will also be made.

If necessary, you can choose to completely regenerate the boot images, or to compress the existing boot

● Optimize the boot image updating process.

☐ Compress the boot image contents to recover space used by removed or modified content.

○ Completely regenerate the boot images.

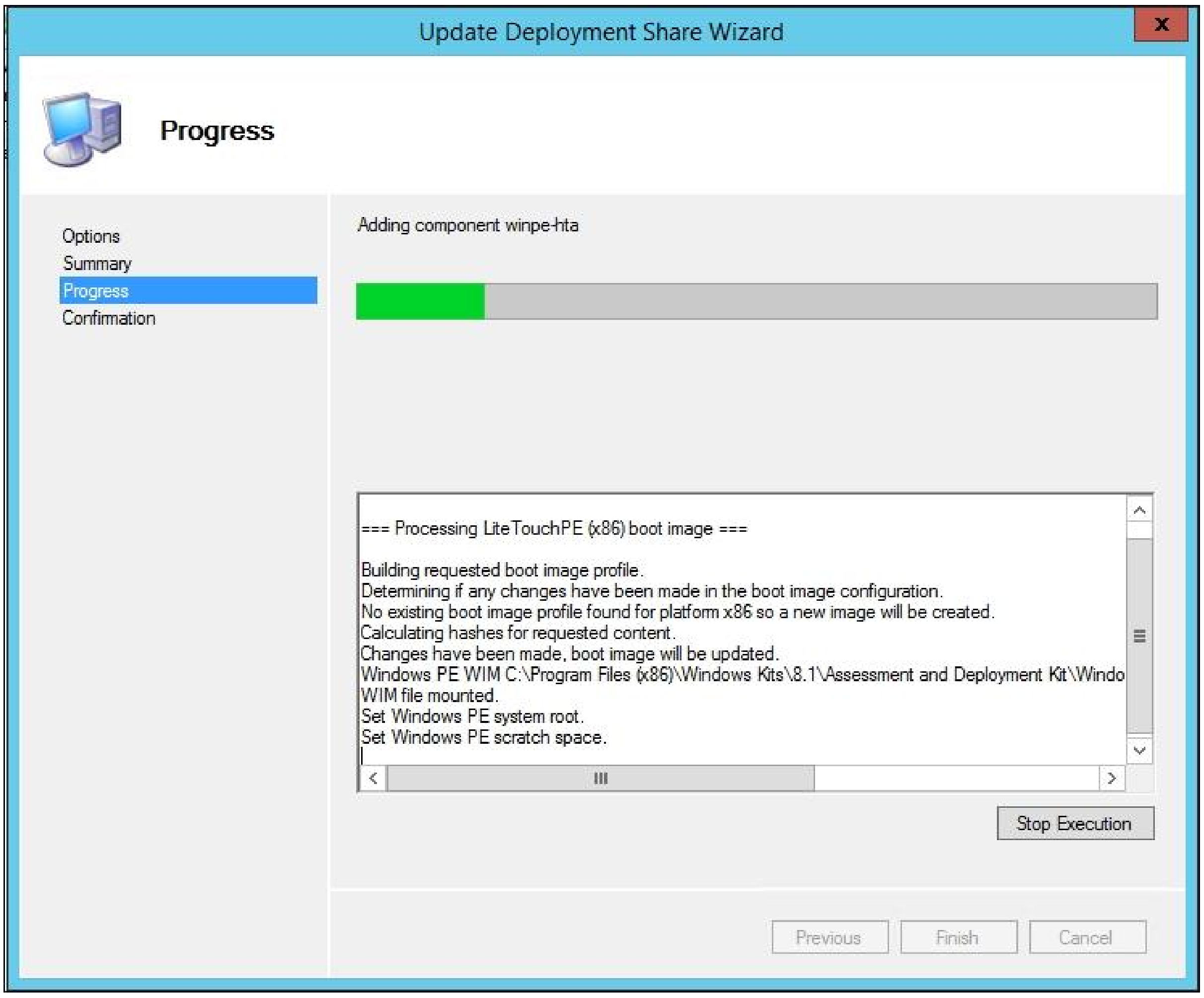
The defaults, as shown in the previous image, are fine here. You will typically only select the compress checkbox when you have retired a model of hardware and want to reclaim this driver whitespace out of the WinPE build. If you don't do this, it'll be consumed as you add newer drivers to the WinPE image. Click on **Next**.

TIP

The **Completely regenerate the boot images** option is typically used as a troubleshooting step when your boot media appears confused and it is best to rebuild from a known good set of settings rather than try to figure out what went wrong.

Automatic boot media creation

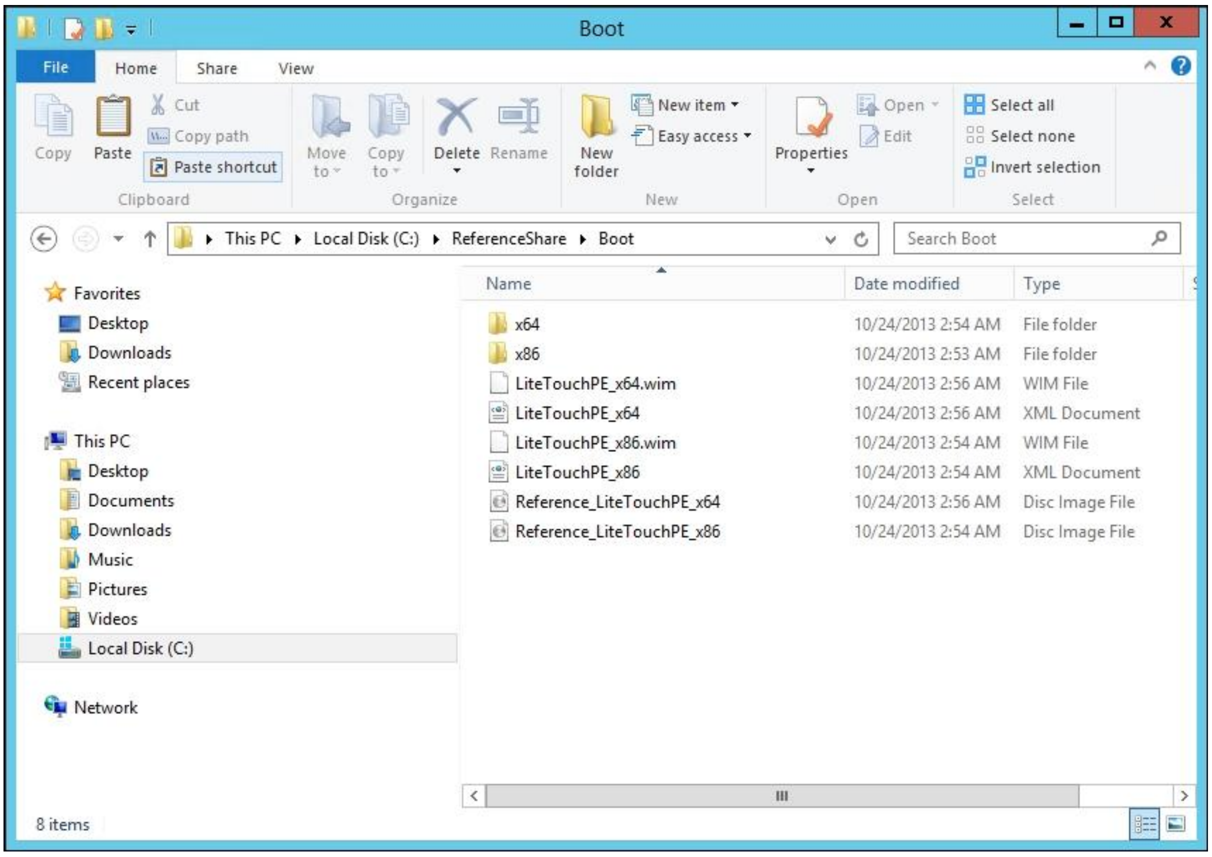
In the following screenshot of the **Progress** screen, MDT indicates that no existing boot image exists, so it is creating a new image. This is correct and expected. After you have created boot media once, and a change is made that requires a media rebuild, the verbiage will be different, but it has the same type of indication that something has changed and an update to the media is needed. Note that it is updating x86 here; it'll update x64 later. If you had unchecked one of the architectures, it would only update the one selected here:



Adding component winpe-hta

```
=== Processing LiteTouchPE (x86) boot image ===

Building requested boot image profile.
Determining if any changes have been made in the boot image configuration.
No existing boot image profile found for platform x86 so a new image will be created.
Calculating hashes for requested content.
Changes have been made, boot image will be updated.
Windows PE WIM C:\Program Files (x86)\Windows Kits\8.1\Assessment and Deployment Kit\Windo
WIM file mounted.
Set Windows PE system root.
Set Windows PE scratch space.
```

After the process is complete, you should be rewarded with boot media, as shown in the following screenshot:

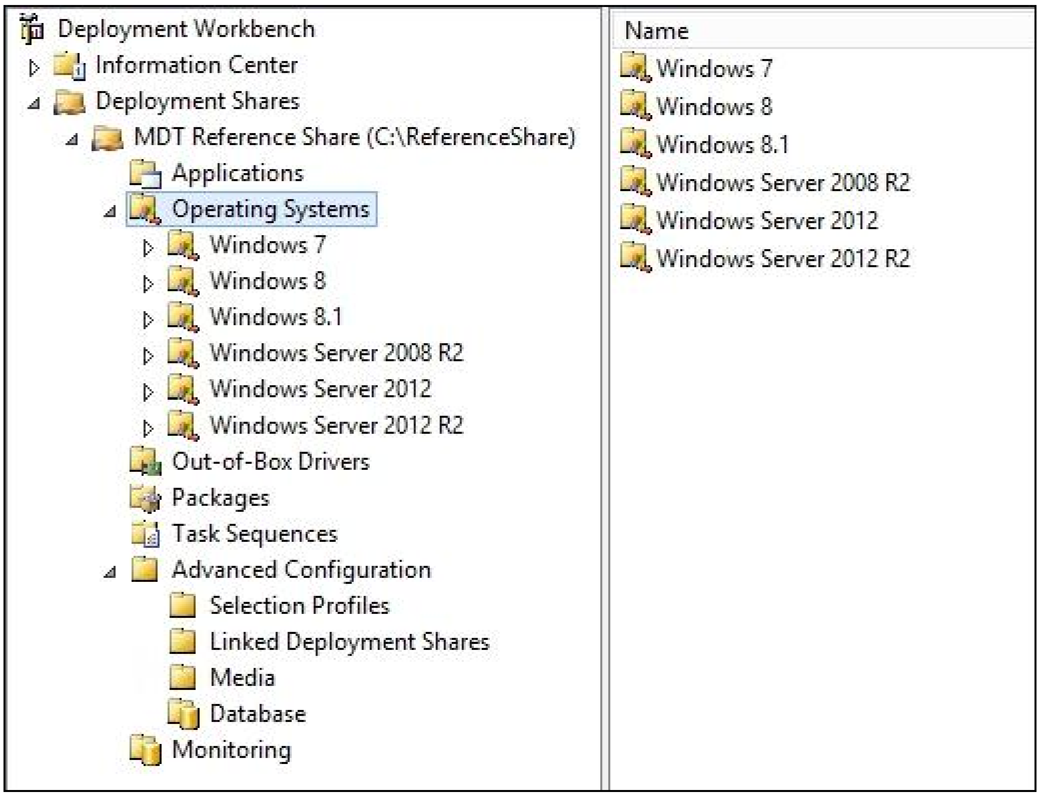


Name	Date modified	Type
x64	10/24/2013 2:54 AM	File folder
x86	10/24/2013 2:53 AM	File folder
LiteTouchPE_x64.wim	10/24/2013 2:56 AM	WIM File
LiteTouchPE_x64	10/24/2013 2:56 AM	XML Document
LiteTouchPE_x86.wim	10/24/2013 2:54 AM	WIM File
LiteTouchPE_x86	10/24/2013 2:54 AM	XML Document
Reference_LiteTouchPE_x64	10/24/2013 2:56 AM	Disc Image File
Reference_LiteTouchPE_x86	10/24/2013 2:54 AM	Disc Image File

