

Guide to Windows Unattended Installation and Windows Image Management

By Hannes Sehestedt
X64 Only Edition
Last Update: January 21, 2024

Contents

[About This Document](#)

[Part 1 - Deploying Windows with Unattended Setup and Creating Bootable Media](#)

[Creating Media for Windows Automated, Unattended Installation Overview](#)

[Things to Know Before You Start](#)

[Start of Detailed Unattended Setup Procedure](#)

[Preparation of Technician System](#)

[Preparation the Reference System](#)

[Workaround for Hyper-V](#)

[Creating the autounattend.xml Answer File](#)

[Bypass Windows 11 Requirements](#)

[Finalizing the Answer File \(autounattend.xml\)](#)

[Create a 2nd Answer File for OOBE \(unattend.xml\)](#)

[Optional - Set Password for Built-in Administrator Account](#)

[Optional - Create Domain Accounts](#)

[Create Local Accounts](#)

[Optional: Automating the First-Time Logon Setup](#)

[Finalizing Answer File \(unattend.xml\)](#)

[Customize Your Windows Installation](#)

[Prepare Assets](#)

[Make Assets Available to Reference System](#)

[Deploy the Assets](#)

[Run Sysprep](#)

[Create an Image of the Reference System](#)

[Status Check](#)

[Optional: Automating Actions Upon Completion of Windows Setup](#)

[Methods to Create Bootable Images, DVD, Drives](#)

[Bootable ISO Image](#)

[Multiple Windows Editions in One Image Using Only MS Tools](#)

[Performing an Unattended Setup with the Multi Edition Media](#)

[How Windows Behaves When Booting from a Multi Edition Image](#)

[Automating Installation for a Multi Edition Boot](#)

[Special Considerations for Virtual Machines](#)

[Bootable DVD](#)

[Bootable Flash Drive Using Only Native MS Tools](#)

Part 2 - Managing Offline Windows Images and Live, Online Installations

Updating Windows Image with Latest Windows Updates

Obtain and Organize Updates

Applying the Updates

Summary of the Update Process

Update WinRE.wim

Update WinPE image (boot.wim)

Update the Main OS (install.wim)

Updating the Base Image and Finalizing the Image

Adding or Removing Drivers to an Offline Image

Get Drivers

Download Method

Export Method

Mount Offline Image

Mounting an Image Located on HD, Thumb Drive, Flash

Mounting an Image Located on Virtual HD (VHD / VHDX)

Optional - Checking for Drivers Already Present

Adding or Removing Drivers

Adding Individually Downloaded Drivers

Adding Multiple Drivers by Recursing a Folder

Removing Drivers

Unmount Offline Image

Adding Boot Critical Drivers to the boot.wim

Updating an Online Windows Installation with the Latest Updates

Automating the Installation of Drivers for Online Windows

Appendix A: Notes on Use of Microsoft Tools

Appendix B: Identifying the Disk ID to Which Windows Will be Installed

Appendix C: Using OSCDIMG to Create an ISO Image

Appendix D: Converting INSTALL.ESD files to INSTALL.WIM

Appendix E: How to download Windows with INSTALL.WIM

Appendix F: Performing a Compact OS Install

Appendix G: Creating Partitions That Meet Microsoft Recommendations During Unattended Setup

Appendix H - A Cleaner Windows Installation by Changing UserLocale

Appendix I - Using STARTNET.CMD to script operations at startup

Credits

About This Document

I started this document as notes for myself to help me easily perform various tasks related to Windows unattended installation and injection of updates. Over time, I have cleaned things up and tried to put the pieces into an orderly workflow where applicable. The result is this document.

Information Sources

One of my greatest sources of information are the excellent tutorials from the user Kari on TenForums.com. In particular, the section on unattended setup could not have been completed without his help. I have also borrowed heavily from tutorials by cereberus, and Brink, both from TenForums. User johngalt was also extraordinarily helpful in helping me to devise a solution to creating partitions during unattended setup that follow the latest guidance from Microsoft. Please note that I have made numerous changes and additions to the procedures and information that I found to allow for greater flexibility and range of options.

I encourage you to look at the [credits](#) at the end of this document for a list of resources that I have found to be extremely helpful to me.

Goals

For some operations, such as creating a bootable flash drive, there are third-party utilities available that can perform the task. I aim to show you ways to perform every task using no third-party utilities, often with advantages over the other methods. This will allow the intellectually curious to understand some of the details of what is happening "under the hood". It also makes it easy to script these tasks. For example, pretty much everything in this document has been used by me to write a program that automates each of the tasks presented.

Terminology

Throughout this document, I reference "images", "editions", and "index" or "indices". It's important to understand what each of these is.

Image - There are several types of images, but common to all of them is the fact that they are containers that hold a collection of files, similar to a .ZIP file. A Windows ISO image file is a single file with a .ISO extension that contains all the Windows distribution files. This is how Microsoft distributes copies of Windows from which optical discs, flash drives, and other installation media can be created. ISO images are popular for storing information from optical discs because they can include data such as boot information for these discs. A single image file can contain multiple Windows flavors or "editions" such as Windows 11 Pro, Home, Education, etc.

Within an ISO image file, you will find yet other types of image files with an extension of .WIM (a **W**indows **I**mage file). The largest of these, the INSTALL.WIM file contains the bulk of the Windows installation files.

Some of the processes described in this document require the modification or updating of files within an image. That may require you to perform tasks such as extracting the files from an ISO image to a folder on your HD and then in turn extracting the files from a Windows WIM image to yet another folder on your HD, making the changes needed, then repackaging everything.

Note that there are Windows ISO images that contain a file named `INSTALL.WIM` but others that contain another image type with the name `INSTALL.ESD` (**E**lectronic **S**oftware **D**istribution). In all cases, except as noted in this document, we want to use the ISO image files that hold an `INSTALL.WIM` file. This is necessary because some of the Microsoft tools used to work with Windows images operate only on `.WIM` files, and not `.ESD` files. If you only have access to media with an `INSTALL.ESD` file, you can convert the `INSTALL.ESD` to an `INSTALL.WIM` as described here: [Appendix D: Converting INSTALL.ESD files to INSTALL.WIM](#). See [Appendix E: How to Download Windows with INSTALL.WIM](#) for the steps to download Windows with a `.WIM` file rather than `.ESD` file.

Edition - The various flavors of Windows (Pro, Home, Education, etc.) are known as editions. An image file (a Windows WIM image and an ISO image file) can contain multiple editions of Windows.

Index - Within Windows Image files (WIM), each Windows edition has an index number associated with it. For example, the Windows 11 Pro edition may be associated with index number 6. When running various commands such as `DISM`, you reference the edition of Windows that the command applies to by using the index number associated with that edition.

A word about "online operations" - This document contains procedures for a couple of "online" operations. While not strictly a part of Windows unattended setup or Windows image management, they are certainly in the spirit of unattended setup. For example, one of these operations will allow for the installation of ALL drivers for a system after the initial installation of Windows. This is the perfect complimentary next step after an initial unattended installation of Windows.

Windows 10 - Working with Windows 11 will in many ways be easier than Windows 10. Windows 11 only supports the x64 architecture (64-bit Windows) while Windows 10 still supports x86 (32-bit Windows) as well. This document omits all the sections that deal with dual architecture images since this no longer applies to Windows 11 and is becoming obsolete since 32-bit versions of Windows 10 are becoming more and more scarce. Everything here applies equally to Windows 10 64-bit editions except the information regarding bypassing Windows 11 system requirements.

ARM Editions - Windows is available in editions for ARM based CPUs. This document is not intended for use with this architecture. It applies only to x64 editions of Windows 10 and 11.

Locale - This document assumes the use of the US English edition of Windows and all settings shown are for that locale. If you are working with Windows for another locale or language, you will need to adjust those language specific settings accordingly.

Part 1 - Deploying Windows with Unattended Setup and Creating Bootable Media

```
*****
* Creating Media for Windows Automated, Unattended Installation *
*****
```

```
*****
* Overview of the Process *
*****
```

There are two main types of unattended setup.

In the first type, you create an "answer file" called **autounattend.xml** that supplies all the information that you would normally provide manually to setup during installation. This file is placed on the root of the Windows distribution media or ISO image. When setup starts, it sees this file and setup proceeds completely automatically with no user intervention required.

This type of unattended setup will install Windows with the options you provide in the answer file. You can also optionally inject drivers specific to your system as well as Windows updates into your image so that a new installation of Windows will already have all your drivers and the latest Windows updates right from the start.

In the second type of unattended setup, two answer files are created. In addition to the **autounattend.xml** answer file, an **unattend.xml** answer file is also created. You will need to create a **reference system** with a new installation of Windows which you will customize and then use to create a new, customized Windows image. The first system, the **technician system**, will have nothing destructive done to it so it's perfectly fine to use your primary system for this. The reference system can be a physical system or a VM that will have a clean install of Windows applied to it.

The advantage with the second method is that you can make almost any modifications that you wish. You can apply desktop wallpaper, set your preferences in File Explorer, customize other Windows settings, install drivers, and even install applications and utilities. Note that UWP apps cannot be pre-installed. These will need to be installed after installation.

A procedure is then followed to prepare the reference system for imaging and the final image incorporating all the changes is created.

That image file (**install.wim**) is then used to replace the **install.wim** file on a standard Windows distribution and the whole thing is then repackaged into a new ISO image. From that ISO image file, installation media such as DVDs and flash drives can be created.

OPTIONAL: For either type of setup, you can inject drivers into the image so that after installation Windows, all system drivers are already present and installed.

TIP: Since drivers can be updated frequently, my personal preference is to NOT inject drivers into an image. Instead, I find it easier to keep a copy of all drivers for the system and to install these after the initial Windows setup. There is an extremely simple method of installing all drivers with a single operation that is extremely simple. This also eliminates the need to maintain multiple such images for all your systems. We will discuss this further later on.

* Things to Know Before You Start *

Before starting your project, please carefully read [Appendix A: Notes on Use of Microsoft Tools](#).

The detailed procedure below will guide you through the configuration of each type of installation discussed earlier. The detailed steps will guide you through the available options.

It is important to understand that the **unattend.xml** answer file is only needed if you are going to create a reference system and a sysprep image to customize your Windows installation. If your goal is only to create a single answer file to perform unattended installation of Windows, then you only need to set up the technician system (no reference system is needed) and you will only be creating a single answer file (**autounattend.xml**).

Note that there is a third option available: You could create customized install media, and simply skip the steps for creating the first answer file (the **autounattend.xml** file). This will result in Windows media that will install your customized version of Windows, but you will still need to manually supply information during setup such as the language information, where to install, username, etc.

Don't get hung up on these details now. The detailed procedure will guide you through all of this. For now, just determine which of these options you want:

- Unattended installation of Windows
- Unattended installation of a fully customized Windows
- Installation of customized Windows but **WITHOUT** unattended installation

With the first option, Windows is not customized but the installation can be entirely automated with one or two possible exceptions, depending upon your choices. The only customization is that you will be able to set language selections, time zone, the username to be created, and a few other simple settings. The second and third options allow you to customize almost everything about your installation but will take a lot more work up front since a separate reference system will need to be created.

Be aware that this documentation will ask you to run various commands such as **DISM**, **ImageX**, or **OSCDIMG**. My suggestion would be to open the command prompt by running the "**Deployment and Imaging Tools Environment**" in elevated mode (as Administrator). As noted in [Appendix A](#), this will ensure that these tools are in the path and readily accessible.

Note that the Deployment and Imaging Tools Environment is installed when the Windows ADK is installed (more on the ADK coming up shortly). You will find this item in **All Apps > Windows Kits** after you install the Windows ADK.

TIP: When you open the Deployment and Imaging Tools Environment, you can issue a "**CD **" to change paths to the root of the volume. This makes the annoyingly long prompt at the command line much shorter and less obtrusive.

* Start of Detailed Procedure *

Two systems are needed unless you are creating only a single answer file for unattended setup: the **technician system** and the **reference system**. The technician system can be your everyday system as nothing destructive will be done to it.

Since the technician system is required in all scenarios, we will begin by setting up the technician system as instructed below.

* Preparation of the Technician System *

You can use a physical system or a VM. Nothing is done on this system that will affect normal operation, so you can do this on your primary everyday machine.

First, install the Windows ADK (Windows Assessment and Deployment Kit) on the technician system. When installing the ADK, you can choose to install only the **Deployment Tools** if you wish. The ADK can be downloaded and installed from here:

<https://docs.microsoft.com/en-us/windows-hardware/get-started/adk-install>

NOTE: An even easier way to install the ADK is to open an elevated command prompt, and then simply run this command:

```
winget install --id Microsoft.WindowsADK --override "/q /features  
OptionID.DeploymentTools /CEIP off /norestart"
```

NOTE: At the time of this writing, winget is not installing the latest version of the ADK. Check the version by doing a **winget search windowsadk** first.

Install the ADK now. Again, when presented with the list of components to install, you need only select the **Deployment Tools**.

Next, mount a Windows ISO image by double-clicking the ISO image file to mount it.

Copy all files from that image to a folder on a hard disk or SSD. In this example I will assume **C:\ISO_Files**. You can then dismount the ISO image by right clicking the mounted drive in **File Explorer** and choosing **Eject**.

NOTE: You must use an HDD or SSD, **NOT** a removable device like a thumb drive. An external HDD or SSD is okay, but not a flash drive that presents itself as removable media. This is due to limitations of some Microsoft utilities.

Start the **Windows System Image Manager (SIM)** app (part of the ADK you just installed and found under **All apps > Windows Kits**).

Select **File > New Answer File**.

If asked if you want to open a Windows image, choose **yes** and point it to **C:\ISO_Files\sources\install.wim**. You can point to a catalog file if you have one instead (see the note below).

If your image file holds multiple Windows editions, select an edition such as Home, Pro, etc. when asked to do so. A catalog file will be created.

NOTE: You can save the catalog file for future use. It will be in the sources folder and is named **install_Windows 10 XXX.clg**, where **XXX** is the edition in question, for example, "PRO". Once the catalog file is created, in the future, rather than opening install.wim, you can point to the catalog file. This can save several minutes since creating the catalog file takes a while.

NOTE: The catalog file name will reference Windows 10, even for Windows 11.

TIP: If all you plan to do is create an **autounattend.xml** and / or an **unattend.xml** answer file, but you don't currently plan to add it to a Windows image, then if you have a catalog file available, you can skip the step above to copy the files from a Windows image to a folder such as **C:\ISO_Files**. Don't worry about that too much for now. It's just a handy piece of information to know when you get more familiar with this process.

If you are creating only a single answer file (**autounattend.xml**), and not capturing a customized installation of Windows, then you will not need a reference system so you can now skip to the section called [Creating the autounattend.xml Answer File](#). Otherwise, continue with [Preparation of the Reference System](#) below.

```
*****
* End of Preparation of the Technician System *
*****
```

```
*****
* Preparation of the Reference System *
*****
```

NOTE: Read this entire section up to "Creating the autounattend.xml Answer File" before you proceed as there are some issues to be aware of before you begin.