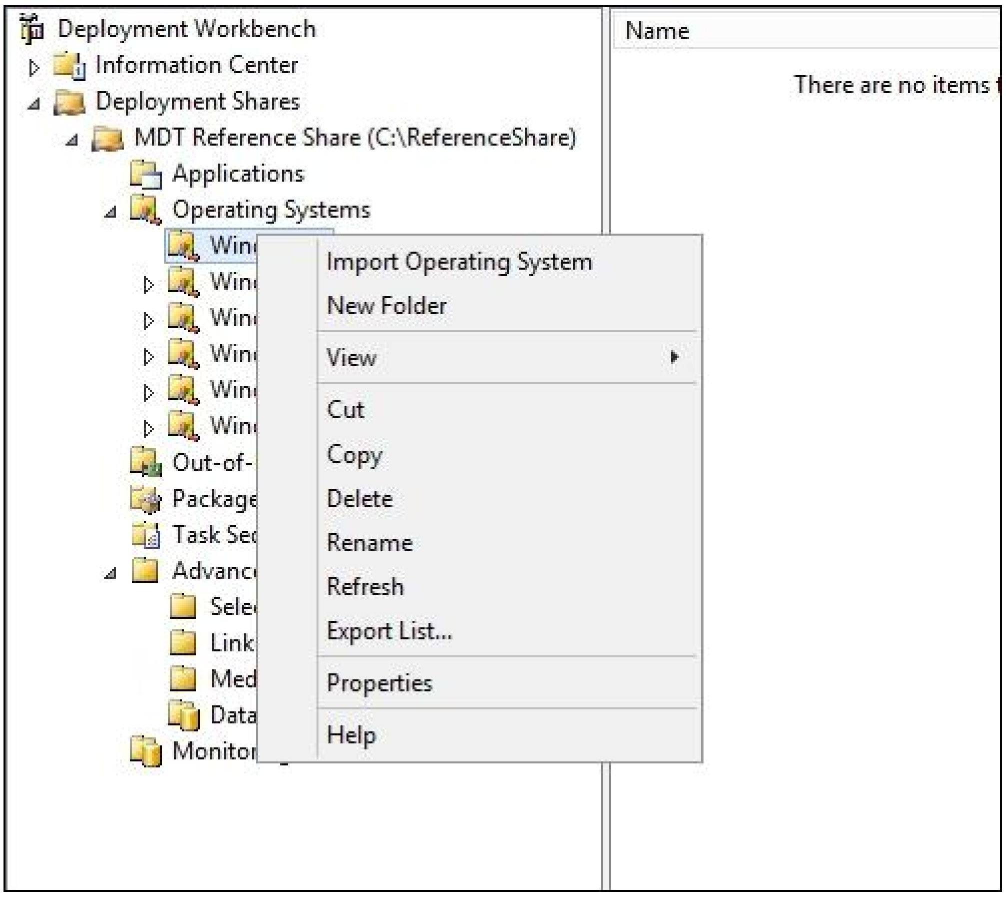
Note that we have WIM and ISO format files, so we can use these in WDS or mount the ISO to a virtual machine (which is the plan in this book), or even write the ISO to a USB stick or CD/DVD for boot media.

Importing an OS

Next we will be importing our OS. Create a folder structure in the **Operating Systems** area. This helps with the organization of selection profiles, which are a tool for driver organization we will get into later. It also looks better and is easier to logically manage, as shown in the following image:

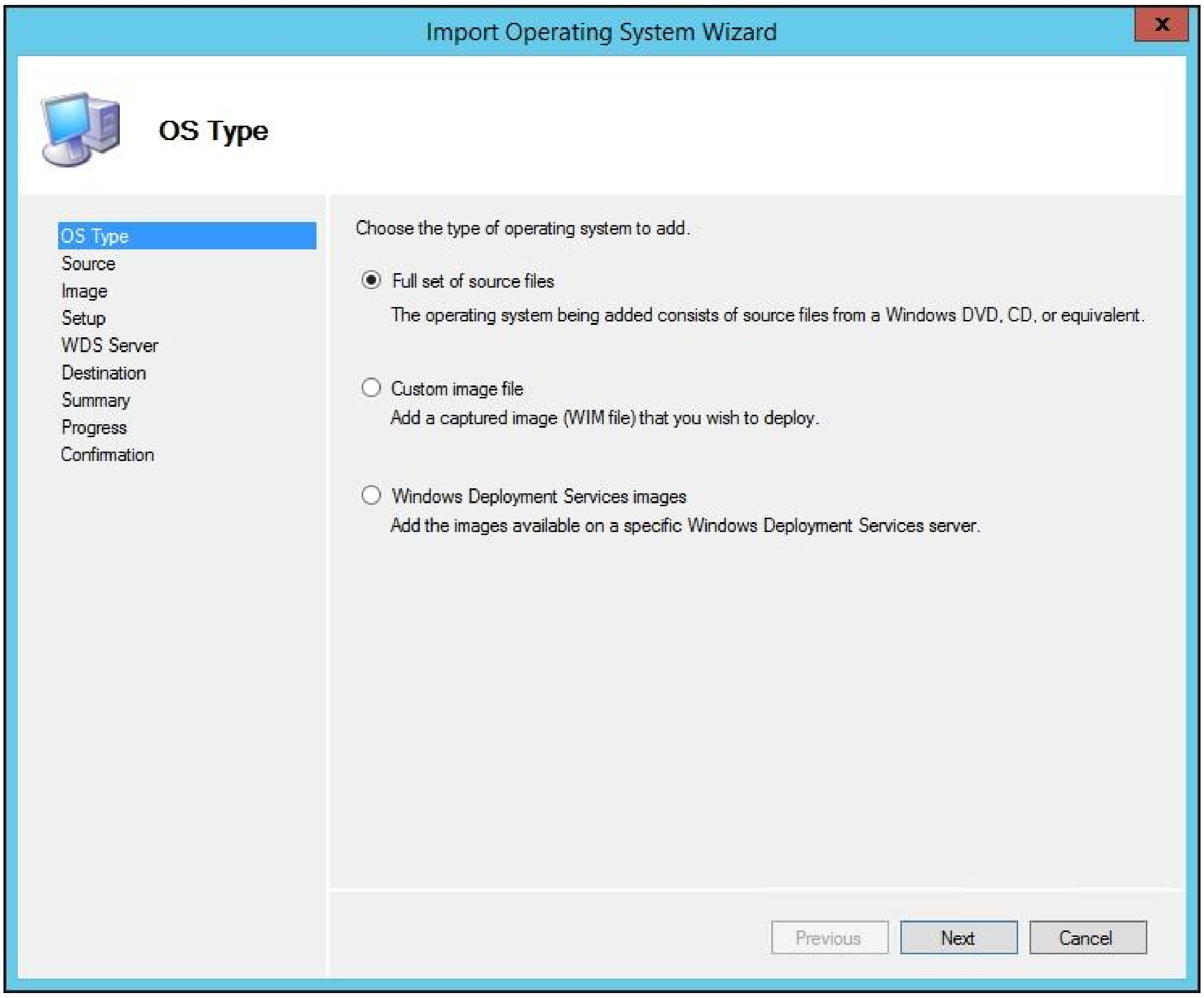


Right-click on the appropriate folder for the OS that you wish to import and click on **Import Operating System**. We need the ISO for media for this. I downloaded mine from MSDN and then mounted the ISO in Hyper-V so that it is presented to the MDT Server as a DVD:



Choosing the type of OS to add

After selecting **Import Operating System**, you'll be asked something that probably doesn't make a lot of sense, **Choose the type of operating system to add**, as shown in the following screenshot:



Choose the type of operating system to add.

● Full set of source files

The operating system being added consists of source files from a Windows DVD, CD, or equivalent.

○ Custom image file

Add a captured image (WIM file) that you wish to deploy.

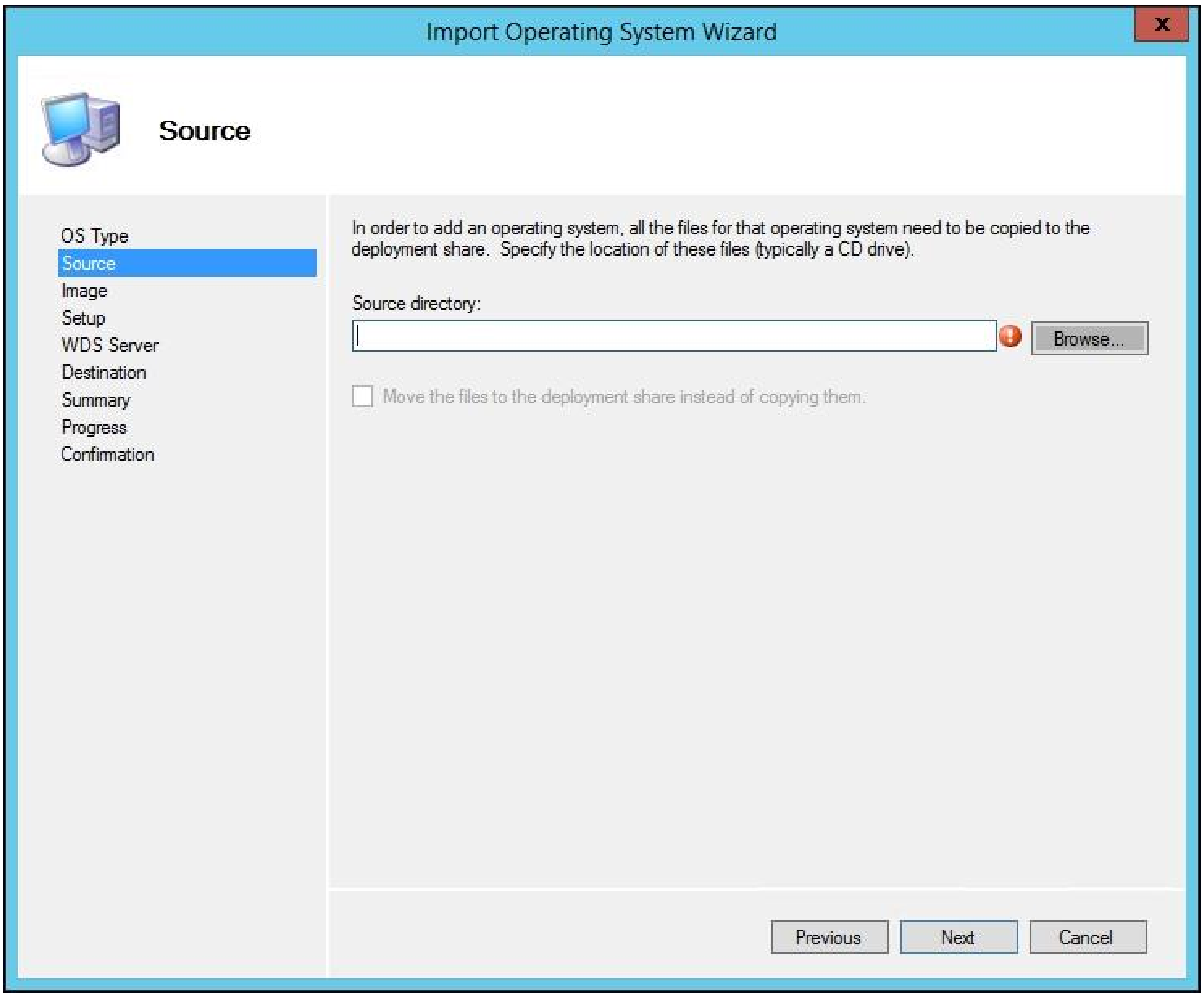
○ Windows Deployment Services images

Add the images available on a specific Windows Deployment Services server.

For a reference image, we'd pretty much always start with a base OS DVD and use the top radio button for **Full set of source files**. However, the options are there to import a custom WIM that is already captured, or a WDS image as well.

Importing an OS from DVD media

In the **Source** screen, which is shown in the following image, point the import wizard to the root of the DVD media:


Import Operating System Wizard

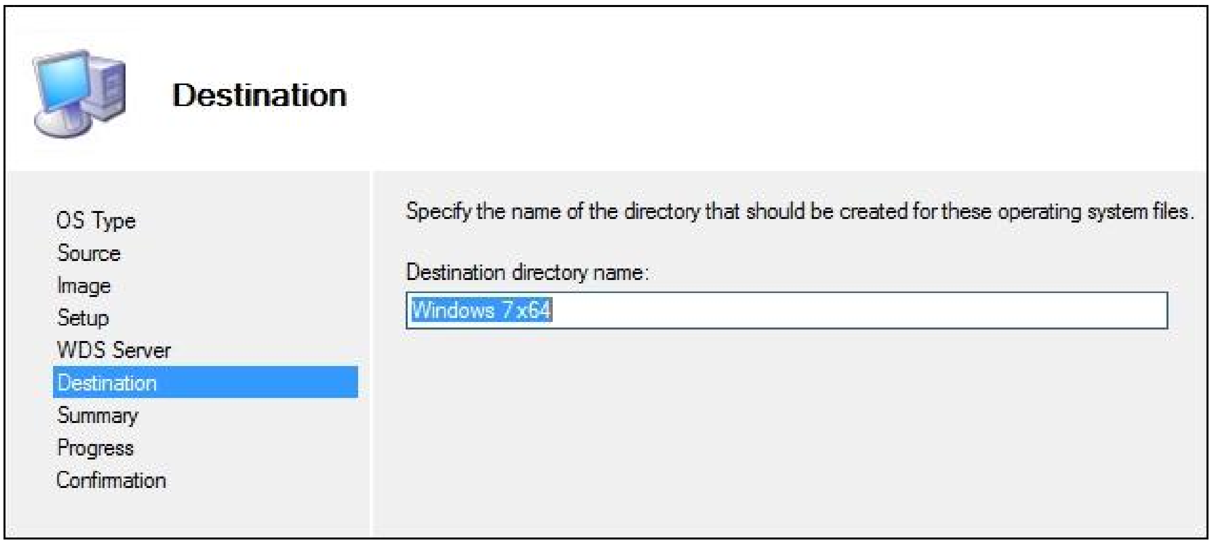
Source

In order to add an operating system, all the files for that operating system need to be copied to the deployment share. Specify the location of these files (typically a CD drive).

Source directory:

Move the files to the deployment share instead of copying them.

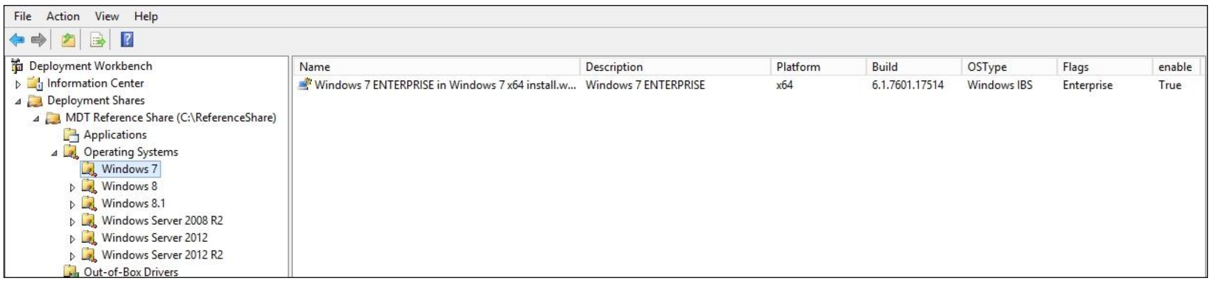
The wizard will read the media and autofills the window with the media to be imported, as shown in the following image:


Destination

Specify the name of the directory that should be created for these operating system files.

Destination directory name: Windows 7 x64

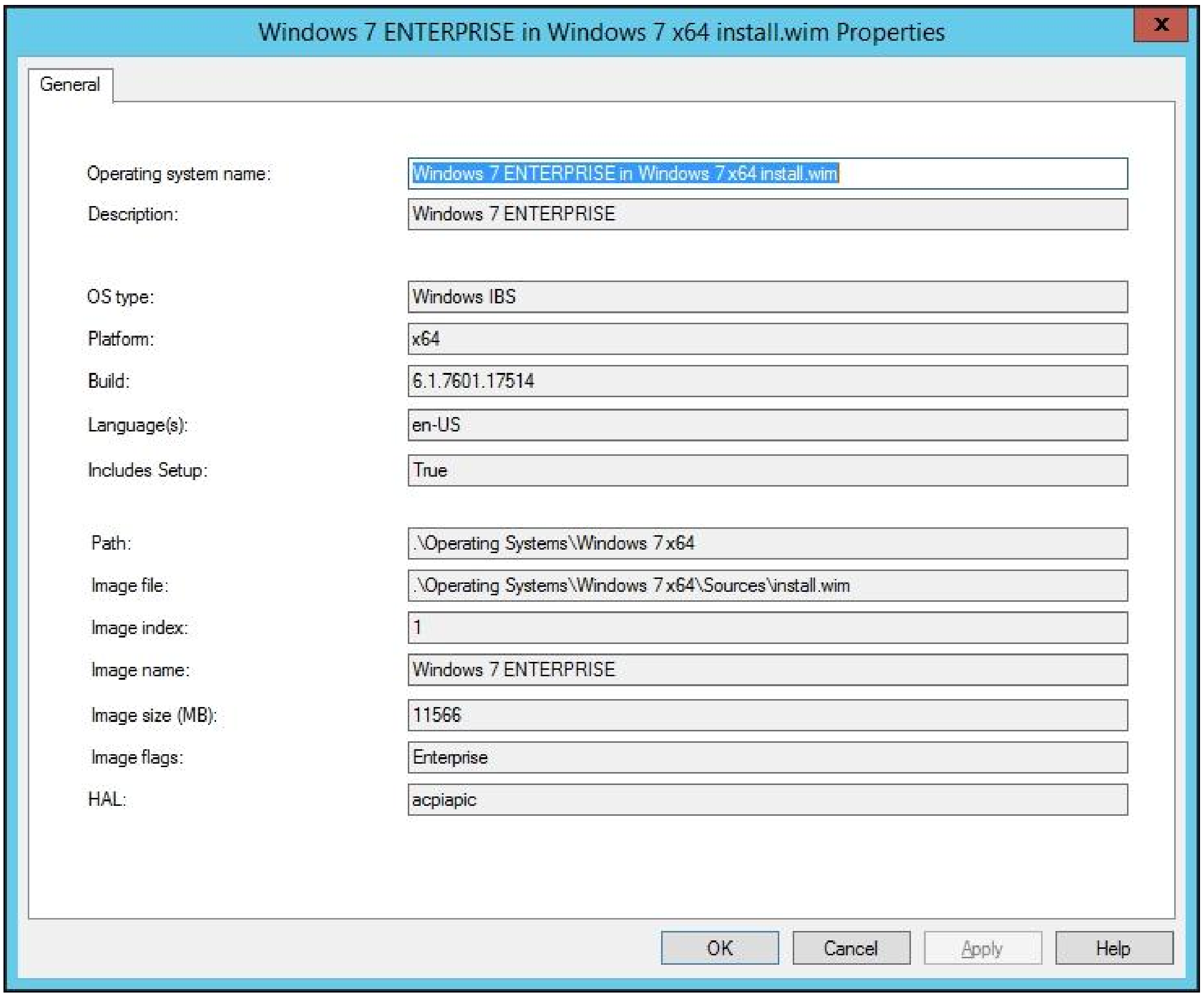
Click on **Next** and then the job will import the media files into the specified OS folder and you'll be rewarded with an OS in your share, as shown in the following screenshot:



Name	Description	Platform	Build	OSType	Flags	enable
Windows 7 ENTERPRISE in Windows 7 x64 install.w...	Windows 7 ENTERPRISE	x64	6.1.7601.17514	Windows IBS	Enterprise	True

Viewing image properties

If there is ever a question on what the image is, a simple right-click and selecting **Properties** will reveal a lot of details, which we can see in the following image:


Windows 7 ENTERPRISE in Windows 7 x64 install.wim Properties
General
OK
Cancel
Apply
Help