IMPORTANT: Later, there will come a point where you need to boot your system from Windows PE or Windows Installation media. Figure out how to boot this media **NOW!** Failing this will ruin the work you have done, and you will need to set up the reference system all over again! You can use a VM for the reference machine if you want. In fact, an advantage of using a VM is that you can create a snapshot (term used by Vmware) or a checkpoint (term used by Hyper-V) before the critical reboot. If you miss the boot and the system boots from the HDD, you can revert to the snapshot / checkpoint so that you can try again.

NOTE: If you need to make bootable media, see [Create a Bootable DVD](#), then come back here.

Start installation **MAKING SURE THAT NETWORKING IS DISABLED** (explanation below). You can simply unplug your network cable if your system uses Ethernet.

When you get to the point in the installation where you are asked for a product key, choose **I don't have a product key**. Proceed up to the point where you are asked to choose your region.

NOTE: This is the screen that is displayed after files are copied to the HD and after the system reboots to continue installation, not the region selection screen near the very start of the installation process. Press **CTRL + SHIFT + F3**. The system will reboot. It will come back up in Audit Mode with a sysprep dialog box. **Click cancel. DO NOT click on OK!**

Now you can make changes to personalization and other Windows settings. Only after completing those changes, enable the network and install Windows updates if you wish.

IMPORTANT: MAKE SURE that you make any personalization settings changes **BEFORE** the network is enabled. After enabling networking, personalization settings cannot be made until Windows is activated.

NOTE: Personalization setting changes can't be done with Hyper-V. If you want to perform personalization for a reference system on Hyper-V, you have a few options:

- Activate Windows in Audit mode by allowing access to the Internet.
- Import a themepack which will allow limited personalization.
- Perform the following procedure as a workaround:

```
*****  
* Start of Workaround Procedure *  
*****
```

Overview:

Create a Virtual Hard Disk (VHD) and deploy Windows to it. Boot the physical machine to that VHD and setup Windows. Perform your personalization tasks. Optionally, create a Hyper-V VM and configure the VHD to boot from that VM.

Detailed Steps:

- 1) Create a VHD

NOTE: The commands below assume you are creating the VHD on F: called **WIN11PRO.vhdx** with a size of 100 GB, a recovery partition of 500MB, and drive letter W:. Modify the commands to fit your needs.

```
diskpart  
create vdisk file=F:\W11PRO.vhdx maximum=102400 type=expandable  
attach vdisk  
convert gpt  
create partition efi size=260  
format quick fs=fat32 label="System"  
create partition msr size=128  
create partition primary  
shrink minimum=5000  
format quick fs=ntfs label="Windows"  
assign letter="W"  
create partition primary  
format quick fs=ntfs label="WinRE"  
set id="de94bba4-06d1-4d40-a16a-bfd50179d6ac"  
gpt attributes=0x8000000000000001  
exit
```

IMPORTANT: Due to FAT32 limitations, the minimum FAT32 partition size is 260 MB on Advanced Format 4K Native drives. While this can be set to only 100 MB, I always use a size of 260 MB to ensure that it works on any drive.

For the line "shrink minimum=500", this is for the recovery partition. It is okay to make this larger than 500MB. I like to use 2048 (2GB).

If C: is BitLocker encrypted, this is okay. However, you should **NOT** install to any other drive that is BitLocker encrypted because you will then need to supply the recovery key each time you boot.

END IMPORTANT NOTES

2) Run the following commands to apply the image to the VHD:

NOTE: If **C:** (the drive on which Windows is installed) is BitLocker encrypted, suspend BitLocker from the GUI or run **manage-bde -protectors -disable C:** before you run the **bcdboot** and **bcdedit** commands below to avoid having to enter your BitLocker recovery key at start. BitLocker will automatically re-enable after the next boot.

Make sure to use the index number associated with the edition of Windows you want to install. To get a list of Windows editions on your media and the associated index number, run this command. Note that the path references the location of the install.wim file located in your mounted Windows image or a folder on your HDD:

```
dism /Get-WimInfo /WimFile:I:\Sources\install.wim
```

Now, run these commands. The first line is long and wraps in this document. It ends with **/applydir:W:**. You can replace **W:** with another drive letter.

```
dism /apply-image /imagefile:I:\sources\install.wim /index:6  
/applydir:W:\  
bcdboot W:\Windows  
bcdedit /set {default} description "Win11 PRO (VHDX)"
```

Note that in the above commands we are making the virtual disk the default boot option. You can change this later by booting into your original Windows installation and running **MSCONFIG** to change the default OS, but I suggest leaving the virtual disk as the default now since you will perform a number of reboots initially and will want to go into that instance of Windows a number of times to set it up.

When you boot this instance of Windows, when you reach the screen where you are asked for region information, press **CTRL + SHIFT + F3** to reboot and enter into **Audit Mode**. Once booted **CANCEL** the sysprep dialog box. Each time you boot you should cancel this dialog. Also, make certain that you **disable any Ethernet connection BEFORE you boot into this instance of Windows**. If the system finds an Internet connection, you will lose the ability to perform personalization tasks **UNLESS** the system you are installing on has the same edition of Windows activated on it. In that case, the instance of Windows in the VHD will also automatically activate. You may have to check for updates to trigger the activation.

3) If **C:** (the Windows boot disk) is BitLocker encrypted you should first suspend BitLocker from the GUI or run **manage-bde -protectors -disable C:** first before you change the default OS or delete the instance of Windows on the virtual disk to prevent BitLocker from asking for the recovery key. Delete the boot entry for the VHD by running **MSCONFIG**, go to the Boot tab, set the default OS back to your primary Windows installation on C:, delete the entry for the VHD boot, and optionally delete the VHD.

4) You can change the order of boot entries in the boot menu from an elevated command prompt by running the commands below. Again, you should suspend BitLocker first as noted above if it is enabled on C:.

Run **bcdedit**

Note the displayorder under the "Windows Boot Manager" section.

Change the order like this:

```
bcdedit /displayorder {identifier_1} {identifier_2} ... {identifier_N}
```

Example:

```
bcdedit /displayorder {5cb10d44-20ee-11ea-85c6-e6elf64324aa} {8ad10c22-19cc-11ab-85c6-e6elf64324aa} {current}
```

You can also do this:

```
bcdedit /displayorder {identifier} /addfirst.
```

Example:

```
bcdedit /displayorder {current} /addfirst
```

Also:

```
bcdedit /displayorder {identifier} /addlast.
```

Example:

```
bcdedit /displayorder {current} /addlast
```

5) If you want to attach the VHD to Hyper-V and boot from it, you must create boot files on the VHD disk like this:

Be aware that once you connect to virtual disk to a VM and boot into Windows, you can't perform personalization tasks unless you activate Windows.

- Create a new VM but do not install Windows to it.
- When creating the VM, don't create a new virtual disk. Instead, tell it to use an existing virtual disk and point it to the virtual disk you created in the steps above.
- Boot the new VM from Windows install media (an ISO image)
- Run **diskpart**.
- Within diskpart, run **list vol**.
- Note the drive letter for the Windows partition (for example, **C:**).
- **exit** from diskpart using the **exit** command.
- run this command: **C:\Windows\System32\bcdboot C:\Windows** - This assumes that the **list vol** command showed that Windows is on **C:**.

```
*****
* End of Workaround Procedure *
*****
```

You can reboot your reference system as often as needed. Just **cancel** the sysprep dialog each time after you reboot.

TIP: Perform changes to Windows settings first since that must happen before the network is enabled, then enable the network, then install any updates you wish. If working with a VM, you may want to create a snapshot / checkpoint as noted below.

```
*****
* Take a snapshot if Reference System is a VM *
*****
```

If your reference system is a VM, shut it down and take a snapshot / checkpoint.

```
*****
* End of Procedure for Setting Up Reference System *
*****
```

```
*****
* Creating the autounattend.xml Answer File *
*****
```

If you **DO** want to create a customized Windows image but you **DO NOT** want an unattended setup of Windows, then please skip to the section [Create a 2nd Answer File for OOB \(unattend.xml\)](#). The result is that you will have a customized Windows installation but without unattended setup. To clarify: Doing this will allow you to customize Windows as you like. You can modify personalization settings in Windows, change other settings in Windows such as File Explorer settings, browser settings, etc. You can even install apps to be included in your custom image. However, during the installation of Windows, you will be asked for the location to which Windows will be installed, your location, time zone, etc.

If you **DO** want the installation of Windows to be automated, please continue with the steps below.

In the Windows Image pane of Windows SIM (System Image Manager), expand **Components**.

For x64 systems, the component names will start with **amd64_Microsoft-Windows** and end with the build number for the **install.wim** used to create the catalog. For x86 systems the names will start with x86. Since we are working only with x64 editions of Windows, select only the names that start with **amd64**. I'll omit these parts of the names to keep them shorter. Make certain to select the blocks that start with **amd64** and NOT something else such as **x86** or **wow64**.

Right click on **International-Core-WinPE** and add it to **Pass 1 windowsPE**.

IMPORTANT: There is also an **International-Core** without the **WinPE** at the end. Make sure to select the one **WITH WinPE** at the end.

Click on this component in the Answer File pane and set the following values:

InputLocale:	en-US
SystemLocale:	en-US
UILanguage:	en-US
UILanguageFallback:	en-US
UserLocale:	en-US

NOTES:

- 1) **UILanguageFallback** can be omitted if the other settings are **en-US**.
- 2) If you set UserLocale to en-001, this would cause all the Microsoft Store apps such as Spotify, Messenger, Amazon Prime, etc. to not get installed. Please see [Appendix H - A Cleaner Windows Installation by Changing UserLocale](#) for details.
- 3) If you want to see what the locale settings are on a system, open an elevated PowerShell and run this command:

```
dism /online /get-intl
```

Please also note that locales can be specified using Hex codes. For example, en-US could be specified as 0409:00000409. A full list of these codes can be found here:

<https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/available-language-packs-for-windows>

Select **International-Core-WinPE > SetupUILanguage** in the Answer File pane, and set the following value:

UILanguage: en-US

In the Windows Image pane, select **Setup** and add it to **Pass 1 windowsPE**.

Expand it and select **UserData** in the Answer File pane. Set the following values:

AcceptEula: True
FullName: Optional - Set to anything
Organization: Optional - Set to anything

Expand **UserData**, select **ProductKey** and set **Key** to one of the following generic installation Product Keys:

IMPORTANT: These Generic product keys apply to both Windows 10 and 11.

Windows 11 Home Single Language: BT79Q-G7N6G-PGBYW-4YWX6-6F4BT
Windows 11 Home: YTMG3-N6DKC-DKB77-7M9GH-8H VX7
Windows 11 Pro: VK7JG-NPHTM-C97JM-9MPGT-3V66T

For other keys matching other Windows editions, see this article:

<https://www.tenforums.com/tutorials/95922-generic-product-keys-install-windows-10-editions.html>

IMPORTANT: The following section is used to configure the disk to which Windows will be installed. You can omit this entire section if you would like to have Windows setup ask where Windows should be installed.

There are a few reasons why you might want to omit the disk configuration settings in your **autounattend.xml** answer file. First, by doing so, you can use the same answer file on systems with different disk configurations. It also allows the same answer file to be used on both BIOS and UEFI based systems. Note that skipping this section will cause setup to pause right near the very beginning and to ask you where Windows should be installed. After this, the rest of setup will proceed unattended.

There is one more advantage to skipping this disk configuration section:

The latest guidance from Microsoft suggests that the recovery tools partition be created last. When the disk is configured using unattended installation, the recovery tools partition is NOT created last. This can potentially result in multiple recovery tools partitions being created over time when new feature updates are installed. Normally, this is not something to be concerned about because these are small partitions. However, if you do skip this section, the partitions created will adhere to the Microsoft guidance but at the expense of pausing setup so that you can manually choose the location to which Windows should be installed. Since this is at the start of setup, this should not be much of an inconvenience if you decide to implement it.

Please note that there are two other methods that you can use to properly create the partitions in a manner that meets the Microsoft recommendations. Please see [Appendix G](#) for details if you are interested in this.

If you decide to implement this (skipping the disk configuration section), or if you elect to follow the steps in [Appendix G](#), please jump to [End Disk Configuration](#) now, otherwise proceed with the steps below.

For **BIOS** based systems you should configure your disk for **MBR**. For a **UEFI** based system, configure your disk for **GPT**.

NOTE: For Windows 11 deployments, you should always use **GPT**. The one exception would be if you are installing on unsupported hardware, using registry keys to bypass the normal hardware requirements as will be explained later.

Right click **Setup** > **DiskConfiguration** in the Answer File pane (in Pass 1 windowsPE) and select **Insert New Disk**.

Select the disk that you just created and set the following values:

```
DiskID:          0
WillWipeDisk:    true
```

IMPORTANT: Before you continue, it is important that you make sure you are selecting the correct Disk ID since the contents of this disk will be lost. In the sample above, I am using an ID of 0. You should always double-check this before booting from unattended install media first. We describe how to do this in [Appendix B: Identifying the Disk ID to Which Windows Will be Installed](#). If you need to change the disk ID number later, you can simply open the answer file in notepad, make the change to the disk ID, then save the file.

Expand **Disk**, right click on **Create Partitions**, select **Insert New CreatePartition**

For **MBR** partitions used with **BIOS** based systems, add **1 more CreatePartition**.
For **GPT** partitions used with **UEFI** based systems, add **3 more CreatePartitions**.

One by one, select a **CreatePartition** item and configure as follows:

NOTE: The **Partition** column in the table below is not a setting that you specify, it's just for reference here to describe the purpose of the partition. You will be supplying values for **EXTEND**, **ORDER**, **SIZE**, and **TYPE** only.

For MBR / BIOS systems set the following values:

PARTITION	EXTEND	ORDER	SIZE (MB)	TYPE
System Reserved	False	1	500	Primary
Windows	True	2	Leave empty	Primary

For GPT / UEFI systems set the following values:

PARTITION	EXTEND	ORDER	SIZE (MB)	TYPE
WinRE	False	1	500	Primary
EFI	False	2	260	EFI
MSR	False	3	128	MSR
Windows	True	4	Leave empty	Primary

IMPORTANT: The minimum FAT32 partition size is 260 MB on Advanced Format 4K Native drives so using 260 MB will allow this to work on all drives. Otherwise, 100 MB is an acceptable size. I always use 260 MB just to be safe.

Continue here for both MBR and GPT systems

Right-click on **Setup > DiskConfiguration > Disk > ModifyPartitions** in the Answer File pane and select **Insert New ModifyPartition**. Create one of these for each partition (2 for MBR/BIOS or 4 for GPT/UEFI).

Set values as follows:

IMPORTANT: DO NOT modify values not specified here. Leave other values alone.