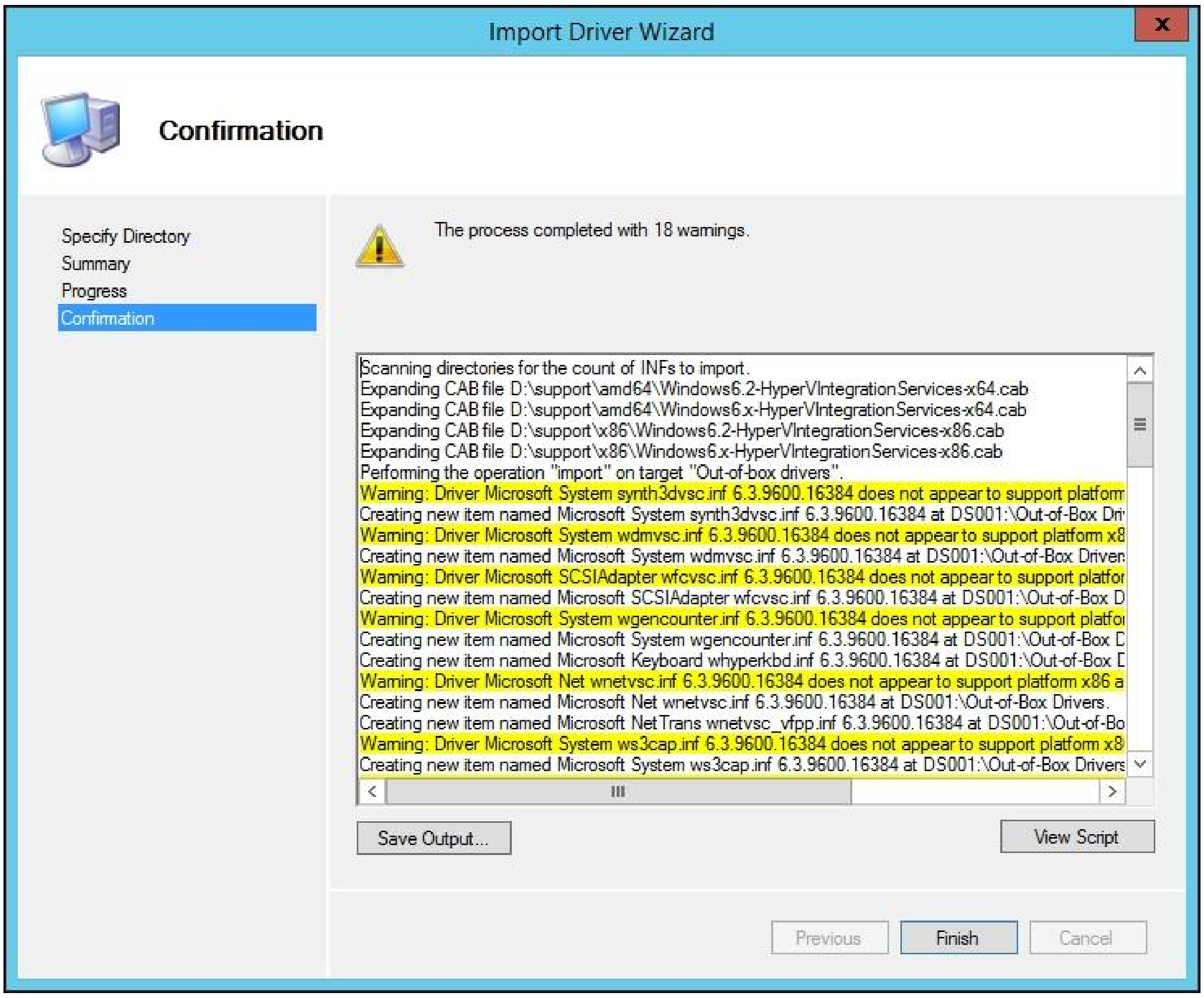
Operating system name:	Windows 7 ENTERPRISE in Windows 7 x64 install.wim
Description:	Windows 7 ENTERPRISE
OS type:	Windows IBS
Platform:	x64
Build:	6.1.7601.17514
Language(s):	en-US
Includes Setup:	True
Path:	.\Operating Systems\Windows 7 x64
Image file:	.\Operating Systems\Windows 7 x64\Sources\install.wim
Image index:	1
Image name:	Windows 7 ENTERPRISE
Image size (MB):	11566
Image flags:	Enterprise
HAL:	acpiapic

Importing Hyper-V drivers

Our next step is to include the new Hyper-V additions in WinPE and the OS.

We would use this, for example, if we are capturing and deploying Windows 7, but are using Windows Server 2012 or 2012 R2 to run the capture from. The same process applies for physical drivers, but remember that we are only capturing a reference image into a VM, not physical. The drivers for actual model numbers will come later, in our deployment share. So in the Hyper-V Virtual Machine Connection, I selected **Action** and then **Insert Integration Components**. This causes the Hyper-V host to mount the ISO for the current Hyper-V additions for the system on to the `CD-ROM`, `D:`, so now I can import these into the **Out-of-Box Drivers** area and update my WinPE media.

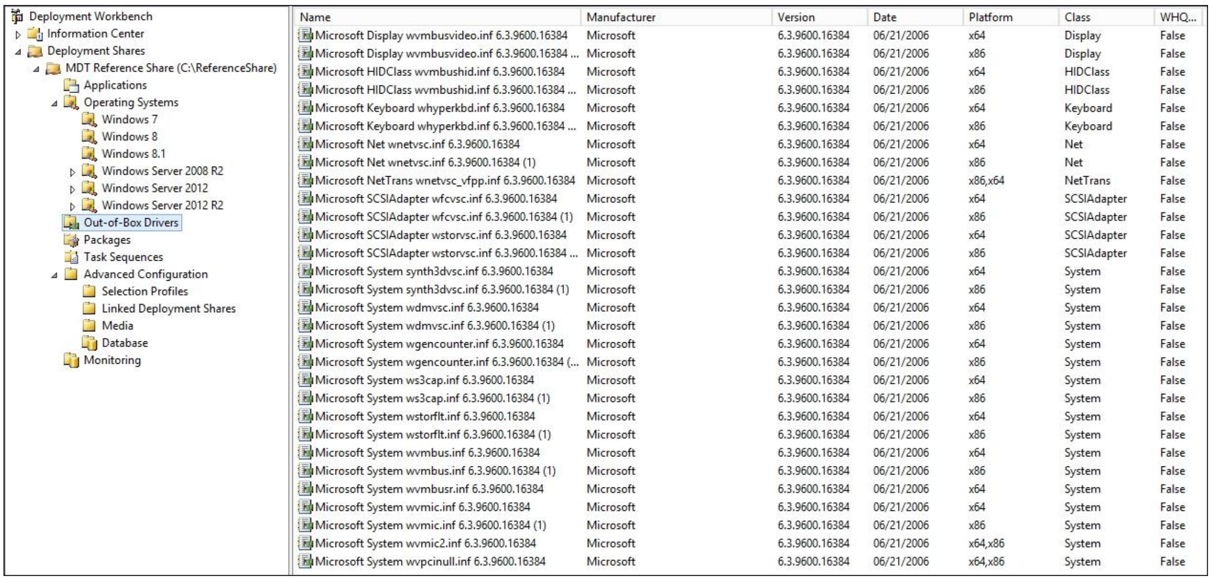
Importing drivers is as simple as putting the drivers into a directory and right-clicking on **Out-of-Box Drivers** and selecting **Import Drivers**. Point the wizard at the root folder where the driver is located and it'll crawl the directory's tree and import all the drivers it can, as shown in the following image:



The process completed with 18 warnings.

```
Scanning directories for the count of INFs to import.
Expanding CAB file D:\support\amd64\Windows6.2-HyperVIntegrationServices-x64.cab
Expanding CAB file D:\support\amd64\Windows6.x-HyperVIntegrationServices-x64.cab
Expanding CAB file D:\support\x86\Windows6.2-HyperVIntegrationServices-x86.cab
Expanding CAB file D:\support\x86\Windows6.x-HyperVIntegrationServices-x86.cab
Performing the operation "import" on target "Out-of-box drivers".
Warning: Driver Microsoft System synth3dvsc.inf 6.3.9600.16384 does not appear to support platform
Creating new item named Microsoft System synth3dvsc.inf 6.3.9600.16384 at DS001:\Out-of-Box Dri
Warning: Driver Microsoft System wdmvsc.inf 6.3.9600.16384 does not appear to support platform x8
Creating new item named Microsoft System wdmvsc.inf 6.3.9600.16384 at DS001:\Out-of-Box Driver
Warning: Driver Microsoft SCSIAdapter wfcvsc.inf 6.3.9600.16384 does not appear to support platfor
Creating new item named Microsoft SCSIAdapter wfcvsc.inf 6.3.9600.16384 at DS001:\Out-of-Box D
Warning: Driver Microsoft System wgencounter.inf 6.3.9600.16384 does not appear to support platfo
Creating new item named Microsoft System wgencounter.inf 6.3.9600.16384 at DS001:\Out-of-Box D
Creating new item named Microsoft Keyboard whyperkbd.inf 6.3.9600.16384 at DS001:\Out-of-Box D
Warning: Driver Microsoft Net wnetvsc.inf 6.3.9600.16384 does not appear to support platform x86 a
Creating new item named Microsoft Net wnetvsc.inf 6.3.9600.16384 at DS001:\Out-of-Box Drivers.
Creating new item named Microsoft NetTrans wnetvsc_vfpp.inf 6.3.9600.16384 at DS001:\Out-of-Bo
Warning: Driver Microsoft System ws3cap.inf 6.3.9600.16384 does not appear to support platform x8
Creating new item named Microsoft System ws3cap.inf 6.3.9600.16384 at DS001:\Out-of-Box Drivers
```

Then your share will reflect drivers in it, as shown in the following image:



Name	Manufacturer	Version	Date	Platform	Class	WHQ...
Microsoft Display wvmbusvideo.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	Display	False
Microsoft Display wvmbusvideo.inf 6.3.9600.16384 ...	Microsoft	6.3.9600.16384	06/21/2006	x86	Display	False
Microsoft HIDClass wvmbushid.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	HIDClass	False
Microsoft HIDClass wvmbushid.inf 6.3.9600.16384 ...	Microsoft	6.3.9600.16384	06/21/2006	x86	HIDClass	False
Microsoft Keyboard whyperkbd.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	Keyboard	False
Microsoft Keyboard whyperkbd.inf 6.3.9600.16384 ...	Microsoft	6.3.9600.16384	06/21/2006	x86	Keyboard	False
Microsoft Net wnetvsc.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	Net	False
Microsoft Net wnetvsc.inf 6.3.9600.16384 (1)	Microsoft	6.3.9600.16384	06/21/2006	x86	Net	False
Microsoft NetTrans wnetvsc_vfpp.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x86,x64	NetTrans	False
Microsoft SCSIAdapter wfcvsc.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	SCSIAdapter	False
Microsoft SCSIAdapter wfcvsc.inf 6.3.9600.16384 (1)	Microsoft	6.3.9600.16384	06/21/2006	x86	SCSIAdapter	False
Microsoft SCSIAdapter wstorvsc.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	SCSIAdapter	False
Microsoft SCSIAdapter wstorvsc.inf 6.3.9600.16384 ...	Microsoft	6.3.9600.16384	06/21/2006	x86	SCSIAdapter	False
Microsoft System synth3dvsc.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	System	False
Microsoft System synth3dvsc.inf 6.3.9600.16384 (1)	Microsoft	6.3.9600.16384	06/21/2006	x86	System	False
Microsoft System wdmvsc.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	System	False
Microsoft System wdmvsc.inf 6.3.9600.16384 (1)	Microsoft	6.3.9600.16384	06/21/2006	x86	System	False
Microsoft System wgencounter.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	System	False
Microsoft System wgencounter.inf 6.3.9600.16384 (...	Microsoft	6.3.9600.16384	06/21/2006	x86	System	False
Microsoft System ws3cap.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	System	False
Microsoft System ws3cap.inf 6.3.9600.16384 (1)	Microsoft	6.3.9600.16384	06/21/2006	x86	System	False
Microsoft System wstorflt.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	System	False
Microsoft System wstorflt.inf 6.3.9600.16384 (1)	Microsoft	6.3.9600.16384	06/21/2006	x86	System	False
Microsoft System wvmbus.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	System	False
Microsoft System wvmbus.inf 6.3.9600.16384 (1)	Microsoft	6.3.9600.16384	06/21/2006	x86	System	False
Microsoft System wvmbusr.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	System	False
Microsoft System wvmic.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64	System	False
Microsoft System wvmic.inf 6.3.9600.16384 (1)	Microsoft	6.3.9600.16384	06/21/2006	x86	System	False
Microsoft System wvmic2.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64,x86	System	False
Microsoft System wvpcinull.inf 6.3.9600.16384	Microsoft	6.3.9600.16384	06/21/2006	x64,x86	System	False

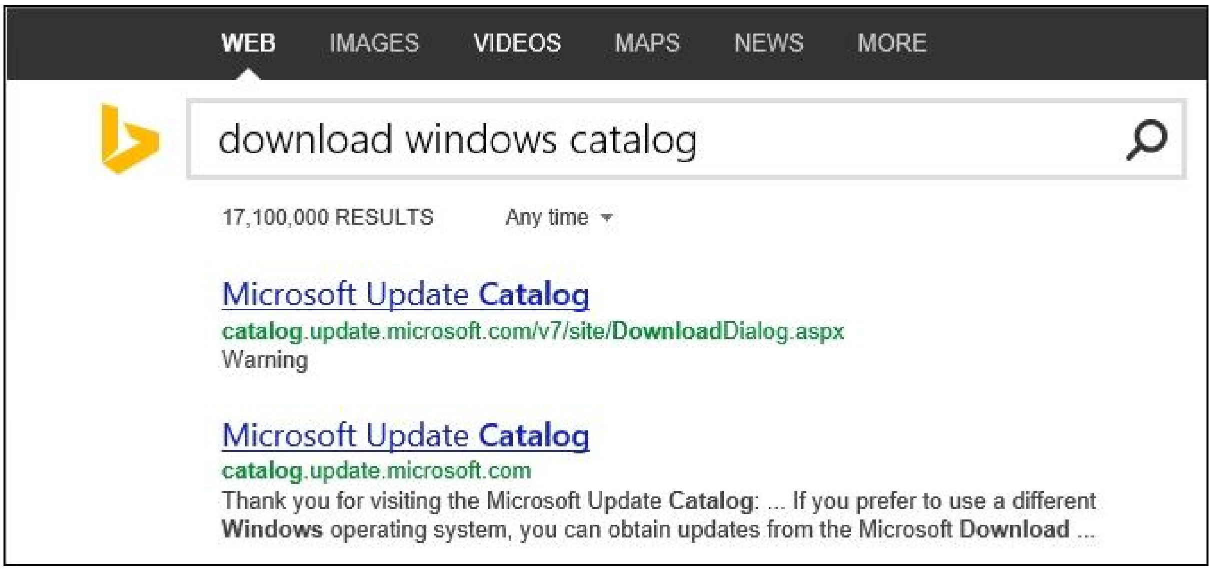
These drivers will auto apply to WinPE base images when you update the deployment share, and they will also be injected (by default) into Windows installations as required by Plug and Play, as part of the task sequence. We will cover this in greater depth in Chapter 6 , *Drivers*.

Importing patches

Now that we have imported Hyper-V drivers, let's move on to importing patches. In the following example, I am going to include the `2775511` hotfix in my base Windows 7 image (and 2008 R2 when I get around to importing the Server SKUs as well). The `2775511` hotfix is an enterprise hotfix rollup, which is available for Windows 7 SP1 and Windows Server 2008 R2 SP1.

Downloading a hotfix

First we must acquire the file. This is done by downloading it from Windows Catalog rather than registering and pulling down a hotfix link over e-mail:



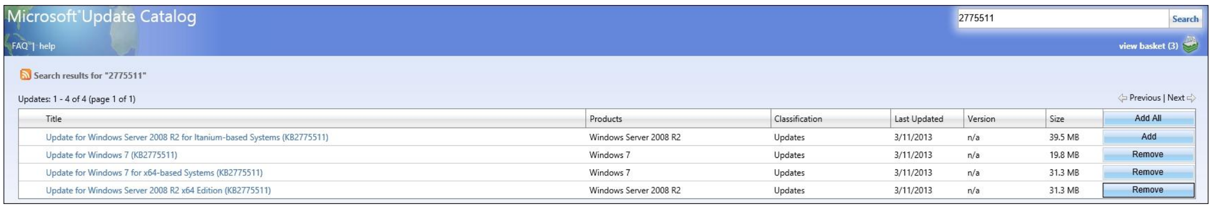
17,100,000 RESULTS Any time

Microsoft Update Catalog
catalog.update.microsoft.com/v7/site/DownloadDialog.aspx
Warning

Microsoft Update Catalog
catalog.update.microsoft.com
Thank you for visiting the Microsoft Update Catalog: ... If you prefer to use a different **Windows** operating system, you can obtain updates from the Microsoft **Download** ...

In the catalog engine, put the hotfix ID (`2775511`) into the search bar and click on **Search**.

Then, you will get the results that hopefully look like the following image:



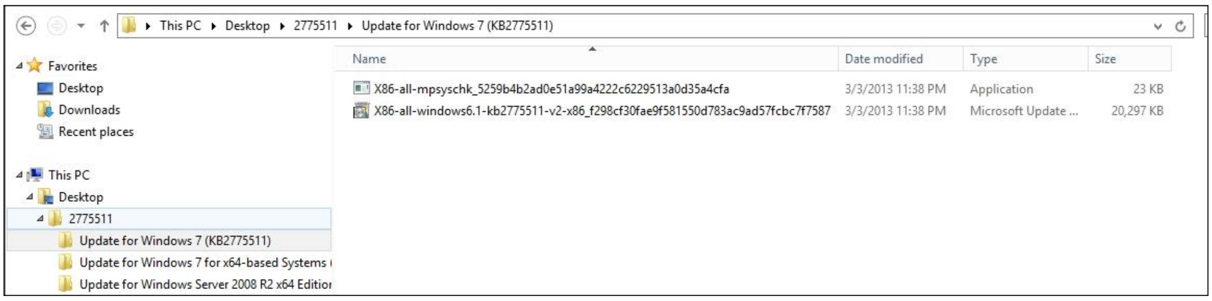
Search results for "2775511"

Updates: 1 - 4 of 4 (page 1 of 1)

Title	Products	Classification	Last Updated	Version	Size	Add All
Update for Windows Server 2008 R2 for Itanium-based Systems (KB2775511)	Windows Server 2008 R2	Updates	3/11/2013	n/a	39.5 MB	Add
Update for Windows 7 (KB2775511)	Windows 7	Updates	3/11/2013	n/a	19.8 MB	Remove
Update for Windows 7 for x64-based Systems (KB2775511)	Windows 7	Updates	3/11/2013	n/a	31.3 MB	Remove
Update for Windows Server 2008 R2 x64 Edition (KB2775511)	Windows Server 2008 R2	Updates	3/11/2013	n/a	31.3 MB	Remove

Add (in this example) all but the **Itanium** option, click on **view basket**, and then select **Download**. Choose a location to download them in a folder on your local system.

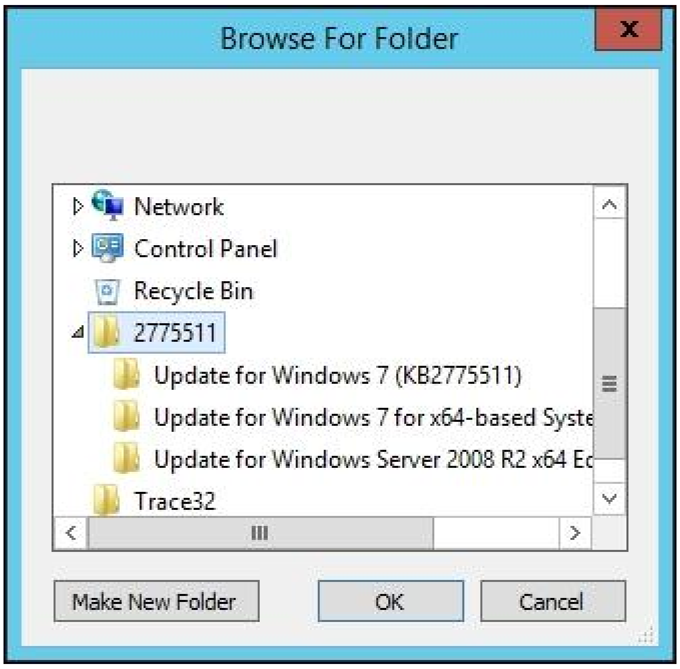
Then they will download it in a folder per hotfix in the specified directory:


This PC ▸ Desktop ▸ 2775511 ▸ Update for Windows 7 (KB2775511)
Favorites
Desktop
Downloads
Recent places
This PC
Desktop
2775511
Update for Windows 7 (KB2775511)
Update for Windows 7 for x64-based Systems
Update for Windows Server 2008 R2 x64 Editior