For MBR / BIOS partitions:

- ModifyPartition 1 (System Reserved):
 - Active = True
 - Format = NTFS
 - Label = System
 - Order = 1
 - PartitionID = 1
- ModifyPartition 2 (Windows):
 - Format = NTFS
 - Label = Windows
 - Letter = C
 - Order = 2
 - PartitionID = 2

For GPT / UEFI partitions:

- ModifyPartition 1 (WinRE):
 - Format = NTFS
 - Label = WinRE
 - Order = 1
 - PartitionID = 1
 - TypeID = DE94BBA4-06D1-4D40-A16A-BFD50179D6AC
- ModifyPartition 2 (EFI):
 - Format = FAT32
 - Label = System
 - Order = 2
 - PartitionID = 2
- ModifyPartition 3 (MSR):
 - Order = 3
 - PartitionID = 3
- ModifyPartition 4 (Windows):
 - Format = NTFS
 - Label = Windows
 - Letter = C
 - Order = 4
 - PartitionID = 4

For MBR disk only:

Expand **Setup** > **ImageInstall** > **OSImage** component in Answer File pane, select **InstallTo**, and set these values:

```
DiskID:          0 (Use same Disk ID you used earlier)
PartitionID:     2
```

This will tell Windows setup to install Windows on partition 2 of disk 0.

For GPT disk only:

Expand **Setup** > **ImageInstall** > **OSImage** component in **pass 1 windowsPE** within the Answer File pane, select **InstallTo**, and set these values:

```
DiskID:          0 (Use same Disk ID you used earlier)
PartitionID:     4
```

This will tell Windows setup to install Windows on partition 4 of disk 0.

```
*****
* End Disk Configuration *
*****
```

Do **only one** of these:

1) If you skipped the disk configuration section or if you followed the steps in [Appendix G](#) as an alternate method of preparing the disk for Windows then do this:

Expand **Setup** > **ImageInstall** > **OSImage** in the Answer File pane and select **InstallFrom**.

Set **path** to **\install.wim**

If the block called **InstallTo** is present under **ImageInstall**, delete it.

2) If you are using the **DiskConfiguration** section (you did **NOT** skip that section), please expand **Setup** in the Answer File pane and remove the **InstallFrom** block.

```
*****
* Optional - Bypass Windows 11 Requirements *
*****
```

If your answer file will be used to install Windows 11 on systems that do not meet the Windows 11 requirements, you can bypass the requirement checks by adding the following entries.

NOTE: It's okay to add these settings even if you use hardware that meets Windows 11 requirements and in an answer file used with Windows 10, it just won't have any effect for Windows 10 or systems meeting requirements. By adding these settings, the same answer file can be used for many different deployments including Windows 10, and Windows 11 systems that meet or do not meet Windows 11 requirements. I prefer to include these settings because I can then use the same answer file for Windows 10 and 11 and for systems that do or do not meet Windows 11 requirements.

In the Windows Image pane expand **Setup > RunSynchronous**. Right click on **RunSynchronousCommand** and add to **Pass 1 windowsPE**.

Add another 3 **RunSynchronousCommand** entries (for a total of 4).

In the Answer File pane, select the **Setup > RunSynchronous > RunSynchronousCommand** entries one at a time and set the following values:

NOTE: The path entries are long, and they may wrap to a second line below. Each entry ends with a "/f".

NOTE: If Order 1 is in use by another setting, start with the next available number and continue incrementing from that number.

```
Order:      1
Path:       reg add HKLM\System\Setup\LabConfig /v BypassTPMCheck /t
reg_dword /d 0x00000001 /f
```

```
Order:      2
Path:       reg add HKLM\System\Setup\LabConfig /v
BypassSecureBootCheck /t reg_dword /d 0x00000001 /f
```

```
Order:      3
Path:       reg add HKLM\System\Setup\LabConfig /v BypassRAMCheck /t
reg_dword /d 0x00000001 /f
```

```
Order:      4
Path:       reg add HKLM\System\Setup\LabConfig /v BypassCPUCheck /t
reg_dword /d 0x00000001 /f
```

```
*****
* End Optional - Bypass Windows 11 Requirements *
*****
```

Choose one of these two options:

- 1) If you are only creating an **autounattend.xml** and **NOT** capturing a sysprep image from a reference system, then skip to the section, [Create a 2nd Answer File for OOBE \(unattend.xml\)](#), but **DO NOT** create a second answer file. Instead, continue adding the items described there to the **autounattend.xml** answer file. The instructions will inform you of any exceptions that you need to be aware of. Just make sure to read all steps carefully. After completing that section, you will be instructed to come back here and follow the instructions below for finalizing the answer file. When you are instructed to return here, skip past item 2 below and go to the [Finalizing the Answer File \(autounattend.xml\)](#) section below.
- 2) If you **ARE** using a reference system to capture a sysprep image, then perform the steps below for finalizing this answer file now, then continue to create the 2nd answer file, (**unattend.xml**), as described in the [Create a 2nd Answer File for OOBE \(unattend.xml\)](#) section below.

```
*****
* Finalizing the Answer File (autounattend.xml)*
*****
```

Before you finalize the answer file, please see [Appendix F](#) to determine if you should configure your answer file to perform a "Compact OS" install.

In the Answer File pane, expand all blocks and locate any unused blocks (the lite colored blocks) and delete them.

Select **Tools > Validate Answer File**. No errors should be displayed.

Save the file as **autounattend.xml** in **C:\ISO_files**.

Please jump to the [Status Check](#) section now.

```
*****
* Create a 2nd Answer File for OOBE (unattend.xml) *
*****
```

The second answer file is for the Out of Box Experience (OOBE). It is used to automate bypassing region selection, keyboard selection, user account creation, etc.

NOTE: You will be placing some components from **Shell-Setup** into both phase 4 and 7. Make sure to observe carefully in the following instructions the correct placement of these components and values.

Open Windows SIM and create a new answer file as you did earlier. You can reuse the same catalog file. **NOTE:** If creating just the one answer file, add the below items to that file (the **autounattend.xml** file you have already been working with).

From the Windows Image pane, add the Following component:

International-Core (NOT **International-Core-WinPE**) to **Pass 7 oobeSystem**.

Expand **Shell-Setup** in the Windows Image pane and add the following components:

Shell-Setup > OEMInformation to **Pass 4 Specialize**

Shell-Setup > OOBE to **Pass 7 oobeSystem**

Shell-Setup > UserAccounts to **Pass 7 oobeSystem**

In the Answer File pane, select **Shell-Setup** in **pass 4 specialize**. Set the following values:

NOTE: Only use **CopyProfile** in the **unattend.xml** answer file. **Do not set this value** if you are only configuring an **autounattend.xml** answer file.

ComputerName:	<If left blank a random name will be assigned>
CopyProfile:	true < See note above!>
OEMName:	Set to anything (optional)
RegisteredOrganization:	Set to anything (optional)
RegisteredOwner:	Set to anything (optional)
TimeZone:	Central Standard Time (See note below)

NOTE: Set **TimeZone** to your time zone. If left blank, **TimeZone** will default to **Pacific Standard Time** for the **en-US** locale. For a list of available time zones, run the command **tzutil /L**.

NOTE: If you specify an asterisk (*) for the ComputerName, then the random name of the computer will be replaced by a string of up to 8 characters maximum that is made up from the **RegisteredOrganization** and **RegisteredOwner** values with the remaining characters being random. You can also specify a name of up to 15 characters in length but be careful to not use the same name on multiple machines. If installing on multiple computers, it is best to leave this field blank and then change the computer name after installation of Windows.

This section and each option are optional: Select **Shell-Setup > OEMInformation** in the Answer file pane to **pass 4 specialize** and set the following values:

Manufacturer:	Set to anything (optional)
Model:	Set to anything (optional)
SupportHours:	Set to anything (optional)
SupportPhone:	Set to anything (optional)
SupportProvider:	set to anything (optional)
SupportURL:	Set to anything (optional)
Logo:	C:\Windows\System32\oemlogo.bmp (optional)

NOTE: If adding **logo** to the **autounattend.xml** answer file rather than the **unattend.xml** answer file, you will need to manually copy the logo file to Windows after installation. This is because you are not creating a reference system to which you can copy this file. The image must be a 120 x 120 BMP image. This will be explained in detail later. For now, just be aware of this, but no action is needed at this time.

Select **International-Core** in **pass 7 oobe system** Answer File pane and set values as follows:

InputLocale:	en-US
SystemLocale:	en-US
UILanguage:	en-US
UILanguageFallback:	en-US
UserLocale:	en-US

NOTE: **UILanguageFallback** can be omitted if other settings are **en-US**.

Select **Shell-Setup** in the Answer File pane in **pass 7 oobeSystem** and set the following values:

RegisteredOrganization: Set to anything (optional)
RegisteredOwner: Set to anything (optional)
TimeZone: Central Standard Time (See note below)

NOTE: Set **TimeZone** to your time zone. If left blank, **TimeZone** will default to **Pacific Standard Time** for the **en-US** locale. For a list of available time zones, run the command **tzutil /L**.

Select **Shell-Setup > OOBE** in **pass 7 oobeSystem** in the Answer File pane and set the following values:

HideEULAPage: True
HideOEMRegistrationScreen: True
HideOnlineAccountScreens: True
HideWirelessSetupInOOBE: True
ProtectYourPC: 1 (See note below)
UnattendEnableRetailDemo: False

ProtectYourPC value can be 1, 2 or 3:

- 1 = Recommended (default) level of protection
- 2 = Only updates are installed.
- 3 = Automatic protection is disabled.

Perform this next step only in the **unattend.xml** if you are creating one. **Do NOT** add this entry to the **autounattend.xml**:

Because we will use the Windows built-in admin account to customize our Windows image, a **C:\Users\Administrator** profile folder will be created and shown to end users although that account itself will be disabled when we sysprep the image. To hide this Administrator folder, expand **Shell-Setup > OOBE** in the Answer File pane in phase **7 oobe System**, and click on **VMModeOptimizations**. Set the value for **SkipAdministratorProfileRemoval** to **False**.

```
*****  
* Optional - Set Password for Built-in Administrator Account *  
*****
```

NOTE: It is not necessary to create a password for the built-in Administrator account since this account will be disabled but you can do so if you wish.

Select **Microsoft-Windows-Shell-Setup > UserAccounts > AdministratorPassword** under **7 OobeSystem** and set a password for the built-in admin account. Be sure to note it somewhere safe and accessible. When the answer file is saved that password will be encrypted and extremely difficult, if not impossible, to recover unless you've recorded it somewhere.

```
*****  
* End of Optional Section to Set Password for Built-in Administrator *  
*****
```

* Optional - Create Domain Accounts *

If you want to add an Active Directory domain and domain account(s), right click **Microsoft-Windows-Shell-Setup > UserAccounts > DomainAccounts** under **7 OobeSystem** and select **Insert New DomainAccountList**.

Select the new **DomainAccountList** and add your local domain by supplying a value for **Domain**.

To add domain users, right-click the new **DomainAccountList**, select **Insert New DomainAccount**. Click the new **DomainAccount**, add the user group and domain account name by supplying values for **Group** and **Name**.

Repeat these steps for each domain account you wish to create.

* End of Optional Section to Create Domain Accounts *

* Create Local Accounts *

NOTE: You should create at least one user in the **Administrators** group. **DO NOT** use the name **Administrator** since a user with that name will already be created but will be disabled.

Expand **Shell-Setup > UserAccounts** in **phase 7 oobe System**, right click **LocalAccounts**, select **Insert New LocalAccount** and set the following values for the **LocalAccount** block that was just created:

Description:	Set to anything (optional)
DisplayName:	(Full name as seen in login screen. Example: John Smith)
Group:	(Enter Administrators or Users)
Name:	(Short Username. Example: JohnS)

Expand the account in the Answer File pane and click on **Password**.

Set **value** to the password you want to assign to this user account. It's okay to leave it blank. The password will be encrypted when you save the answer file.

NOTES:

1) In order to enter a blank password, you need to follow this procedure: Type something, anything, in the password field and hit **Enter**. Then, remove the password that you typed and again hit **Enter**. You will note that the Password block now changes from a light shade to a dark shade indicating that the entry is recognized. If you don't do this and simply leave the field blank, it won't be properly recognized.

2) If you are going to change the password once a password is committed, you will need to remove the **Password** block and then add it again from the **Windows Image** pane. Note that you can determine if a password was previously committed by looking at the property called "**PlainText**". If this is set to "**false**", then a password was previously committed and encrypted, in which case you should delete and re-add the **Password** block.

If you want to add another user account, repeat the above steps. Make sure at least one of these is added to the **Administrators** group.

```
*****
* Optional - Automating the First-Time Logon Setup *
*****
```

NOTE: Add these settings to the **autounattend.xml** answer file if you are creating only an **autounattend.xml** answer file and not creating a sysprep image. If you are creating a sysprep image (and thus, a second answer file named **unattend.xml**), then add these settings to the **unattend.xml** answer file instead.

Using the steps up until this point, Windows installation will proceed to the point where you logon for the first time. During that first logon it takes another several minutes for setup to complete while you get the screens that say things like "Hi", "We're setting everything up for you", "This may take a few minutes", "Almost done", etc.

If you want to automate this, then you can add a onetime autologon by making the following additions to your answer file. If you do not want to do this just skip to [Finalizing the Answer File \(unattend.xml\)](#).

From the Windows Image pane in Windows System Image Manager, expand **Shell-Setup**. Add both **AutoLogon** and **FirstLogonCommands** to **Pass 7 oobe System**.

Select **Shell-Setup** > **AutoLogon** in the Answer File pane and set the following values:

```
Enabled:      true
LogonCount: 1
Username:    <Use an admin username you created>
```

NOTE: **Username** should be what you entered for **Name** when you created the user, **NOT DisplayName**.

Expand **AutoLogon** and select **Password**.

Set **Value** to the password for the username you specified for this user.

NOTES:

1) In order to enter a blank password, you need to follow this procedure: Type something, anything, in the password field and hit **Enter**. Then, remove the password that you typed and again hit **Enter**. You will note that the Password block now changes from a light shade to a dark shade indicating that the entry is recognized. If you don't do this and simply leave the field blank, it won't be properly recognized.

2) If you are going to change the password once a password is committed, you will need to remove the **Password** block and then add it again from the **Windows Image** pane. Note that you can determine if a password was previously committed by looking at the property called "**PlainText**". If this is set to "**false**", then a password was previously committed and encrypted, in which case you should delete and re-add the **Password** block.

Right click on **FirstLogonCommands** in the Answer File pane and select **Insert New SynchronousCommand**.

Select **SynchronousCommand** and set the following two values (Note that the first entry for **CommandLine** is a long line and will wrap in this document. It ends with the **/f**)

```
CommandLine:      reg add "HKLM\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Winlogon" /v AutoLogonCount /t REG_DWORD /d 0 /f
Order:            1
```

NOTE: The entry you just made adds a registry entry because of a bug with how AutoLogon is handled. AutoLogon erroneously adds 1 to LogonCount value. As a result, you must subtract 1 from the number of times you want it to logon. To logon once, you must make the LogonCount 0. Unfortunately, Windows properly understands a LogonCount of 0 as meaning "Never AutoLogon". To correct this, we must make the LogonCount 1 and use the registry entry to correct for this problem.

NOTE: If you are creating a sysprep image, this will also affect your reference system. At the point where you have completed creating the image file on the reference system later in this procedure and you reboot normally, the reference system will logon and complete the setup and auto logoff if that option is chosen. This is expected behavior.

```
*****
* End of Optional Procedure to Automate the First Time Logon Setup *
*****
```

If you are not creating a sysprep image from a reference system and you are only creating the autounattend.xml answer file, jump back to [Finalizing the Answer File \(autounattend.xml\)](#) above since the following steps will not apply to you.

```
*****
* Finalizing the Answer File (unattend.xml)*
*****
```

In the Answer File pane, expand all blocks and locate any unused blocks (the light colored blocks) and delete them.

Select **Tools > Validate Answer File**. No errors should be displayed in the messages pane.

Create a folder on the technician system to store your assets (I'll assume **C:\Assets**) and save your file to this location as **unattend.xml**. The assets are items that you will need access to on the reference system later. Additional assets are discussed in the section below.

```
*****
* Customize Your Windows Installation *
*****
```

On your reference system, customize Windows as you see fit. You can apply changes to Windows settings, install apps, etc. It's fine to reboot as often as needed.

IMPORTANT: If you need to reboot at any time, when the sysprep dialog pops up after reboot, cancel it.

NOTE: After you install any apps, don't run them. This will avoid a situation where they create user specific AppData folders. Also, DO NOT add or remove apps from the Microsoft Store! This will cause an error for Sysprep.

```
*****
* Prepare Assets *
*****
```

Begin by saving the following bulleted items to the **C:\Assets** folder or wherever you saved the **unattend.xml** file:

- The **unattend.xml** file (you should have already saved this).
- Create a text file in that same folder called **RunOnce.bat** with the following contents:

```
if not DEFINED IS_MINIMIZED set IS_MINIMIZED=1 && start "" /min "%~dpnx0" %*
&& exit
echo Y | del %appdata%\microsoft\windows\recent\automaticdestinations\*
rd c:\image
rd c:\scratch
del %0
```

IMPORTANT: Some lines above may wrap due to length. The first line starts with "if" and ends with "exit". The 2nd line starts with "echo" and ends with "*".

NOTE: The first line of the script simply forces the script to run minimized.

The second line of the script works around an issue in Sysprep which causes end users to see the built-in admin's recent files (the stuff we were working with when customizing the Windows image). This resolves that issue.

NOTE: If you are performing a onetime automatic logon as described previously and you want to have the system automatically logoff after Windows completes setup, then change the last line of the **RunOnce.bat** to this:

```
del %0 & logoff
```

DON'T RUN THIS FILE as it will delete itself! It will be used every time a new user signs in for the first time clearing **This PC**, **Quick Access**, and **Recent files** views, then deletes itself. Without it, some leftovers from the reference machine's built-in administrator account would be shown in **Quick Access**.

- Any installers for software that you want to install.
- Any themepack files that you want to include.
- The **oemlogo.bmp** file as described previously (if used). Note that this file must be a 120x120 BMP file.

```
*****
* Make Assets Available to Reference System *
*****
```

Make the assets you saved available to the reference system. For example, you can use a thumb drive, external HD, or an ISO Image. Note that you cannot connect to network shares while in Audit Mode. If your reference system is a VM, you can create an ISO image out of your assets and simply mount it as a virtual optical disk.

TIP: You can create an ISO image of your assets by following these steps:

- Place all your assets into a folder. Subfolders within this folder are fine.
- Open the Deployment and Imaging Tools Environment
- Run the following command:

```
OSCDIMG.EXE -o -m -h -k -u2 -udfver102 -L"Vol_Name" "Source" "Destination"
```

Be aware that there is no space after the "-L".

Vol_Name is the Volume name that would show up when the ISO image is mounted. You can use "" to indicate no volume name or omit the **-L"Vol_Name"** since this is optional.

Source is the entire path to the folder where you saved the assets.

Destination is the full destination path including the filename of the ISO image you are creating.

Example: `OSCDIMG.EXE -o -m -h -k -u2 -udfver102 -L"My Assets" "C:\Asset Files" "%UserProfile%\Desktop\My Assets.ISO"`

END TIP

```
*****
* Deploy the Assets *
*****
```

From your assets, do the following on the reference system:

If you have specified a logo, save **oemlogo.bmp** to **C:\Windows\System32**. The logo must be a 120 x 120 BMP image.

Save the **unattend.xml** answer file to **C:\Windows\System32\Sysprep**

For your Themepacks, there is no need to save them. Just double-click them to install and apply them. Run the themepack that you want to be active on your system last. They will be automatically saved, and the last one run will become the active themepack.

Create two folders on the reference system:

C:\Image
C:\Scratch

Open an elevated command prompt and run this command:

```
cmd.exe /c cleanmgr /sageset:65535 & cleanmgr /sagerun:65535
```

This will open the **Advanced Disk Clean-up** app. Select every item on the list and then click on **OK**. Close the command prompt and proceed when it is done.

IMPORTANT: Do the following step only after you have performed all needed reboots because the **RunOnce** file will run the next time you reboot the system, and we don't want that to happen yet.

Save **RunOnce.bat** to **C:\Users\Administrator\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup**

NOTE: AppData may be hidden depending upon your settings. To navigate to this path, open **File Explorer**, type in **%appdata%** in the address bar and hit **Enter**. You can then navigate the rest of the path.

***** If running on a VM, now would be a good time to take a snapshot / checkpoint just in case you need to roll back *****

```
*****  
* Run Sysprep *  
*****
```

Press **WinLogoKey + R** and run this command:

```
C:\Windows\System32\Sysprep\sysprep.exe /generalize /oobe
```

Sysprep will run and then shut down the system.

```
*****  
* Create an Image of the Reference System *  
*****
```

Read this entire section before starting! It may help you in executing these steps.

Boot the reference machine from WinPE or Windows install media. Make sure to boot media of the same architecture (**x64**) as the reference machine. ***** DO NOT LET IT BOOT FROM THE HD! *****. When booting, you may be asked to press any key to boot from the DVD. **IT WON'T GIVE YOU LONG. BE READY!** Note that you will see this message referencing a DVD even if you boot from a flash drive.

If you miss the boot and Windows boots from the HD, you can go back to your snapshot / checkpoint and try again if you are on a VM. If you are on a physical machine and the system boots from the internal HDD or SSD, you will have to start over!

Tip: If your reference system is running on VMware Workstation, select **VM > Power > Power on to Firmware**. From the firmware menu choose the Virtual CDROM option to boot from. If you are working with a BIOS based VM then the firmware will look different, and you will need to go into the boot menu and set the CD-ROM as the first boot device and save the changes to the BIOS configuration. Make sure you have set the correct ISO image to connect before you boot the VM! You will still need to press a key when prompted, so be ready.

For Hyper-V, open settings for the VM and point the DVD Drive to the appropriate image file. Select **Firmware** and move the DVD Drive to the top of the boot order. You will still need to press a key when prompted, so be ready.

End Tip

If prompted to press any key to boot from DVD **be sure to quickly press a key** because you are given very little time to do so.

Once the process stops at the first static screen (this is the screen where you are asked for language and keyboard information), press **SHIFT + F10** to open a command prompt. Run the following commands:

```
diskpart < Wait for the "diskpart"> prompt before running the next command
list vol
```

Take note of the drive letter assigned to the Windows partition. This will be the large NTFS partition.

Note that it is possible that the Windows partition has no drive letter assigned to it. If no drive letter is assigned, then note the volume number that the Windows partition has. Run the commands below. In this example, I assume that the Windows partition is volume 1:

```
select volume 1
assign
list vol
```

The Windows partition will now have a drive letter. Note that letter.

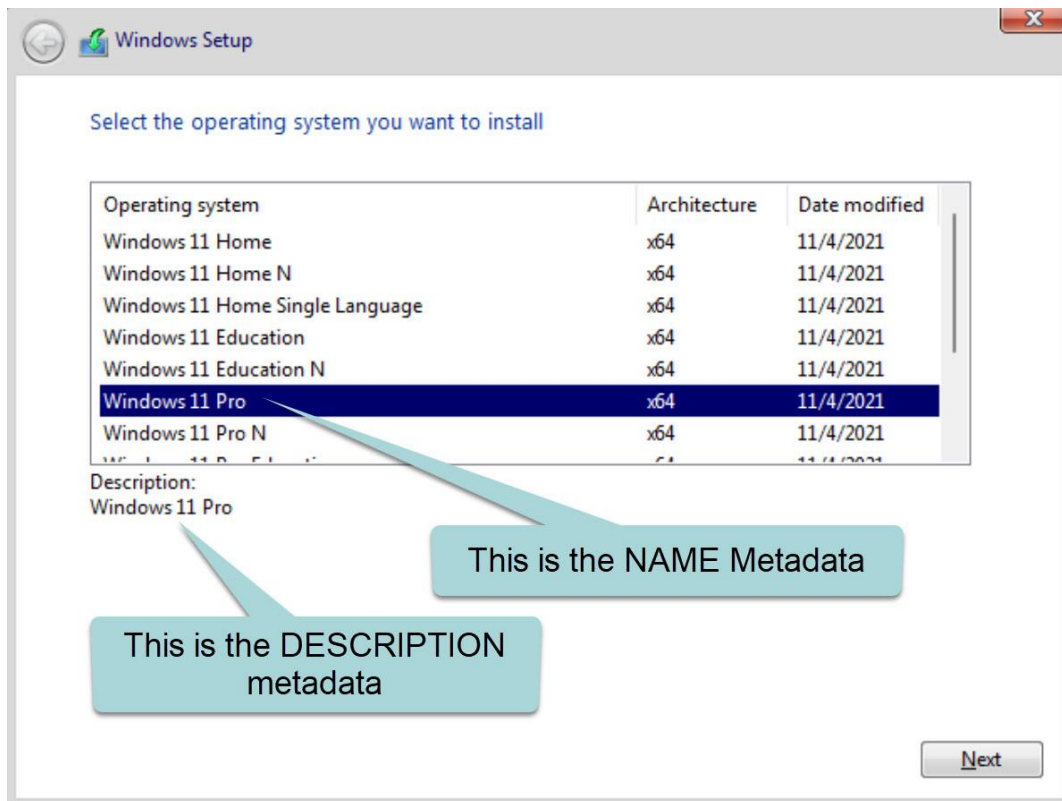
Run this command to exit from diskpart:

```
exit
```

Run the following command at the command prompt, modifying it where needed or desired. Replace the **C:** in the **/imagefile**, **/capturedir**, and **/scratchdir** switches if the Windows drive letter noted above is not **C:**. Note that the following is all one single long command.

```
DISM /capture-image /imagefile:C:\Image\install.wim /capturedir:C:\
/name:"W11 Pro" /description:"Win 11 Pro 21H2 x64" /ScratchDir:C:\Scratch
/compress:maximum /checkintegrity /verify /bootable
```

This will capture the image. Note that the **/name** parameter sets the name that is displayed by Windows Setup. As an example, when you boot a multi edition Windows image or disk, you get a menu showing options such as "Windows 11 Pro", "Windows 11 Home", etc. The names that are displayed are what you put in the **/name** switch. The **/description** switch determines what is shown below that menu when an entry is highlighted in the menu. This is an example:



IMPORTANT: Both the name and description fields are mandatory! If you need to modify the name or description later for a Windows edition within the WIM, see the information discussed in the section called [Extracting Windows editions](#).

When the above command completes (it may take a while), reboot the system by closing the command prompt and the Windows setup screen. Allow it to boot normally from the HD. This will take a while because the system will go through the OOBE (Out of Box Experience). Note that at this point, we already have an image captured of the reference system, so we don't care about the reference system any longer other than the fact that we need to get the image off that system. We are rebooting it so that we can copy off the image file (**install.wim**).

Reminder: If you configured AutoLogon in the optional steps earlier, the reference system will also autologon once when booted and then log you back off if you configured the **runonce.bat** to perform a logoff. Just log back on again with the admin user account and password that you created if that happens.

Copy the **install.wim** file from **C:\Image** folder on the reference system to **C:\ISO_Files\Sources** on the technician system replacing the file that is already there. Use a thumb drive, network, etc. to move the file.

If you used a different folder, substitute the correct location. Note that at this point you should be able to connect to shares from within the reference system, so you can copy it over the network if you want.

Make sure that the **autounattend.xml** that you created earlier is in place in your **C:\ISO_Files** folder as instructed earlier in the section [Finalizing the Answer File \(autounattend.xml\)](#).

When done, shut down the reference system.

```
*****
* End of Procedure: *
* Creating Media for Windows Automated, Unattended Installation *
*****

*****
* Status Check *
*****
```

At this point you have in your **C:\ISO_Files** folder all the files needed to create an ISO image file and bootable media with your customized Windows setup. Depending upon the option selected at the start, this may be as simple as all the same files that were in your original Windows image but with an **autounattend.xml** answer file added for unattended setup. It may also contain a new **install.wim** file that holds a customized Windows image that you created on your reference system either with or without an **autounattend.xml**.

IMPORTANT: Make sure you understand this! If you have created only an **autounattend.xml** answer file but you have **NOT** replaced the **install.wim** file, then all the editions that were on the original ISO image will still be present because you didn't modify the original **install.wim** file. For example, Windows 11 Pro, Home, Education, etc. will all still be present if they were present in your original image. However, if you created a sysprep image of a reference system, that new **install.wim** will contain only the one edition of windows that you installed on that reference system. If you want to combine multiple editions of Windows that come from one or multiple ISO image files, see the section [Multiple Windows Editions in One Image Using Only MS Tools](#) before you create your bootable media.

So, what's next? As an advanced topic, you can review the section below for optional steps that you can take to automate actions after Windows setup completes. If you prefer, you can skip that section and go right to creating an ISO image or bootable media such as a DVD or flash drive, or both from the files in your **C:\ISO_Files** folder. Continue to [Methods to Create Bootable Images, DVD, Drives](#) to accomplish this.

```
*****
* Optional: Automating Actions Upon Completion of Windows Setup *
*****
```

Run a script after setup is complete (SetupComplete.cmd)

%WINDIR%\Setup\Scripts\SetupComplete.cmd Order of Operations

1. After Windows is installed but before the logon screen appears, Windows Setup searches for the **SetupComplete.cmd** file in the **%WINDIR%\Setup\Scripts** directory.
2. If a **SetupComplete.cmd** file is found, Windows Setup runs the script. Windows Setup logs the action in the **C:\Windows\Panther\UnattendGC\Setupact.log** file.

Setup does not verify any exit codes or error levels in the script after it executes `SetupComplete.cmd`.

Warning: You cannot reboot the system and resume running `SetupComplete.cmd`. You should not reboot the system by adding a command such as `shutdown -r`. This will put the system in a bad state. The `SetupComplete.cmd` should be allowed to run to completion.

3. If the computer joins a domain during installation, the Group Policy that is defined in the domain is not applied to the computer until `Setupcomplete.cmd` is finished. This is to make sure that the Group Policy configuration activity does not interfere with the script.

Windows Unattend scripts:

Create one of these settings in your `unattend.xml` answer file to run during the Windows setup process.