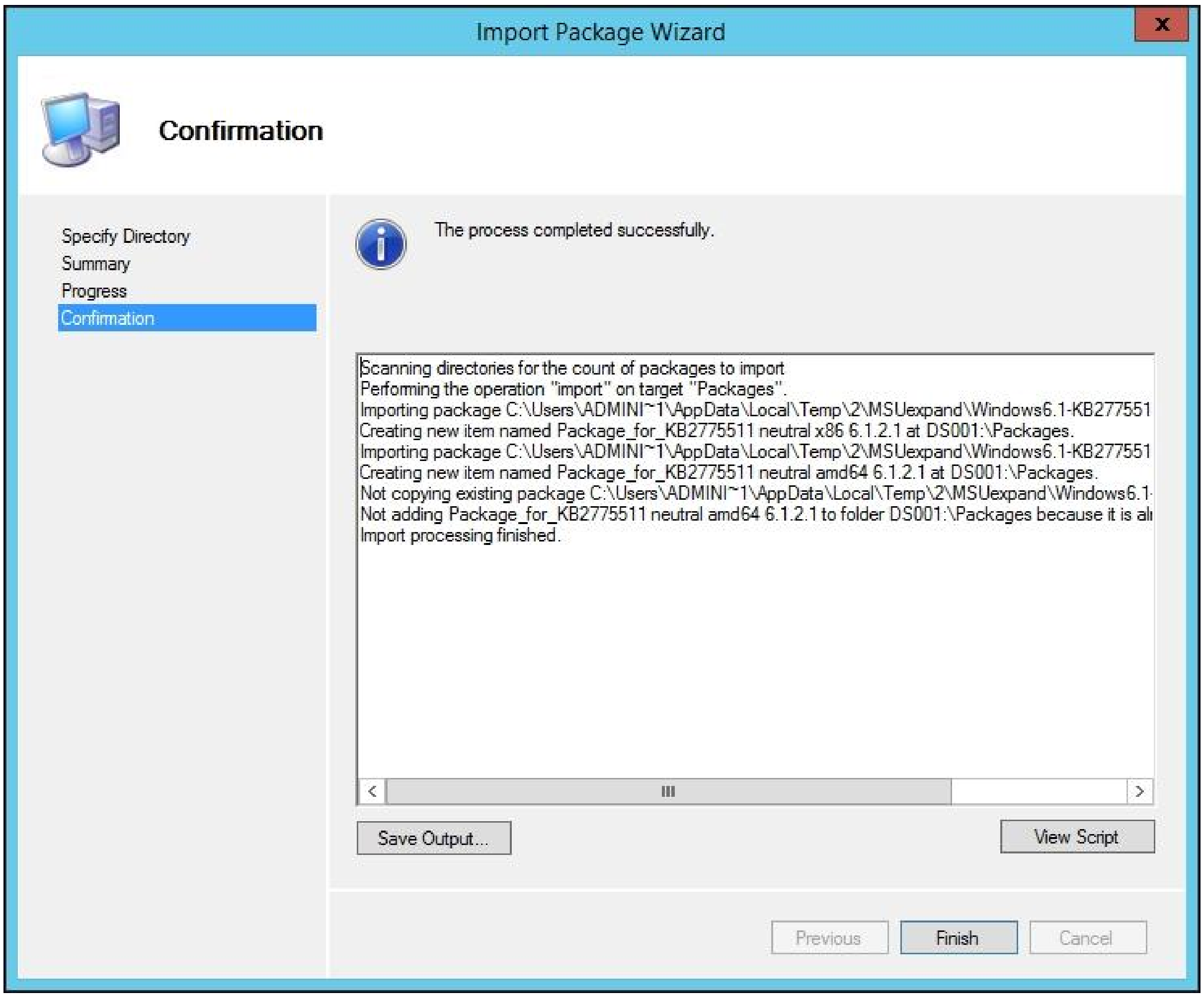
Name	Date modified	Type	Size
X86-all-mpsyschk_5259b4b2ad0e51a99a4222c6229513a0d35a4cfa	3/3/2013 11:38 PM	Application	23 KB
X86-all-windows6.1-kb2775511-v2-x86_f298cf30fae9f581550d783ac9ad57fcbc7f7587	3/3/2013 11:38 PM	Microsoft Update ...	20,297 KB

Setting up a packaged import

Point the MDT wizard to the location where you downloaded the hotfix for a package import:


Browse For Folder
Network
Control Panel
Recycle Bin
2775511
Update for Windows 7 (KB2775511)
Update for Windows 7 for x64-based Syste
Update for Windows Server 2008 R2 x64 Ec
Trace32
Make New Folder
OK
Cancel

Use the wizard to import the packages:



The process completed successfully.

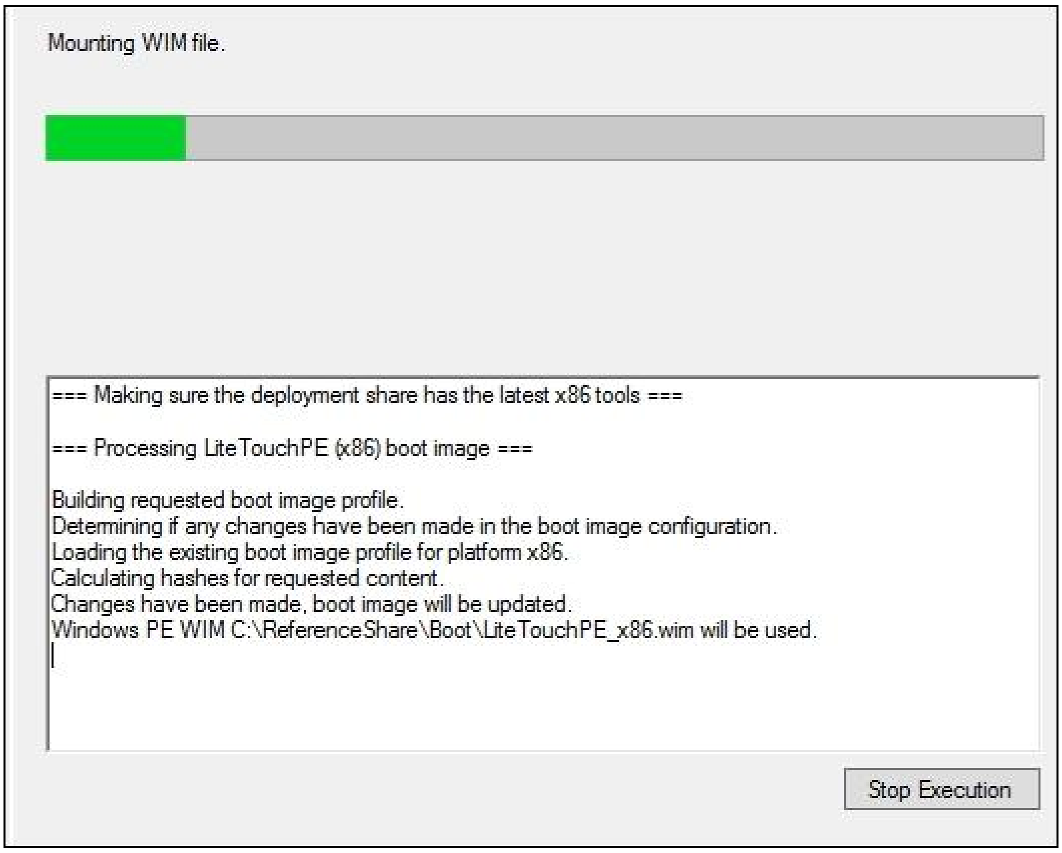
```
Scanning directories for the count of packages to import
Performing the operation "import" on target "Packages".
Importing package C:\Users\ADMINI~1\AppData\Local\Temp\2\MSUexpand\Windows6.1-KB277551
Creating new item named Package_for_KB2775511 neutral x86 6.1.2.1 at DS001:\Packages.
Importing package C:\Users\ADMINI~1\AppData\Local\Temp\2\MSUexpand\Windows6.1-KB277551
Creating new item named Package_for_KB2775511 neutral amd64 6.1.2.1 at DS001:\Packages.
Not copying existing package C:\Users\ADMINI~1\AppData\Local\Temp\2\MSUexpand\Windows6.1-
Not adding Package_for_KB2775511 neutral amd64 6.1.2.1 to folder DS001:\Packages because it is al
Import processing finished.
```

This process is quite easy. For Windows 7, we'll want to add some additional hotfixes, and do the same for 2008 R2. Windows 8, 2012, and higher versions have introduced a servicing stack change that makes it less likely for us to need to pull down one-off hotfixes, but the capability is still here.

Updating the deployment share to include the hotfix

Now that we have added a hotfix, it is time to update the deployment share so that the hotfix can be included in WinPE if needed.

Right-click on **MDT Reference Share** under **Deployment Shares** and select **Update Deployment Share**. You should be able to see a screen similar to the following:



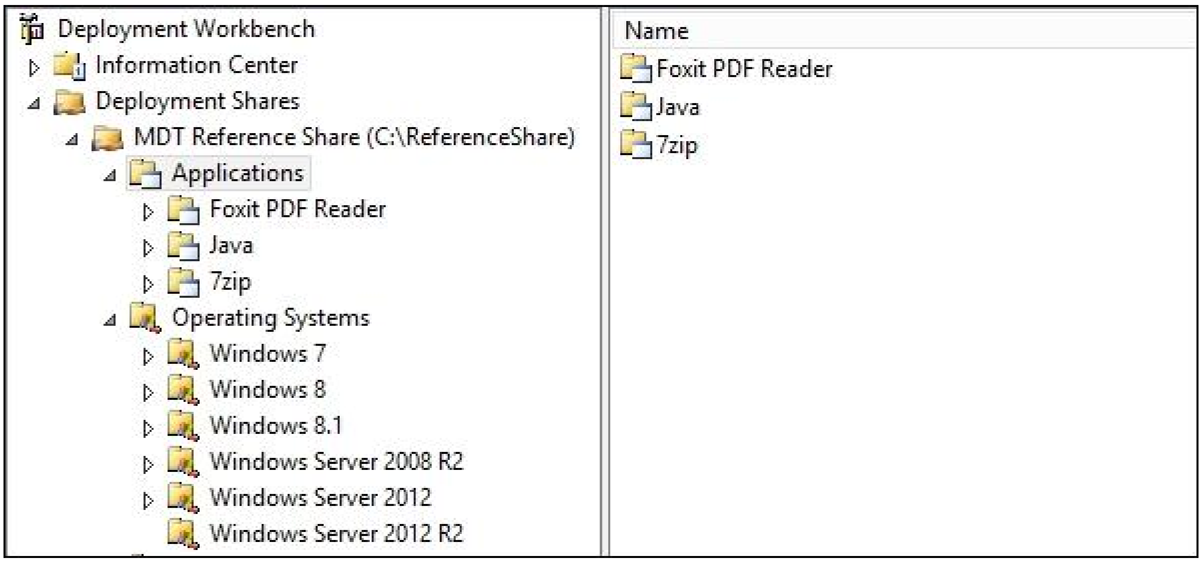
Mounting WIM file.

```
=== Making sure the deployment share has the latest x86 tools ===

=== Processing LiteTouchPE (x86) boot image ===

Building requested boot image profile.
Determining if any changes have been made in the boot image configuration.
Loading the existing boot image profile for platform x86.
Calculating hashes for requested content.
Changes have been made, boot image will be updated.
Windows PE WIM C:\ReferenceShare\Boot\LiteTouchPE_x86.wim will be used.
```

Adding applications to our reference image

Now that we've added patches and drivers, let's add a few applications to our reference image, taking us from a thin image to a hybrid image.

First, we will make a directory structure, as seen in the following image:



Automating image updates

Now, we can just put `setup.exe` for each of these into the folders and say that we're done. However, you don't want to walk through the installer for each application manually every time we capture a reference image. You want this process to be automated so that when new patches are released, we can just fire off a task sequence and let it update the image automatically.

Finding .msi files with the ITNinja repository

To tweak the setup to be automatic for all your different applications, I would recommend checking http://www.itninja.com/. ITNinja is a repository of automatic installation experiences from your peers, which are documented for all time.