NOTE: `Shell-Setup\LogonCommands\AsynchronousCommand` now works like `LogonCommands\AsynchronousCommand`: all commands using these settings now start at the same time and won't wait for the previous command to finish.

Some of these settings run in the user context, others run in the system context depending on the configuration pass.

- Add **Setup > RunAsynchronousCommand** or **RunSynchronousCommand** to run a script as Windows Setup starts. This can be helpful for setting hard disk partitions.
- Add **Deployment > RunAsynchronousCommand** or **RunSynchronousCommand** to the **auditUser** configuration pass to run a script that runs when the PC enters audit mode. This can be helpful for tasks like automated app installation or testing.
- Add **Shell-Setup > LogonCommands > AsynchronousCommand** or **FirstLogonCommands > SynchronousCommand** to run after the Out of Box Experience (OOBE) but before the user sees the desktop. This can be especially useful to set up language-specific apps or content after the user has already selected their language. Note that **FirstLogonCommands** runs only once but **LogonCommands** runs each time.

Use these scripts sparingly because long scripts can prevent the user from reaching the Start screen quickly. For retail versions of Windows, additional restrictions apply to these scripts. For info, see the Licensing and Policy guidance on the Microsoft OEM Partner Center.

NOTE: When you add a script using **FirstLogonCommands**, it will be triggered on the next boot, even if you boot into audit mode using **Ctrl + Shift + F3**. To boot to audit mode without triggering these scripts, add the setting: **Deployment > Reseal > Mode = Audit**.

Installing Apps from the RunOnce.bat

If you need to modify the **RunOnce.bat** file discussed previously to install apps or perform other actions, you can add the following lines to allow Windows to determine the drive letter of the installation media:

```
FOR %%I IN (A B C D E F G H I J K L M N O P Q R S T U V W X Y Z) DO IF  
EXIST %%I:\Installers\installationstuff.txt SET DRIVE=%%I
```

This line will search each drive letter for a file in the root called **installationstuff.txt**. Once found, the drive letter will be saved in the variable **DRIVE**. This allows you to install apps as in this example:

```
"%DRIVE%:\Installers\MyApp" /s
```

Note that you can replace the filename with any path and filename you wish and **RunOnce.bat** can also be named anything.

If you need to modify the **RunOnce.bat** in an existing sysprep image, do this:

NOTE: I won't provide every command to perform the below steps. You can find that information elsewhere in this document. The section on injecting updates has all the DISM commands for mounting, unmounting, and ISO image burning.

Extract Windows to a location such as **C:\Project\ISO_Files**.

Mount the **install.wim** to a location such as **C:\Project\Mount**.

From the mounted **install.wim**, go to this location (replace **WinUser** with an Admin user you created other than Administrator):

```
C:\Project\Mount\Users\WinUser\AppData\Roaming\Microsoft\Windows\Start  
Menu\Programs\Startup
```

Modify the **RunOnce.bat**

Place the Installers folder and **installationstuff.txt** files in the **C:\Project\ISO_Files** folder - again, modify as desired, but make sure your **RunOnce.bat** references the correct location(s).

Unmount the **install.wim**.

Create a new ISO image.

```
*****
* Methods to Create Bootable Images, DVD, Drives *
*****
```

It is important that you understand the purpose of each of the sub-sections that follow before you proceed.

[Bootable ISO Image](#) - This process will take a set of Windows distribution files located in a folder on your HD and create an ISO image from these files. It does not matter if the folder on the HD contains a single edition of Windows or multiple editions. This procedure will not alter that. It merely makes an ISO image out of these files and folders. This is a perfect choice to use after you have completed the [Creating Media for Windows 10 Automated, Unattended Installation](#) process.

[Multiple Windows Editions in One Image Using Only MS Tools](#) - This procedure will allow you to take Windows editions from one or multiple ISO images and combine them all into one ISO Image. This can include customized images created using sysprep.

[Bootable DVD](#) - This procedure will allow you to take a Windows ISO image and create a bootable DVD from that image. This will also work with Blu-Ray media.

[Bootable Flash Drive Using Only Native MS Tools](#) - Allows you to take a Windows ISO image and create a bootable flash drive, HD, SD card, etc. from that image.

```
*****
* Bootable ISO Image *
*****
```

Either copy the contents of a Windows ISO to a folder on a HD, thumb drive, etc. and note the path to the folder, or, if you already have Windows files extracted to a folder, note the path to that folder. In this example, I'll assume **C:\ISO_Files**.

NOTE: If you arrived here after following the "Creating Media for Windows Automated, Unattended Installation" section earlier in this document, then you already have files copied to your HD that have now been updated.

Make sure you have the **Windows ADK** installed as previously instructed.

Go to **Start > All Apps > Windows Kits** and open **Deployment and Imaging Tools Environment** in elevated mode (as Admin).

You will end up at a command prompt with a long path. Type **cd** and hit **Enter** to go back to the root. This simply makes the prompt shorter and the command line less cluttered.

Enter the following command. Note that this is a single long command. Be careful - there are no spaces in some of the places you might expect a space to be present:

```
oscdimg.exe -m -o -u2 -udfver102 -
bootdata:2#p0,e,bc:\iso_files\boot\etfsboot.com#pEF,e,bc:\iso_files\efi\micro
soft\boot\efisys.bin c:\iso_files c:\Win11PRO.iso
```

Replace the 3 occurrences of **c:\iso_files** with the correct path to where your files are located. Also, replace the **c:\Win11PRO.iso** at the end of command with the correct path and name for the output file that you want to save. For paths with spaces, include them inside of double quote marks (").

IMPORTANT: If you have an **autounattend.xml** answer file on the root of your image, I strongly suggest noting this somewhere obvious. If you boot a VM from this image, or if you boot a system from media created with this image, setup will begin without warning and can wipe out your HD depending upon the selected options!

```
*****
* Multiple Windows Editions in One Image *
*       Bootable from BIOS and UEFI       *
* Created Using Only Native Windows Tools *
*****
```

IMPORTANT NOTE: When you have finished this section and are ready to deploy to a thumb drive, I suggest using this procedure: [Bootable Thumb Drive Using Only Native MS Tools](#). The advantages of using this method are outlined in that section.

END OF NOTE

This procedure will allow you to create an ISO image from which multiple editions of Windows can be installed. As an example, you might create a single ISO image that allows installation of all the following Windows editions:

```
Windows 11 Pro (Original)
Windows 11 Pro with the Latest Updates
Windows 11 Home (Original)
etc.
```

You can include syspreped images and unattended installation can be setup easily.

Create the following folders:

```
C:\ISO_Files      < Used to store base image
C:\WIM            < Will hold WIM with all editions of Windows in image
```

NOTE: You can use any folder names you want, just be sure to change them in the commands below as well.

Create a Base Image:

Extract the contents of a Windows image to the **C:\ISO_Files** folder.
Delete the **C:\ISO_Files\Sources\install.wim** file.

This is your base image. Go to [Extracting Windows editions](#) (below).

Extracting Windows editions:

You will now extract all Windows editions and copy them into a single **install.wim** file located in **C:\WIM**.

Mount an ISO image that contains one of the Windows editions that you want to add to this compilation. In this example, I've mounted the ISO image to drive **E:**.

Run the following command to get a list of editions in this ISO image:

```
dism /Get-WimInfo /WimFile:E:\Sources\install.wim
```

Note the index number associated with the edition of Windows you want to use. As an example, I'm going to assume index #6 applies to Win 11 Pro and will use that in my command below.

Run the following command:

```
DISM /Export-Image /SourceImageFile:E:\Sources\install.wim /SourceIndex:6  
/DestinationImageFile:C:\WIM\install.wim /DestinationName:"W11 PRO version  
21H2"
```

DestinationName in the above command is the same as **Name** in the ImageX command shown below. This is the name displayed on the boot menu for this image. The **Description** appears below the menu for the currently highlighted image name in the menu. Both the **NAME** and **DESCRIPTION** are **MANDATORY!** You can, however, omit the **/DestinationName:** option above and the already existing name will then be kept. Just be certain that there is an already existing name. The above command does not have an option to change the **DESCRIPTION** so if one needs to be added you will need to run the command below. If a **DISM /Get-Wiminfo** (as shown above) indicates that **DESCRIPTION** is not defined, then you must add one! To do so, extract the contents of the ISO image (you should have already done this with the **DISM /Export-Image** command above), make sure the **install.wim** is not read-only, then run the following ImageX command:

```
ImageX /info <path to install.wim> <index_number> "Name" "Description" /check
```

Note that as you export images to **C:\WIM**, the first export will be index 1, the second will be index 2, etc. It does not matter what the index was in the source! For example, if you export index 6 first, then it will become index 1 in the new image since you exported it first.

Again, you can run the **DISM /Get-WimInfo** command at any time to see the info for the WIM file.

Remember, both **NAME** and **DESCRIPTION** are necessary.

TIP: You can just leave off the **DestinationName:** option in the **DISM /Export-Image** command and then set both the **NAME** and **DESCRIPTION** with the ImageX command.

You can keep exporting additional Windows editions. They do not all need to come from the same image file. Use of multiple different ISO images is perfectly fine, however, **they should be the same Windows version and build number.**

You should now end up with all editions of Windows in the **C:\WIM\install.wim** file.

When done, copy the **install.wim** file to **C:\ISO_Files\sources**, overwriting any **install.wim** file located there.

Final Preparations:

Create a text file called **ei.cfg**. Place the following two lines in that file:

```
[Channel]  
Retail
```

NOTE: Use something like notepad to create this file. Don't use a program like Word that can insert special formatting characters into the file.

Place the **ei.cfg** file in **C:\ISO_Files\sources**.

Notes about **ei.cfg**: Let's say that you have a computer that shipped with Windows 11 Home, but you upgraded to Pro. You may find that when you try to perform a clean install, Windows simply starts installing Home and doesn't even show you the menu from which you can select other editions. By placing this file in the sources folder, we prevent that from happening. This happens because Windows setup reads a signature in the BIOS that tells setup which edition of Windows the system was shipped with.

Note that when doing an unattended install, this file is not needed because the **autounattend.xml** specifies the edition of Windows to install, but you can leave the **ei.cfg** in place as it won't hurt anything. My preference is to always include an **ei.cfg** file.

Creating the Final ISO image:

Open **Deployment and Imaging Tools** in elevated mode from the Windows ADK.

Run the following command to create the final ISO image. Be careful! There are spots in this command where you may expect a space, but no space is present:

```
oscdimg.exe -m -o -u2 -udfver102 -L"VolumeName" -  
bootdata:2#p0,e,b"c:\iso_files\boot\etfsboot.com"#pEF,e,b"c:\iso_files\efi\mic  
rosoft\boot\efisys.bin" "c:\iso_files" "C:\ISO Images\My Image.iso"
```

In the above example, **VolumeName** is the optional name to give the volume. Just specify "" if you want no volume name or leave off the **-L"VolumeName"** parameter entirely.

To create a bootable thumb drive from the image you just created, I suggest following this method: [Bootable Flash Drive Using Only Native MS Tools](#).

```
*****  
* Performing an Unattended Setup with the Multi Edition Media *  
*****  
  
*****  
* How Windows Behaves When Booting from a Multi Edition Image *  
*****
```

Let's suppose that you place an **autounattend.xml** answer file on the root of the image or boot media and that the answer file was made for Windows 11 Pro. When you boot, if only a single edition of Windows in the image is based upon Windows 11 Pro, that edition will be the one that is installed. However, let's say that you have a standard Win 11 Pro edition and a Win 11 Pro Sysprep edition on that media. Let's also assume that you have an autounattend.xml answer file that has a generic key for the Windows Pro edition. When you boot from that media to install Windows, you will be presented a menu showing only these two Win 11 Pro based editions and you will need to choose one or the other because setup cannot determine which of the two Windows Pro editions to install. In this situation, follow the steps below if you want to automate the selection of the correct Windows edition.

```
*****
* Automating Installation for a Multi Edition Boot *
*****
```

In order to select which edition to have unattended setup install when there are more than one of the same edition (for example, multiple Win 11 Pro editions), you will need to modify your **autounattend.xml** answer file like this:

Load your **autounattend.xml** answer file in the Windows System Image Manager. In the Windows Image pane, expand the following tree:

```
Setup > ImageInstall > OSImage > InstallFrom
```

Right click on **MetaData** and select **Add Setting** to **Pass1 windowsPE**

In the Answer File pane, click on **InstallFrom** in the tree and set the following value:

```
Path          \install.wim
```

In the Answer File pane, click on **MetaData** in the tree and set these values:

```
Key           /IMAGE/INDEX
Value         <the index number of the Edition you want to install>
```

NOTE: Each Windows edition will have a unique index number: 1, 2, 3, 4, etc.

To determine the index number for the value above, run this command (assuming the image is mounted as **E:**):

```
dism /Get-WimInfo /WimFile:E:\Sources\install.wim
```

Example:

If you want to install the edition with an index of 6, set these values:

```
Key           /IMAGE/INDEX
Value         6
```

Note that you can also choose the edition of Windows by specifying the **NAME** or **DESCRIPTION** of the Windows edition like this rather than the index number:

```
/IMAGE/NAME
or
/IMAGE/DESCRIPTION
```

The same command shown above that lists the index also shows the **NAME** and **DESCRIPTION** for each Windows edition.

As always, perform a validity check on your answer file, and save it as **autounattend.xml** on the root of your media.

NOTES: If you are installing to a VM, see the note below called [Special Considerations for Virtual Machines](#) that can make things much easier for you. In addition, note that if you use a metadata entry noted above, you can omit the generic license key if you wish, but you can also leave it in place.

```
*****
* Special Considerations for Virtual Machines *
*****
```

The above steps are super easy to accomplish on writable media such as a thumb drive because you can simply drop your answer file onto the physical media, but for an ISO image, you probably don't want to have to recreate the image every time to automate installation of different Windows editions.

Consider creating a separate small ISO file with nothing but your answer file. Creating that ISO can take only seconds. Simply attach it as a second DVD-ROM drive to your VM. This works because Windows setup will search each drive for an answer file, not just the drive that has your Windows installation files.

```
*****
*               END OF SECTION               *
* Multiple Windows Editions in One Image *
*       Bootable from BIOS and UEFI       *
* Created Using Only Native Windows Tools *
*****
```

```
*****
* Create a Bootable DVD *
*****
```

Follow the steps in the section [Bootable ISO Image](#), then use your favorite DVD burning application to burn that ISO image to disc. As an alternative, in Windows, right-click on the ISO image and choose "Burn disc image". If you have a Blu-Ray writer, you can also use Blu-Ray media.

```
*****
* Bootable Flash Drive Using Only Native MS Tools *
*****
```

- 1) Run diskpart.
- 2) From the diskpart> prompt, run this command:

list disk
- 3) From the size of the disks, try to determine which disk ID is your thumb drive. If that is not enough information, run these commands to get more detail on the disk. In this example, I want more info on disk 4:

```
select disk 4
detail disk
```

Here is some sample output:

```
DISKPART> list disk
```

Disk ###	Status	Size	Free	Dyn	Gpt
Disk 0	Online	476 GB	0 B		*
Disk 1	Online	3726 GB	0 B		*
Disk 2	Online	465 GB	1024 KB		*
Disk 3	Online	7452 GB	0 B		*
Disk 4	Online	238 GB	0 B		

```
DISKPART> select disk 4
```

Disk 4 is now the selected disk.

```
DISKPART> detail disk
```

```
SanDisk Extreme Pro USB Device
Disk ID: DC727760
Type    : USB
Status  : Online
Path    : 0
Target  : 0
LUN ID  : 0
Location Path : UNAVAILABLE
Current Read-only State : No
Read-only  : No
Boot Disk  : No
Pagefile Disk : No
Hibernation File Disk : No
```



```
Crashdump Disk : No
Clustered Disk : No
```

Volume ###	Ltr	Label	Fs	Type	Size	Status
Info						
-----	---	-----	-----	-----	-----	-----

Volume 7	Z	My Software	NTFS	Removable	238 GB	Healthy
C:\MountPoints\My Software\						

- 4) When you determine the correct disk, run these commands. I am assuming disk 5 in the following example:

```
select disk 5
clean (if you get an error running this command, run it again)
convert mbr
create partition primary size=2000
active
format fs=fat32 quick
assign
create partition primary
format fs=ntfs quick
assign
exit
```

NOTE: The **assign** commands will assign the next available drive letter. If you want to assign specific drives letters, for example **E:**, use the command **assign letter=E**.

- 5) We will now copy files from the source to the two partitions on the thumb drive. To be clear, all the files and folders that you are being asked to copy come from your source folder such as a folder on your hard disk that contains all the Windows files, or from a mounted Windows ISO image.

Start by checking to see if your original source has a file called **ei.cfg** in **\sources** folder. If that file is already present, skip to step 6.

Create a text file called **ei.cfg**. We will use that below. Place the following 2 lines into that file:

```
[Channel]
Retail
```

Notes about ei.cfg: Let's say that you have a computer that shipped with Windows 10 Home, but you upgraded to Pro. You may find that when you try to perform a clean install, Windows simply starts installing Home and doesn't even show you the menu from which you can select other editions. By placing this file in the sources folder, we prevent that from happening. This happens because Windows setup reads a signature in the BIOS that tells setup which edition of Windows the system was shipped with.

Note that when doing an unattended install, this file is not needed because the autounattend.xml specifies the edition of Windows to install, but you can leave the **ei.cfg** in place as it won't hurt anything. I always include this file.

6) Follow these steps to copy files to your thumb drive:

- Copy all files and folders **EXCEPT** the **\sources** folder to the FAT32 partition (the first, smaller partition).
- If you are planning to use an **autounattend.xml** answer file, place it in the root of the first (FAT32) partition.
- Create a folder called **sources** on the FAT32 partition.
- Copy the file **\sources\boot.wim** to the FAT32 partition **\sources** folder.
- Create a folder called **sources** on the NTFS partition (the second, larger partition).
- Copy all files and folders from the **\sources** folder **EXCEPT** **boot.wim** to the **\sources** folder on the NTFS partition.
- If you created an **ei.cfg** file, copy it to the **\sources** folder of the NTFS partition.

At this point, you should have a bootable thumb drive.

NOTE: When booting you may be presented with an option to boot either the FAT32 or the NTFS partition. Choose the FAT32 partition. If it is not clear which one the FAT32 partition is, simply try either. If it doesn't work, reboot and try the other.

Part 2 - Managing Offline Windows Images and Live, Online Installations

```
*****
* Updating Windows Image with Latest Windows Updates *
*****
```

```
*****
* Obtain and Organize Updates *
*****
```

First, you will need to obtain updates. Here are resources for Windows updates. Search for Windows updates that apply to your version of Windows. For example, in the search box, type the following line. Include the quotes as shown:

"Windows 11 version 23H2" x64.

The Microsoft Update Catalog is located here:

<https://www.catalog.update.microsoft.com/>

I find the easiest way to locate updates on the update catalog is to sort by date. You can also narrow it down to the current updates only like this:

"Windows 11 version 23H2" x64 2023-12

Also, note that some updates such as the Safe OS Dynamic Update and the Setup Dynamic Update are not released every month. In fact, it's possible that there may be no such update for your version of Windows. You should determine when your ISO image was released and search back in time to that date for these updates. For example, as I write this paragraph it is January, 2024. Looking at the update history for Windows 11 23H2 (see link below) I see that the first updates were available in October, 2022. So, in the update catalog I would work my way back in time looking for each type of update and only grabbing the most recent one of each update type. I would go back as far as October of 2022. **EXCEPTION: OOBE ZDP** updates are **NOT** cumulative. As a result, you should download **ALL** available updates of this type, not just the most recent.

See the following link for a list of updates (does not include every type of update):

Windows 10 Update History:

<https://support.microsoft.com/en-us/topic/windows-10-update-history-8127c2c6-6edf-4fdf-8b9f-0f7be1ef3562>

Windows 11 Update History:

<https://support.microsoft.com/en-us/topic/windows-11-version-23h2-update-history-59875222-b990-4bd9-932f-91a5954de434>

NOTE: I would suggest searching for and downloading the latest version of each of the following updates:

- Windows LCU/SSU (Latest Cumulative Update. **DON'T** use the **DYNAMIC** version)
- Standalone SSU (Will rarely exist)
- Windows Safe OS Dynamic Update
- Windows Setup Dynamic Update
- Cumulative Update for .NET Framework 3.5, 4.8, and 4.8.1
- Windows OOBE ZDP Updates. For OOBE ZDP updates, download **ALL** of these updates, not just the most recent because these are NOT cumulative.

Save the updates to a folder. In this example, we are saving to **C:\WinUpdates**.

Create a directory structure like this:

```
C:\WinUpdates
    LCU                < Place the Latest Cumulative Update in this folder
    SSU                < If a standalone SSU exists, add it here
    Other              < Other updates such as .NET updates and OOBE ZDP updates
    SafeOS_DU          < For the SafeOS (WinRE) Dynamic Update
    Setup_DU           < For the Setup Dynamic Update
    PE_Files           < Files to add to the boot.wim (Win PE)
```

NOTE: When downloading updates from the Microsoft Update Catalog, the **Setup DU** will show a description such as **Windows 10 Dynamic Update**. The **SafeOS DU** will show **Safe OS Dynamic Update**. For the **LCU** (Latest Cumulative Update) download the update named **Cumulative Update...** and **NOT** the **Dynamic Cumulative Update....** The correct cumulative update will have a .MSU extension, not a .CAB extension. While the Dynamic Cumulative update might work, I have not tested it. Note that the structure above is only a suggestion, however, that is what I will use in the examples that follow.

Updates are cumulative. This means that the latest update contains all the previous updates. As a result, you should apply the latest update of any given kind to Windows. **DO NOT** apply multiple updates of the same kind.

EXCEPTION: The **OOBE ZDP** updates are not cumulative. Download **ALL** of the OOBE ZDP updates that are available for your version of Windows.

Note: When a newer version of an update becomes available, it is okay to apply this to a Windows image that has had a previous update applied. The new update will intelligently update only those components that need to be updated.

The following table shows the order in which updates should be applied. The number in parenthesis in the table relates to language items and features on demand which we don't deal with in this document. You can simply skip those items. If you want to see a sample of how to address those items, look at this article from Microsoft:

<https://docs.microsoft.com/en-us/windows/deployment/update/media-dynamic-update>

	WinRE (winre.wim)	Main OS (install.wim)	WinPE (boot.wim)	New media
Add SSU Dynamic Update	1	9	17	
Add Lang Pack	(2)	(10)	(18)	
Add Localized Optional Packages	(3)		(19)	
Add Font Support	(4)		(20)	
Add TTS Support	(5)		(21)	
Update Lang.ini			(22)	
Add Features On Demand		(11)		
Add Safe OS Dynamic Update	6			
Add Setup Dynamic Update				26
Add setup.exe from WinPE				27
Add boot manager from WinPE				28
Add LCU		12	23	
Clean Image	7	13	24	
Add Optional Components		14		
Add .NET and .NET Updates, OOBE ZDP, Other		15		
Export Image	8	16	25	

NOTE 1: Microsoft now ships the Latest Cumulative Update (LCU) and Servicing Stack Update (SSU) as a single combined package in the LCU update. The SSU should be extracted from the combined package in order to install it as is shown in the steps below. However, there may be rare cases where a separate Standalone SSU is still published. If that happens, apply that SSU first before the combined LCU / SSU.

NOTE 2: Microsoft has an older article that swaps columns 2 and 3 in the above table. That is okay, so long as each section gets the appropriate updates.

```
*****
* Applying the Updates *
*****
```

IMPORTANT:

When updating Windows images, if you apply the latest updates every month when they are released, I suggest that you apply the updates to a clean Windows image rather than applying updates to an already previously updated image. The reason for this is that certain types of updates such as enabling NetFX3, can have pending operations which will cause problems for updates being applied thereafter. In practice, I've never encountered a problem with this, but it is something to be aware of.

Make sure you have the **Windows ADK** installed (only the **Deployment Tools** are needed). If not, download and install it now.

Go to **Start > All Apps > Windows Kits** and open **Deployment and Imaging Tools Environment** in elevated mode (as Admin).

You will see a command prompt with a long path. Type **cd ** and hit **Enter** to go back to the root. This makes the prompt shorter and the command line less cluttered.

If you have any difficulties and need to start over, in particular, if you had an image mounted with DISM, perform a cleanup as in this example:

First, you can check to see if there are any open DISM mounts like this:

```
Dism /Get-MountedImageInfo
```

Then, if there are any open mounts run these commands. Not needed if there are no open mounts.

```
DISM /Unmount-Image /MountDir:"C:\Project\Mount" /Discard
DISM /Cleanup-WIM
DISM /Cleanup-Mountpoints
C:\windows\system32\takeown /f "C:\Project\*.*" /r /d y
icacls "C:\Project\*.*" /T /grant %username%:F
```

Finally, clear the contents of the directories and make sure to empty the recycle bin as this mount can still cause difficulties if stuck open in the recycle bin.

IMPORTANT: If you find that the above does not work and you cannot delete the folder where an image was mounted, log off, log back on, and then try the above again. If you still have difficulties, make sure you have cleaned out the recycle bin, reboot the system and try the above commands one more time.

Start by creating a directory structure for your project as shown below. This is an example, use whatever you want. This is the structure that I will use:

```
C:\Project
  Assets                < Will be used to store files once updated
  ISO_Files             < Used to store the original Windows files to be updated
  Logs                 < Used to save log files from the commands being run
  Mount                < Used for mounting the main OS image (install.wim)
  Scratch              < Used by the DISM command
  SSU                  < Used to save the extracted SSU
  Temp                 < Used by the DISM command
  WIM                  < Used to store the updated install.wim
  WINPE                < Used to hold the original boot.wim file before updating
  WINRE                < Used to hold the original winre.wim file before updating
  WINPE_MOUNT          < Used for mounting the boot.wim
  WINRE_MOUNT          < Used for mounting the winre.com
```

Create a directory to place Windows Updates into. Example: **C:\WinUpdates**.

NOTE: In this example, I'm going to create a directory structure for the Windows updates that looks like this:

```
C:\WinUpdates
  LCU                  < Place the Latest Cumulative Update in this folder
  SSU                  < If a standalone SSU exists, add it here
  Other                < Other updates such as .NET updates and OOBE ZDP updates
  SafeOS_DU            < For the SafeOS (WinRE) Dynamic Update
  Setup_DU             < For the Setup Dynamic Update
  PE_Files             < Files such as scripts to add to the boot.wim (Win PE)
```

TIP: All folders should have only a single update. The **Other** and **PE_Files** folders can contain multiple files. This is because only one Cumulative Update is needed, and the same is true for other items - only the latest version of each is needed. **EXCEPTION:** For **OOBE ZDP** updates, download **ALL** available updates of this type, not just the most recent because OOBE ZDP updates are not cumulative

In the sections that follow, we will perform these operations:

- Update WinRE.wim
- Update the main OS (install.wim)
- Update WinPE (Boot.wim)
- Install additional updates (.NET updates, OOBE ZDP updates, etc.)

Extract the contents of a Windows ISO image or copy the files from a bootable Windows flash drive to **C:\Project\ISO_Files**.

Get the Windows update package(s) from the Microsoft Update Catalog. For example, grab the latest cumulative update listed for Windows 10. Place the .msu file update to a folder, for example, **C:\WinUpdates\LCU\windows10.0-kb4016871-x64_27dfce9dbd92670711822de2f5f5ce0151551b7d.msu**.

Tip: Rename the file to make it easier to work with. Example, **KB4016871.msu**.

Make sure to grab all the available updates and put them in the folders noted earlier. It's perfectly fine if some update types have no updates available.

```
*****
* Summary of the Update Process *
*****
```

There are three files that we need to update from the Windows distribution:

install.wim	Holds the main Windows OS
boot.wim	Holds WinPE - Windows Preinstallation Environment
winre.wim	Holds WinRE - Windows Recovery Environment

The first component that we want to update is the **winre.wim**, but this file is located WITHIN the **install.wim**. As a result, we will first need to mount the **install.wim** so that we can retrieve the **WinRE.wim**. Think of this as being like having a ZIP file inside of another ZIP file. Once the **winre.wim** is updated, we will update the **install.wim**, and then, finally, the **boot.wim**.

In addition, we will update other files outside of these WIM files.

At this time, we need to sync a few files that may be mismatched.

With all those pieces updated, we will gather all the files from your original Windows distribution media and replace the original files that we updated with those updated versions.

When that is done, you can create a new ISO image or media with the updated Windows files.

```
*****
* End of Summary *
*****
```


Open an elevated PowerShell or command prompt. Enter the following command to check what editions are included in the main OS image:

```
DISM /Get-WimInfo /WimFile:"C:\Project\ISO_Files\Sources\install.wim"
```

Note the index number of your selected edition. In this example we are using an index number of 6 which, in my image, represents Windows 11 Pro.

Mount the image of your preferred Windows edition using its index number, index 6 in this example:

```
DISM /mount-image /imagefile:"C:\Project\ISO_Files\sources\install.wim"  
/index:6 /mountdir:"C:\Project\Mount"
```

By mounting the **install.wim**, we extract the contents of the WIM to another location (**C:\Project\Mount**). We can alter, add or remove files here, then when we dismount, all the changes are written back to the original file. Think of it much like unzipping a ZIP file, changing some files or adding / removing files, then re-zipping.

```
*****  
* Update WinRE.wim *  
*****
```

NOTE: In some instances, such as with a Syspreped image, a **WinRE.WIM** may be a hidden or system file.

If you have none of these updates skip this section on updating the **WinRE.wim**: A Standalone SSU, an LCU, or a Safe OS Dynamic Update.