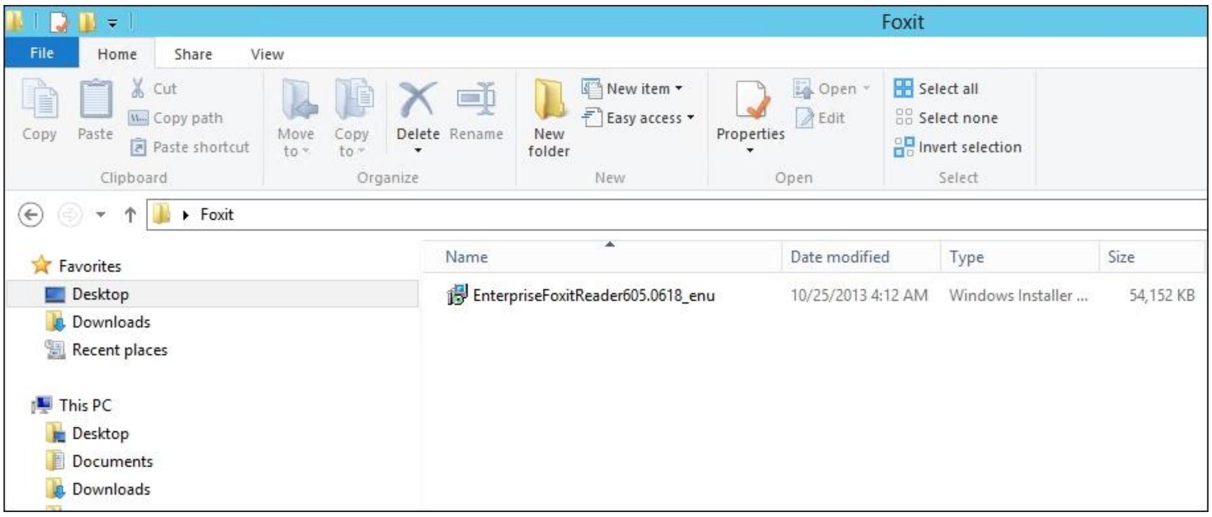
So, for Foxit Reader, I went through some gyrations and form filling on Foxit's site, as documented at the ITNinja site, and was rewarded with a `.msi`, which ITNinja says, can be silently installed using the following command:

```
msiexec /i EnterpriseFoxitReader605.0618_enu.msi /qn
```

Your version may change as the product is updated, but this is the general idea.

Placing the .msi file

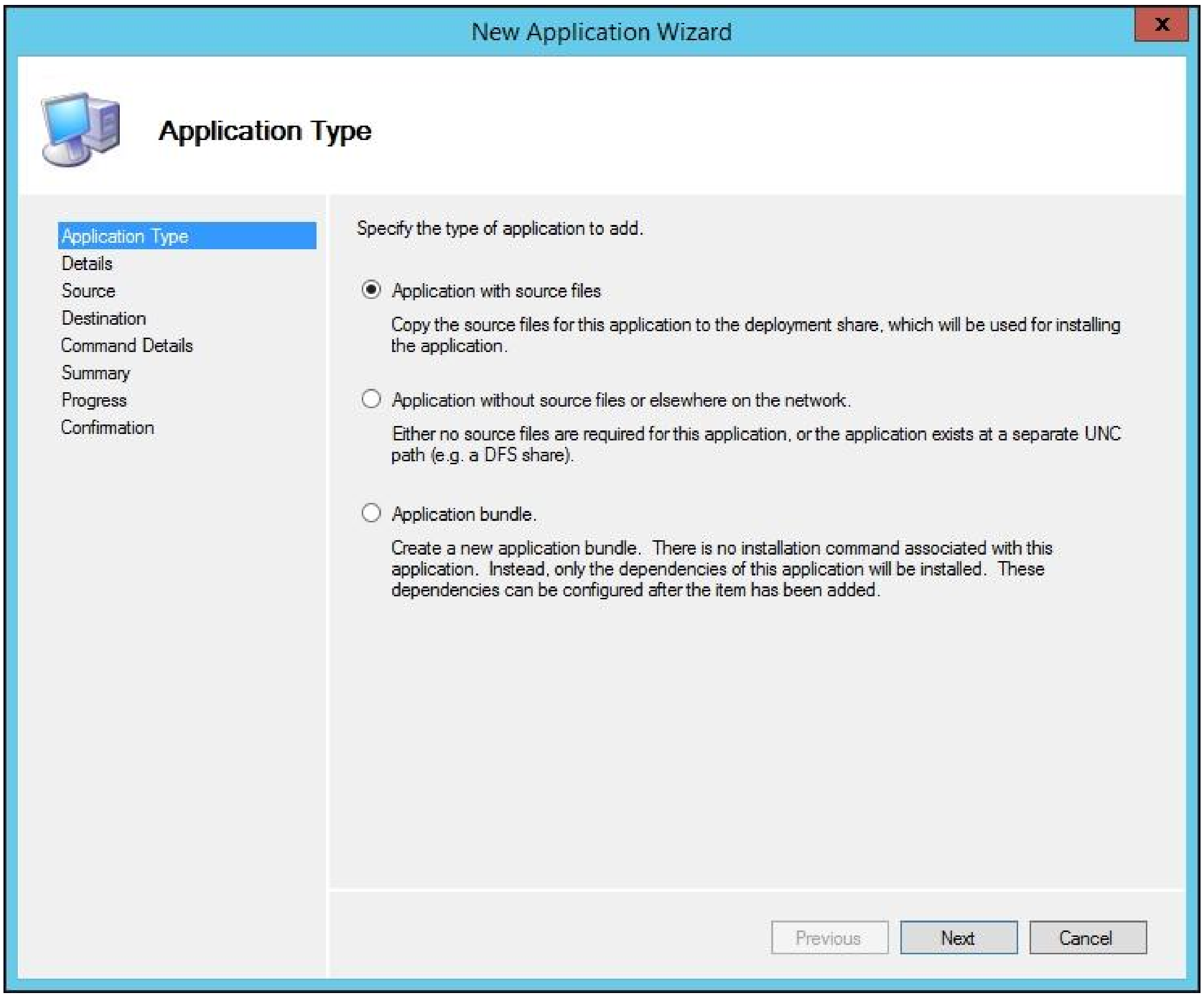
Place the `.msi` file that we downloaded in its *own* individual directory, as shown in the following screenshot. This directory needs to be unique to the `.msi` file as we'll be navigating to it later:



Name	Date modified	Type	Size
EnterpriseFoxitReader605.0618_enu	10/25/2013 4:12 AM	Windows Installer ...	54,152 KB

Setting up a new application

To set up our new application in MDT, just like for drivers or patches, we will right-click on the application folder, and select **New Application**. This generates the **New Application Wizard** window, as shown in the following image:



Specify the type of application to add.

(●) Application with source files

Copy the source files for this application to the deployment share, which will be used for installing the application.

( ) Application without source files or elsewhere on the network.

Either no source files are required for this application, or the application exists at a separate UNC path (e.g. a DFS share).

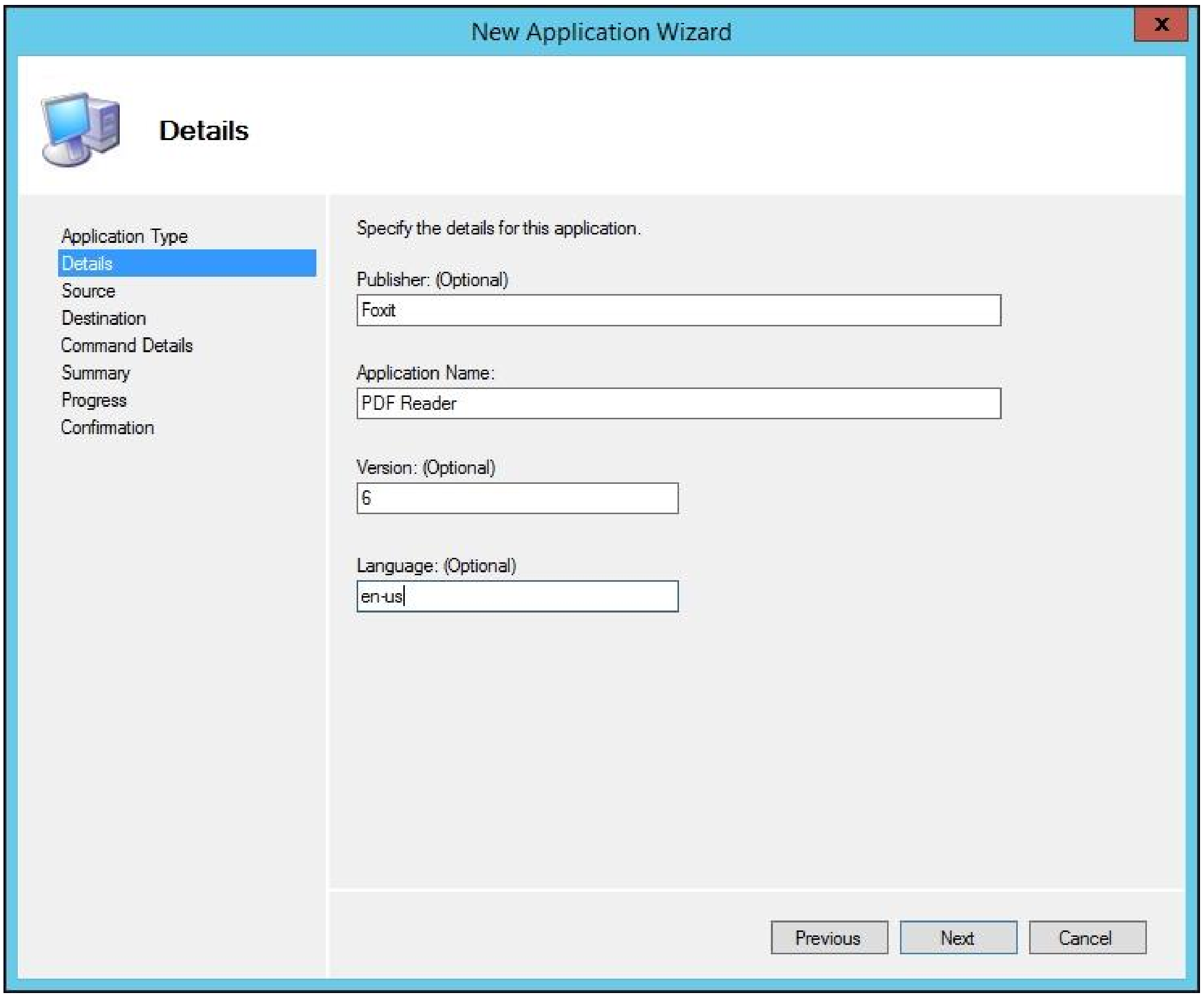
( ) Application bundle.

Create a new application bundle. There is no installation command associated with this application. Instead, only the dependencies of this application will be installed. These dependencies can be configured after the item has been added.

In this particular case, we'll select the first option, as we have the source files (`.msi`). If one had a lot of packaged applications on a network **Distributed File System** (**DFS**) or filer scenario, the second option is perfectly viable as well.

Specifying application details

Once we've selected the option that best suits our application, click on **Next**. In the following window, we will fill in the form to specify our application's details:



Specify the details for this application.