From the directory where the **install.wim** is mounted, copy the **winre.wim** file. In this example you would copy **winre.wim** from **c:\Project\Mount\Windows\System32\Recovery** to **C:\Project\WinRE**

```
copy /B C:\Project\Mount\Windows\System32\Recovery\winre.wim  
C:\Project\WinRE
```

Mount index number 1 in the **winre.wim** (winre.wim has only one index)

```
DISM /mount-image /imagefile:"C:\Project\WinRE\WinRE.wim" /index:1  
/mountdir:"C:\Project\WinRE_Mount"
```

Although the SSU and LCU are now combined into the LCU, there could be rare cases where a Standalone SSU published. In order to handle this, you should first apply the Standalone SSU, if it exists.

```
DISM /Add-Package /Image:"C:\Project\WinRE_Mount"  
/PackagePath="C:\WinUpdates\SSU" /LogPath="C:\Project\Logs\dism.log"
```

The following commands will extract the SSU (if one exists) and inject the SSU and Safe OS Dynamic Update into WinRE. If there is no LCU / SSU Update or Safe OS Dynamic Update, skip that item. The use of **/LogPath** here and in all further commands is optional.

NOTE: The "f:" in the following command is NOT a drive letter. Don't change it.

```
expand "C:\WinUpdates\LCU\*.MSU" /f:"SSU*.cab" "C:\Project\SSU"

DISM /Add-Package /Image:"C:\Project\WinRE_Mount"
/PackagePath="C:\Project\SSU" /LogPath="C:\Project\Logs\dism.log"

DISM /Add-Package /Image:"C:\Project\WinRE_Mount"
/PackagePath="C:\WinUpdates\SafeOS_DU" /LogPath="C:\Project\Logs\dism.log"
```

TIP: Be aware that there are several ways to add packages to a WIM file. You can specify just a single file to inject into the WIM, you can specify a list of updates to inject, or you can point to a folder and have all updates located in the folder injected into the WIM. Here are some examples illustrating this:

Example: Adding a cumulative update to a mounted image:

```
DISM /Add-Package /Image:"C:\Project\Mount"
/PackagePath="C:\WinUpdates\kb4016871.msu"
/LogPath="C:\Project\Logs\dism.log"
```

Example: add multiple updates by specifying updates (this is one long command):

```
DISM /Add-Package /Image:"C:\Project\Mount"
/PackagePath="C:\WinUpdates\kb00001.msu"
/PackagePath="C:\WinUpdates\kb00002.msu"
/PackagePath="C:\WinUpdates\kb00003.msu" /LogPath="C:\Project\Logs\dism.log"
```

Example: Adding multiple updates in a folder by just pointing to the folder:

```
DISM /Add-Package /Image:"C:\Project\Mount" /PackagePath="C:\WinUpdates"
/LogPath="C:\Project\Logs\dism.log"
```

Note that for several of the update types such as the LCU, the above is completely unnecessary because you only need the latest version of those updates, meaning that only one will be installed. But this may be helpful for the **other** category where multiple **other** updates may need to be installed.

END TIP

The next command performs a cleanup operation:

```
DISM /Cleanup-Image /Image:"C:\Project\WinRE_Mount" /StartComponentCleanup
```

Now, unmount the image:

```
DISM /Unmount-Image /MountDir:"C:\Project\WinRE_Mount" /Commit
```

Once unmounted, the updates that you applied will have been committed to the **winre.wim** file located in the **C:\Project\WinRE** folder.

Export index 1 from the **WinRE.wim** (there is only one index in **WinRE.wim**):

```
DISM /Export-Image /SourceImageFile:"C:\Project\WinRE\WinRE.wim"  
/SourceIndex:1 /DestinationImageFile:"C:\Project\Assets\WinRE.wim"
```

A note about the "export" operation: When we performed the cleanup operation above, it marks duplicate files such as those replaced by updates for deletion. It is not until an export is performed that this data gets cleaned up. This is why you perform an export operation rather than simply copying the image file from the source to the destination when you are done updating it.

```
*****  
* Update the Main OS (install.wim) *  
*****
```

Updating the main OS is similar to updating the **WinRE.wim**. Here is an overview of the differences:

- As before, we will inject updates and then perform a cleanup EXCEPT that we will install optional components **AFTER** the cleanup because some actions such as enabling the NetFX3 capability (not covered in this document) require post installation actions. This would cause a failure if a cleanup were performed with those actions pending. You can run this command against a mounted image and look for "**State : InstallPending**" to see if such a state exists:

```
PowerShell "Get-WindowsCapability -path 'C:\Project\Mount'"
```

As an example, if NetFX3 is enabled, you would see:

```
Name   : NetFX3~~~~  
State  : InstallPending
```

- We will copy the **WinRE.wim** to the **install.wim** before we perform the cleanup.

NOTE: If you are choosing **NOT** to update the WinRE and WinPE components (for example, if no updates were available for these), then simply skip any steps later that ask you to copy those files (winre.wim and boot.wim).

For the **install.wim**, the components that we will install will be the Standalone SSU, the SSU from the combined LCU / SSU package, the LCU, and finally optional components such as the .NET cumulative update, OOBE ZDP updates, etc. Again, recall that the SSU was previously extracted from the combined LCU / SSU package but we will first install the Standalone SSU if one exists.

Once again, I won't explain each command since we have already done so. I'll just show examples of the commands that need to be run. Recall that the **install.wim** is already mounted (that was just about the first thing that we did).

```
DISM /Add-Package /Image:"C:\Project\Mount" /PackagePath="C:\WinUpdates\SSU"  
/LogPath="C:\Project\Logs\dism.log"
```

```
DISM /Add-Package /Image:"C:\Project\Mount" /PackagePath="C:\Project\SSU"  
/LogPath="C:\Project\Logs\dism.log"
```

```
DISM /Add-Package /Image:"C:\Project\Mount" /PackagePath="C:\WinUpdates\LCU"  
/LogPath="C:\Project\Logs\dism.log"
```

We will now copy the updated recovery image (**WinRE.wim**) to the mount location for the **install.wim**:

```
copy /B "C:\Project\Assets\WinRE.wim"  
"C:\Project\Mount\Windows\System32\Recovery" /Y
```

Perform the cleanup:

```
DISM /Cleanup-Image /Image:"C:\Project\Mount" /StartComponentCleanup  
/ResetBase /ScratchDir:"C:\Project\Temp"
```

IMPORTANT: When updating the **install.wim**, be aware that if the image was previously updated, it is possible that image cleanup may fail. This is because a cleanup cannot occur if there are pending operations. A common source of this issue is enabling NetFX3 in your image. If you do this, then you should apply future updates to a clean image created before you applied other updates or at least before you installed NetFX3. You will also note that we apply updates from the **Other** category **AFTER** we perform the cleanup. This is for the same reason - because certain updates in this category can leave pending operations. It is for this reason that I earlier suggested applying updates to a clean Windows image rather than one that that you have already updated previously. If you are updating only the items shown in this document and not adding things like NetFX3 then this is not going to be a problem.

If you want to check to see if your image has any pending operations, run this command now:

```
PowerShell "Get-WindowsCapability -path 'C:\Project\Mount'
```

As an example, if NetFX3 is enabled, you would see:

```
Name   : NetFX3~~~~  
State  : InstallPending
```

As an alternative, you could run the command below to filter and show only items with "InstallPending":

```
PowerShell "Get-WindowsCapability -path 'C:\Project\Mount' | Where-Object {  
$_.State -eq 'InstallPending' }"
```

End of Important Note

```
*****  
* Update WinPE image (boot.wim) *  
*****
```

Updating the **boot.wim** is the same as for the **WinRE.wim**. The only differences are:

- The **boot.wim** contains 2 images (index 1 and 2) so we need to update both.
- WinPE is located on the original media in the **\sources** folder, so we will copy it from there.
- The updates we will inject are the Standalone **SSU**, the SSU from the LCU package, and the LCU. Recall that we have already extracted the SSU from the combined LCU / SSU package.

I won't explain the purpose of each command again, I'll just supply the commands here.

Start by copying **boot.wim** from **C:\Project\ISO_Files\Sources** to **C:\Project\WinPE:**

```
copy /B "C:\Project\ISO_Files\sources\boot.wim" "C:\Project\WinPE"
```

Continue with the following commands:

```
DISM /mount-image /imagefile:"C:\Project\WinPE\boot.wim" /index:1  
/mountdir:"C:\Project\WinPE_Mount"
```

```
DISM /Add-Package /Image:"C:\Project\WinPE_Mount"  
/PackagePath="C:\WinUpdates\SSU" /LogPath="C:\Project\Logs\dism.log"
```

```
DISM /Add-Package /Image:"C:\Project\WinPE_Mount"  
/PackagePath="C:\Project\SSU" /LogPath="C:\Project\Logs\dism.log"
```

```
DISM /Add-Package /Image:"C:\Project\WinPE_Mount"  
/PackagePath="C:\WinUpdates\LCU" /LogPath="C:\Project\Logs\dism.log"
```

```
DISM /Cleanup-Image /Image:"C:\Project\WinPE_Mount" /StartComponentCleanup
```

```
DISM /Unmount-Image /MountDir:"C:\Project\WinPE_Mount" /Commit
```

```
DISM /Export-Image /SourceImageFile:"C:\Project\WinPE\boot.wim"  
/SourceIndex:1 /DestinationImageFile:"C:\Project\Assets\boot.wim"
```

To update index 2 in the **boot.wim**, run the commands below. Take note that we reference **index:2** in the first command this time. In addition, note that we are not yet unmounting or exporting this second index because we will need to copy some files from the boot.wim later while it is still mounted.

```
DISM /mount-image /imagefile:"C:\Project\WinPE\boot.wim" /index:2  
/mountdir:"C:\Project\WinPE_Mount"
```

```
DISM /Add-Package /Image:"C:\Project\WinPE_Mount"  
/PackagePath="C:\WinUpdates\SSU" /LogPath="C:\Project\Logs\dism.log"
```

```
DISM /Add-Package /Image:"C:\Project\WinPE_Mount"  
/PackagePath="C:\Project\SSU" /LogPath="C:\Project\Logs\dism.log"
```

```
DISM /Add-Package /Image:"C:\Project\WinPE_Mount"  
/PackagePath="C:\WinUpdates\LCU" /LogPath="C:\Project\Logs\dism.log"
```

```
DISM /Cleanup-Image /Image:"C:\Project\WinPE_Mount" /StartComponentCleanup
```

```
*****  
* Install Other Updates *  
*****
```

Now we install the **other** updates (.NET cumulative update, OOBE ZDP updates, etc.). You can place all these updates in the **other** folder. The next command will install them all:

```
DISM /Add-Package /Image:"C:\Project\Mount"  
/PackagePath="C:\WinUpdates\Other" /LogPath="C:\Project\Logs\dism.log"
```

We will not yet unmount and export the **install.wim** since we will need to copy some files from it while it is still mounted.

Some helpful commands (completely optional):

You can verify the packages that were installed with this command:

```
DISM /Get-Packages /image:"C:\Project\Mount"
```

The above command points to a folder before you dismount and commit the changes.

Enter the following command to check what editions are included in an image along with the associated index numbers:

```
dism /Get-WimInfo /WimFile:"C:\Project\Assets\install.wim"
```

```
*****  
* Updating the Base Image, Syncing *  
* Files, and Finalizing the Image *  
*****
```

For the "Setup Dynamic Update" we simply extract the contents to our media directly. To do so, run this command (replace **SetupDU.CAB** with the name of your file for the Setup Dynamic Update). Skip this command if you have no Setup Dynamic Update file. To apply the Setup Dynamic Update, run this command:

```
Expand "C:\Project\Setup_DU\SetupDU.CAB" -F:*  
"C:\Project\ISO_Files\Sources"
```

NOTE: the **F:** is **NOT** a drive letter. Don't change this!

Inside both the **boot.wim** (index number 2) and the **install.wim** (the index number that you have been working with) are some files that are supposed to be duplicates of each other. It is possible that these files are now no longer identical and that they are out of sync. The commands below correct this.

To perform the synchronization, run these commands:

```
Copy /b /y "c:\project\winpe_mount\sources\setup.exe"  
"c:\project\iso_files\sources\setup.exe"  
  
copy /b /y "c:\project\winpe_mount\windows\boot\efi\bootmgfw.efi"  
"c:\project\iso_files\efi\boot\bootx64.efi"  
  
copy /b /y "c:\project\winpe_mount\windows\boot\efi\bootmgr.efi"  
"c:\project\iso_files\bootmgr.efi"
```

NOTE: If you want to add other files to be available during Windows setup, add the files to the **C:\Project\WinPE_Mount** directory now. When Windows setup starts, these files will be available on the Ramdrive (**X:**). As an example, you might want to do this if you need scripts to be available during setup. Please note that this is something that is **very rarely used** so you can skip this unless you have a very specific reason to copy files to this location.

Now, we will close the **boot.wim** and export it. Then we will do the same for the **install.wim**:

```
DISM /Unmount-Image /MountDir:"C:\Project\WinPE_Mount" /Commit

DISM /Export-Image /Bootable /SourceImageFile:"C:\Project\WinPE\boot.wim"
/SourceIndex:2 /DestinationImageFile:"C:\Project\Assets\boot.wim"
```

Note that when we export index number 2, because the destination file is the same as when we exported index 1 earlier, the export adds that image to the file. You now have an image file with both index 1 and 2 updated. Also, note that for the second index of the **boot.wim**, we mark it as bootable with the **/Bootable** switch.

Now we can unmount and export the **install.wim** file:

```
DISM /Unmount-Image /MountDir:"C:\Project\Mount" /Commit

DISM /Export-Image
/SourceImageFile:"C:\Project\ISO_Files\sources\install.wim" /SourceIndex:6
/DestinationImageFile:"C:\Project\Assets\install.wim"
```

When you export the **install.wim**, the new **install.wim** will have only one index (index number 1) because you have exported only 1 Windows edition - the one associated with index 6 in this example. So even if your original index number was 6, that edition of Windows will now be index 1 in the new image.

Finally, we need to copy the updated **boot.wim** file and the updated **install.wim** to the base image:

```
copy /B "C:\Project\Assets\boot.wim" "C:\Project\ISO_Files\Sources" /Y

copy /B "C:\Project\Assets\install.wim" "C:\Project\ISO_Files\Sources"
/Y
```

In the folder **C:\Project\ISO_Files** you now have all your files from your Windows Distribution with all the updates in place.

You can now follow the steps in [Methods to Create Bootable Images, DVD, Drives](#) to create bootable media from these files.

If you simply want to create a bootable ISO image, enter the following command. Be careful; there are spots where you might expect a space, but none is present:

```
oscdimg.exe -m -o -u2 -udfver102 -
bootdata:2#p0,e,bc:\project\iso_files\boot\etfsboot.com#pEF,e,bc:\project\iso
_files\efi\microsoft\boot\efisys.bin c:\project\iso_files c:\Win11PRO.iso
```

Replace the three occurrences of **c:\project\iso_files** with the correct path to where your files are located. Also, replace the **c:\Win11PRO.iso** at the end of command with the correct path and name for the output file that you want to save. If that path contains spaces, enclose it in quotes.

```
*****  
* End of Procedure for Updating Windows Image with Latest Updates *  
*****
```



```
*****
* Adding or Removing Drivers from an Offline Image *
*****
```

This procedure is only needed if you want to add or remove drivers to or from an image. You can do this with a plain Windows installation image or an already customized image that you have created as detailed previously.

```
*****
* Get Drivers *
*****
```

Use one or both methods below to first obtain drivers.

```
*****
* Get drivers: Download Method *
*****
```

Download required drivers from manufacturer's site.
Extract the drivers in order to be able to access the .inf file.