Publisher: (Optional)
Foxit

Application Name:
PDF Reader

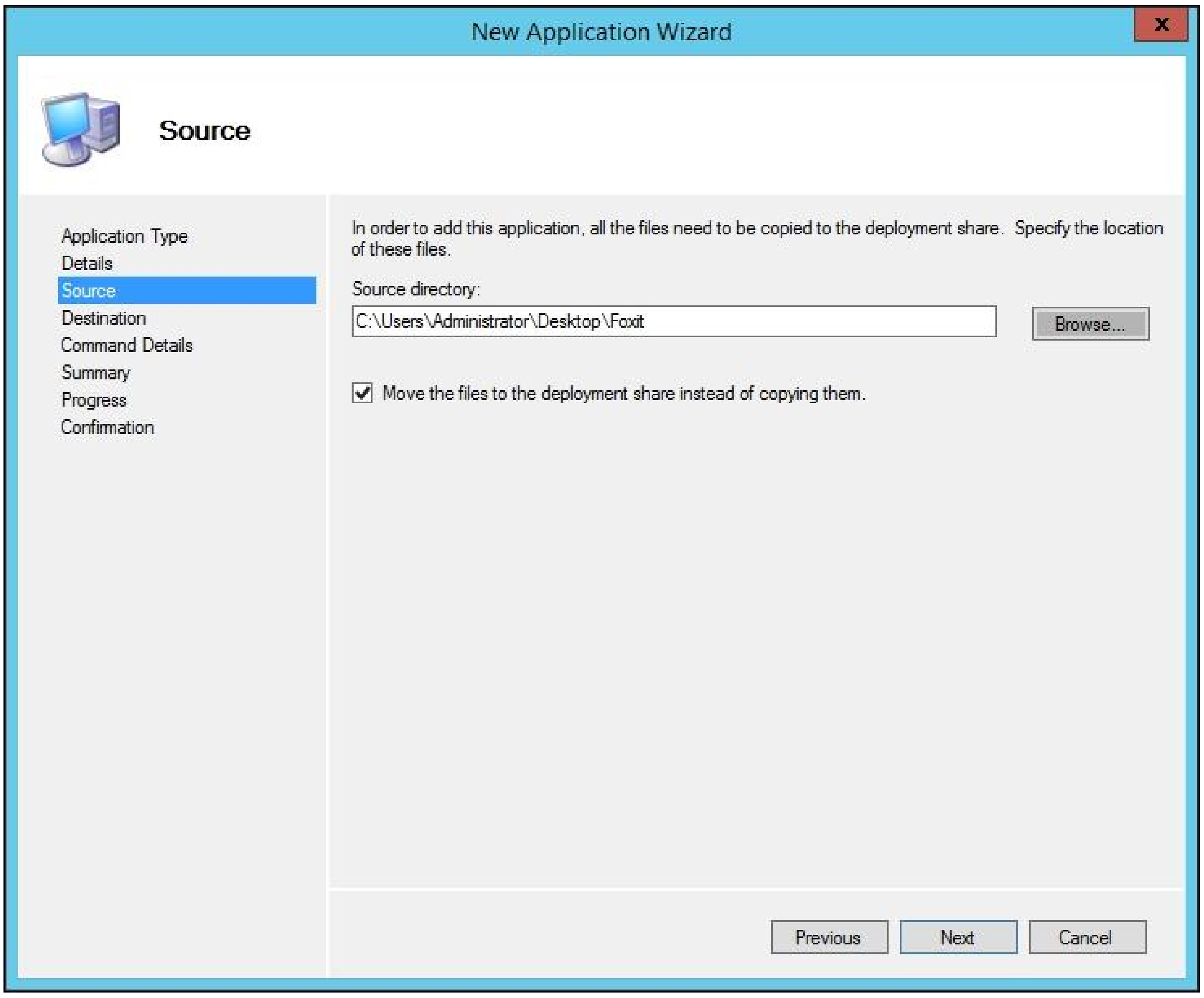
Version: (Optional)
6

Language: (Optional)
en-us

Once you're done, click on **Next**.

Finding the .msi source directory

On the **Source** page, we simply browse to the unique directory that we created previously for our `.msi` file. Since I have no real need for the `.msi` file after this action, I will check the box to **Move the files to the deployment share instead of copying them**, as shown in the following screenshot:



In order to add this application, all the files need to be copied to the deployment share. Specify the location of these files.

Source directory: C:\Users\Administrator\Desktop\Foxit

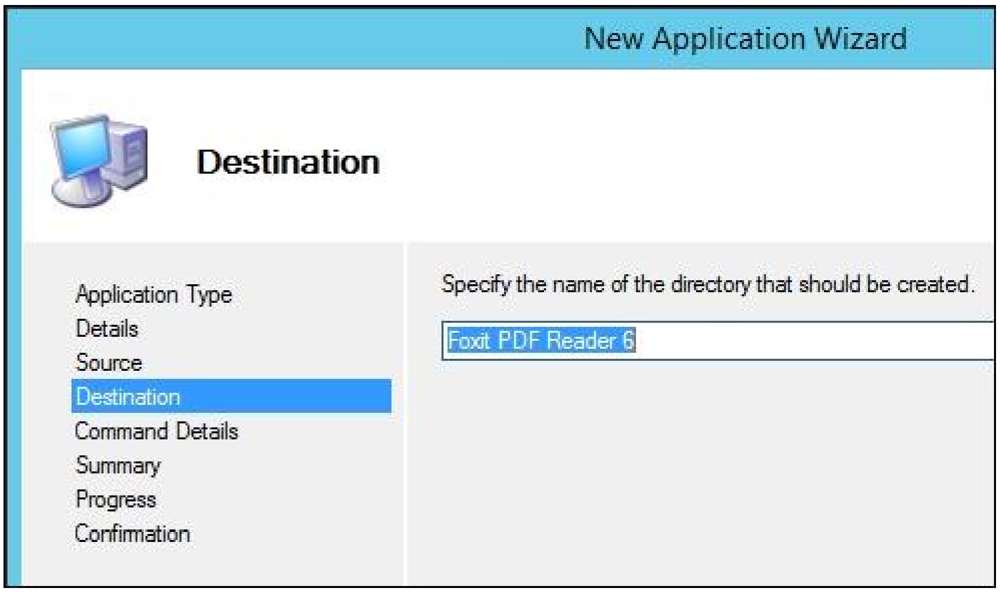
[✓] Move the files to the deployment share instead of copying them.

TIP

In an enterprise environment, it may be better to leave this unchecked so that you have a library of the `.msi` files that you have used in the past.

Specifying the destination directory

In the following screen, the directory to be created to move the `.msi` files to is autogenerated based on the information we previously provided:

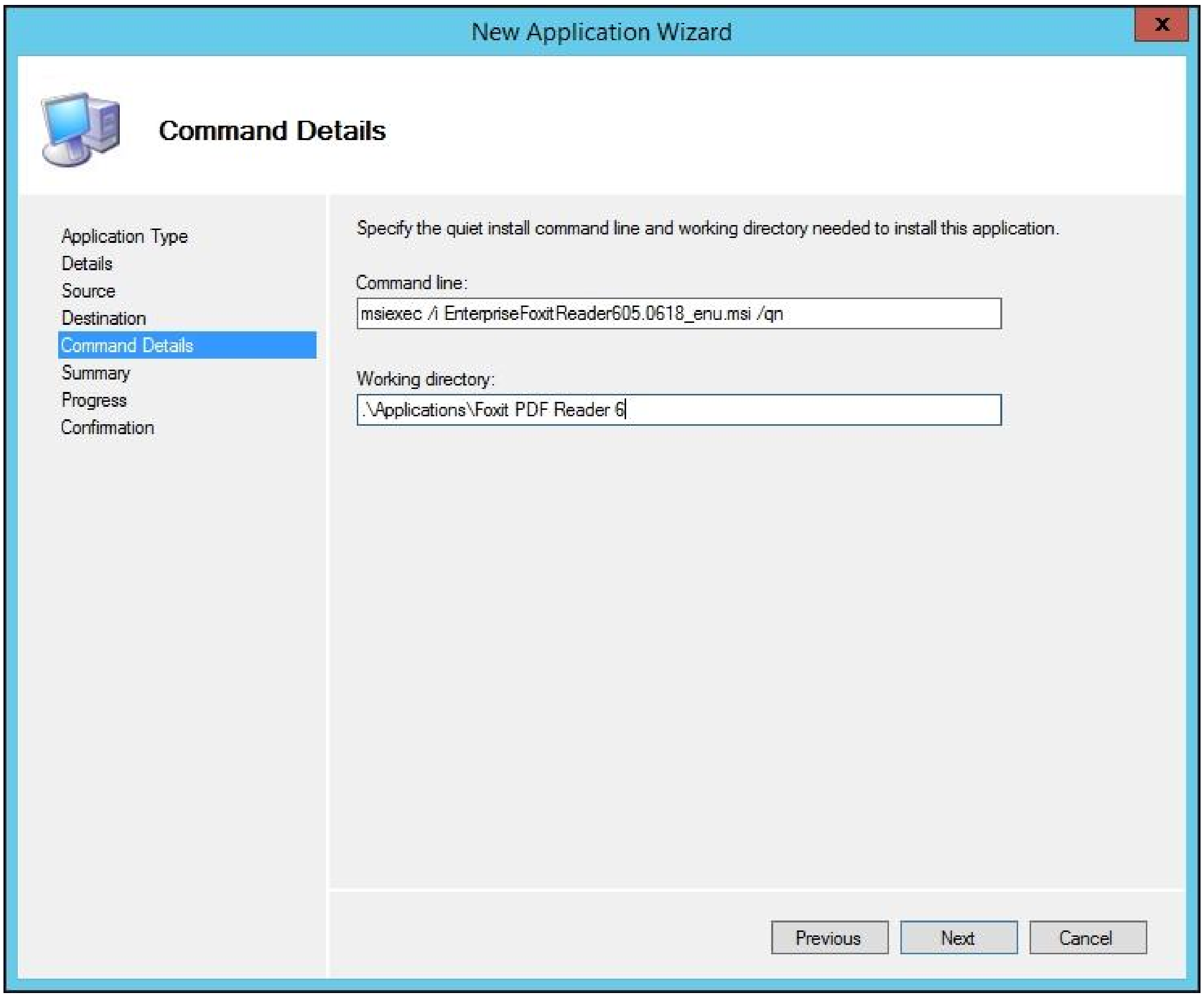


Specify the name of the directory that should be created.

Foxit PDF Reader 6

Entering command details

The **Command Details** screen is where the magic happens. We enter the commands to automatically install the application here. For example, Foxit silent install will be configured as `msiexec /i EnterpriseFoxitReader605.0618_enu.msi /qn`:



Specify the quiet install command line and working directory needed to install this application.

Command line:

```
msiexec /i EnterpriseFoxitReader605.0618_enu.msi /qn
```

Working directory:

```
.\Applications\Foxit PDF Reader 6
```

Click on **Next**, confirm whether your settings are correct, and away we go. The application will be created, and then we can use it in a task sequence to essentially embed the application to Windows at deployment time.

I will repeat the process we went through just now for the other two applications (Java and 7-Zip), and now it's time to proceed with creating a task sequence to capture this reference image.

NOTE

Windows 10

All the concepts shown in this chapter are still valid for Windows 10 image creation. In addition the following information is relevant to Windows 10 specifically:

- Windows 10 enables sideloading for enterprises in a managable fashion. Therefore you can unlock a device for sideloading using an enterprise policy, or through the settings of the machine. License keys are not required for Windows 10 application sideloading and domain join is not required for sideloading to work properly anymore.

- Windows 10 as guest VM is only supported on Hyper-V on Server 2012 R2 or newer or Hyper-V on Windows 10. You will need minimum MDT 2013 Update 1. I recommend using newest available MDT (at time of printing this was MDT 2013 Update 2). Please use Windows 10 ADK version 1507 or newer.