The following is an example of extracting the contents of a driver called **driver.cab**. Start by opening an elevated command prompt and changing to the directory where the driver is located, then run these commands:

```
md .\extracted
expand .\driver.cab -f:*.* .\extracted
```

These commands will create a subdirectory called **extracted** and will extract the driver to that folder. "**f:**" is **NOT** a drive letter – don't change it.

```
*****
* Get drivers: Export Method (3 ways to export) *
*****
```

Create a new folder for exported drivers, in this example I'll use **C:\Drivers**. Follow any one of the options below. They all do the same thing.

1) Using DISM: To export all hardware drivers from an existing Windows installation, use the following command in elevated command prompt or PowerShell, replacing the path **C:\Drivers** with your actual path:

```
DISM /Online /Export-Driver /Destination:C:\Drivers
```

2) Using PNPUTIL: Run from an elevated command prompt or PowerShell:

```
pnputil /export-driver * C:\Drivers
```

3) Using PowerShell: This method provides more detail on the extracted drivers than just the cryptic directory names provided by the DISM or PNPUTIL methods.

Run these commands in the elevated PowerShell:

```
$Drivers = Export-WindowsDriver -Online -Destination C:\Drivers
$Drivers | Select-Object ClassName, ProviderName, Date, Version | Sort-Object
ClassName
```

You can use any variable in place of \$Drivers in the above command and you can sort however you want.

```
*****
* End of Get Drivers Sub-Section *
*****
```

```
*****
* Mount Offline Image *
*****
```

Create a folder to which you will extract your Windows ISO image, for example, **C:\ISO_Files**. If you already have this folder from the steps above, you can use those files. Otherwise, extract the files from your ISO image to that folder now.

Create a folder to mount the offline image. For example, **C:\Mount**.

Open an elevated command prompt. Enter the following command to check what editions are included in the image:

```
DISM /Get-WimInfo /WimFile:C:\ISO_Files\Sources\install.wim
```

```
*****
* Mounting an Image Located on HD, Thumb Drive, Flash Media *
*****
```

Mount the image of your preferred Windows edition using its index number, index 8 in this example:

```
DISM /Mount-Image /ImageFile:C:\ISO_Files\Sources\install.wim /Index:8
/MountDir:C:\Mount
```

Skip to [Adding or Removing Drivers](#) below.

```
*****
* Mounting an Image Located on Virtual HD (VHD / VHDX) *
*****
```

When mounting a VHD file, the index number is always 1. Set **/ImageFile** as path to virtual hard disk file, for instance mounting a VHDX file named **W11PRO.vhdx** which is stored in folder **F:\VHD_Files**, you would use the following mount command:

```
DISM /Mount-Image /ImageFile:F:\VHD_Files\W11PRO.vhdx /Index:1
/MountDir:C:\Mount
```

This will take some time. Please note that the drive where the Mount folder is located needs some free space. I do not recommend mounting a WIM image to a folder on a drive with less than 15 GB free space.

Mounting a VHD / VHDX file requires at least as much free space on the drive where the Mount folder is located as the size of the virtual hard disk. If you mount a dynamically expanding 64 GB virtual hard disk to be serviced, you need at least 70 GB free space.

```
*****
* Optional: Checking for Drivers Already Present *
*****
```

Optional: You can check which drivers are already present in an offline image with the following command:

```
DISM /Image:C:\Mount /Get-Drivers
```

```
*****
* Adding or Removing Drivers *
*****
```

```
*****
* Adding Individually Downloaded Drivers *
*****
```

If you downloaded individual drivers, you could add them to an offline mounted image with the following command:

For signed drivers, use this command:

```
DISM /Image:C:\Mount /Add-Driver /Driver:C:\Drivers\DriverName.inf
```

For unsigned drivers, use this command:

```
DISM /Image:C:\Mount /Add-Driver /Driver:C:\Drivers\DriverName.inf
/ForceUnsigned
```

Replace the mounted folder path **C:\Mount** in the above command with the actual path to your mount folder. Replace the driver path and name **C:\Drivers\DriverName.inf** with the actual path to the downloaded driver.

```
*****
* Adding Multiple Drivers by Recursing a Folder *
*****
```

If you have a folder that contains multiple drivers in subfolders, such as when exported from an existing installation as described earlier, you can add all drivers with one simple command, replacing the mount folder and drivers folder paths with your actual paths:

```
DISM /Image:C:\Mount /Add-Driver /Driver:C:\Drivers /Recurse
```

The **/Recurse** switch will cause the folder and all subfolders to be processed, adding all drivers found.

```
*****
* Removing Drivers *
*****
```

If you want to remove a driver from offline image, use the following command:

```
DISM /Image:C:\Mount /Remove-Driver /Driver:C:\Drivers\DriverName.inf
```

With **/Add-Driver** and **/Remove-Driver** you can add or remove multiple drivers:

```
DISM /Image:C:\Mount /Add-Driver /Driver:C:\Drivers\DriverName.inf
/Driver:C:\Drivers\Driver2Name.inf
```

```
*****
* Unmount Offline Image *
*****
```

You can now save changes and unmount an offline image with following command:

```
DISM /Unmount-Image /MountDir:C:\Mount /Commit
```

The **/Commit** switch is the important one, it commits (saves) all changes to the image.

Follow the section in [Methods to Create Bootable Images, DVD, Drives](#) to create media.

```
*****
* End of Procedure for Adding and Removing Drivers *
*****
```

```
*****
* Adding Boot Critical Drivers to the boot.wim *
*****
```

Create three folders. In this example I will use the following:

```
C:\Mount          < Will contain the boot.wim to inject drivers into
C:\Mount\Drivers  < Will contain drivers to inject
C:\Mount\BootWIM  < A temporary directory to mount the boot.wim into
```

Copy the drivers you wish to inject into the **C:\Mount\Drivers** folder.
Copy the **boot.wim** file to the **C:\Mount** folder.

NOTE: The drivers need to be extracted so that the .inf files are accessible.

Open a CMD prompt as Administrator.

You can see information about the image file (including available indexes) with this command:

```
dism /get-wiminfo /wimfile:c:\Mount\boot.wim
```

Note the available indexes from the above command.

Repeat the below procedure twice. Once using an index of "1" and again for index "2".

Use the following DISM commands to mount the **Boot.wim**:

```
DISM /Mount-Wim /WimFile:C:\Mount\boot.wim /Index:1
/MountDir:C:\Mount\BootWIM
```

Use the following DISM command to add the driver:

```
DISM /Image:C:\Mount\BootWIM /Add-Driver /Driver:C:\Mount\Drivers
/recurse
```

Use the following DISM command to unmount the **Boot.wim**:

```
DISM /Unmount-Wim /MountDir:C:\Mount\BootWIM /Commit
```

```
*****
* End Procedure for Adding Boot Critical Drivers to the boot.wim *
*****
```

```
*****
* Updating an Online Windows Installation with the Latest Updates *
*****
```

To apply updates to Windows while it is running (an online update), do the following:

- 1) Run the LCU update by double-clicking it. This will install the SSU if one is present. Otherwise, the LCU will be installed.
- 2) Run the LCU again to install the LCU. If the LCU was already installed, this will complete quickly as it will detect that the LCU was already installed.
- 3) From the **other** category of updates, run each installer one at a time.

All other updates are not needed since they only apply to windows images.

```
*****
* End of Procedure for Updating an Online Windows Installation *
*****
```

```
*****
* Automating the Installation of Drivers for Online Windows *
*****
```

If you followed the section [Get drivers: Export Method \(3 ways to export\)](#) to export drivers, you will have a folder with all your drivers. The **pnputil** command as shown below can install all those drivers with just one command.

```
pnputil /add-driver C:\Drivers\*.inf /subdirs /install
```

NOTE: Run the command from an elevated command prompt. The **/subdirs** will traverse all subdirectories. As a result, if you have a folder with many multiple drivers, each in their own subdirectory, you can install all of them with this one command.

If a driver is distributed in a .CAB file, ZIP file, EXE, etc. first extract it so that you have access to the .INF files. The below example shows how you can extract a file from a CAB file.

```
expand c:\Update\ASMEDIA.CAB -f:*.inf c:\Extracted
```

NOTE: The **"f:"** in the command is NOT a drive letter - don't change it.

Appendix A: Notes on Use of Microsoft Tools

When using tools such as **DISM**, **ImageX**, **OSCDIMG**, etc. you can potentially encounter errors if the wrong version of a utility is run. This is very easy to do because there are often multiple versions on a system.

A common error is the inability to dismount an image with **DISM**.

For any scripts that you write, I suggest that you either change directories to the location where the ADK version is located or specify the full path to the command. As an example, DISM is located here by default:

```
C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment  
Kit\Deployment Tools\amd64\DISM.
```

When you run these commands manually, I suggest that you run the "Deployment and Imaging Tools Environment" as Admin from your ADK installation. This places all of the utilities you need in the path, ensuring that they can be found, and that the correct version will be run.

NOTE: If you have any difficulties and need to start over, unmount any images and perform a cleanup. This is important because files that remain mounted will prevent you from working with the same folder names again because a stale mount is still present at that location. Here is an example:

First, you can check to see if there are any open DISM mounts like this:

```
DISM /Get-MountedImageInfo  
Then...  
DISM /Unmount-Image /MountDir:"C:\Project\Mount" /Discard  
DISM /Cleanup-WIM  
DISM /Cleanup-Mountpoints  
C:\windows\system32\takeown /f "C:\Project\*.*" /r /d y  
icacls "C:\Project\*.*" /T /grant %username%:F
```

Finally, make sure to clean out the recycle bin as this can still hold references to folders that will cause difficulties.

IMPORTANT: If you find that the above does not work and you cannot delete the folder where an image was mounted, log off, log back on, and then try the above again. If you still have difficulties, make sure you have cleaned out the recycle bin, reboot the system and try the above commands one last time.

Appendix B: Identifying the Disk ID to Which Windows Will be Installed

When creating an answer file that specifies the disk to which Windows will be installed, you may need to specify the disk to which Windows is to be installed. It is important to get this right since that disk will be wiped clean.

Complicating things is the fact that Windows setup will not always see disks with the same ID that a full installation of Windows does.

To determine the correct Disk ID, do this:

- 1) Boot from a Windows installation media (thumb drive, DVD, etc.) that has the same version of Windows that you plan to install. If you have added drivers to your Windows image, ensure that the media you are booting has the same added drivers.
- 2) When the boot pauses at the first static screen (the screen where regional information is requested of you), press **SHIFT + F10** to open a command prompt.
- 3) Run the command **diskpart**.
- 4) Once **diskpart** has started, run the command **list disk**.
- 5) From the size of the disks, you should be able to tell which the correct disk is. If that is not enough information for you then run these commands:

NOTE: Replace the "0" (zero) with the number of the disk you want to see details for.

```
select disk 0
detail disk
```

You can repeat the above two commands for additional disks. Just specify the disk number of interest in the first command.

- 6) When done, make sure to remember the disk number to which you want to install Windows and then run **exit**.

As an alternative you could temporarily disconnect all other disks than the one to which you want to install Windows on, or you could disable all other disk ports in the BIOS if your BIOS has such an option. This will ensure that the only disk seen by Windows setup is the one that you want to install Windows onto.

Appendix C: Using OSCDIMG to Create an ISO Image

Use the following syntax to create a bootable ISO image using the Microsoft OSCDIMG utility:

```
oscdimg -m -o -u2 -udfver102 -l"VolumeName" -  
bootdata:2#p0,e,b"c:\project\ISO_Files\boot\etfsboot.com"#pEF,e,b"c:\pr  
oject\ISO_Files\efi\microsoft\boot\efisys.bin" "c:\project\ISO_Files"  
"c:\destination\image.iso"
```

Please be careful with the above syntax. There are no spaces in a lot of the areas you might normally expect spaces.

In the above syntax, note the following:

VolumeName - This is the volume name to assign to the image. When burned to a DVD or Blu-Ray disc, the disc would show this name for the volume name. The maximum length of the volume name is 32 characters. To omit a volume name, you can use **-L""** or you can leave off the **-L "VolumeName"** parameter altogether.

c:\project\ISO_Files - this is the location where the Windows files are located.

c:\destination\image.iso - This is the full path and file name of the ISO image you want to create.

NOTE: For any paths that include spaces, make sure to include the path in double quote marks (").

Appendix D: Converting INSTALL.ESD files to INSTALL.WIM

Use this procedure only if necessary. My suggestion would be to download a Windows image that already has an **install.wim** rather than an **install.esd**. Please see [Appendix E](#) for instructions on how to accomplish this.

Copy the **install.esd** file to a temporary folder such as **C:\ISO_Files**. Run this command:

```
dism /Export-Image /SourceImageFile:C:\ISO_Files\Sources\install.esd  
/SourceIndex:1 /DestinationImageFile:C:\ISO_Files\Sources\install.wim  
/Compress:Max /CheckIntegrity
```

You can repeat the same export command with the index number changed to convert additional editions of Windows. Note that the first export command will take a while but the export commands for the additional indices will complete much quicker.

Appendix E: How to download Windows with INSTALL.WIM

Start by going to the Microsoft Software download page ([Software Download \(microsoft.com\)](https://www.microsoft.com/software-download/windows10)) and choose either Windows 10 or Windows 11 to go to the Windows Media Creation tool for that version of Windows. Continue with the steps below.

For Windows 10

Optional

If you want to see the build and version of Windows the site will make available, follow these steps. If you don't want to know this information, skip to "End of Optional Steps" below.

Select **Download tool now** and open it. Wait for the license terms to be displayed.

Go to `C:\$Windows.~WS\Sources` and open the file `products.xml` in notepad.

NOTE: The folder `$Windows.~WS` may be hidden. You may need to type in the path `C:\$Windows.~WS\Sources` in File Explorer in order to access that folder.

Look for any line that starts with `<FileName>`. Following that you will see the build number of the Windows image being made available by this web site.

Example: `<FileName>19043.928.210409-1212.21h1_release_svc_refresh_CLIENTCONSUMER_RET_x64FRE_zh-tw.esd</FileName>`

From this we can see that this is Windows 10 version 21H1, build 19043.928.

Close the Media Creation Tool Window (the one showing the license agreement). You will be asked if you are sure that you want to quit. Click on **Yes**.

End of Optional Steps

Back on the screen where you are given a choice to download the tool, press **F12**. If you are asked if you want to open the Dev Tools, choose to open them.

Press **CTRL + SHIFT + M** to toggle Device Emulation. You want to have device emulation turned on. You will know that device emulation is on if under the URL you see a line that lists a device type and screen resolution. In that left pane, under the URL, choose a non-Windows device such as "iPad Pro".

Refresh the browser page by pressing **CTRL-R**.

The displayed screen should change to show some different options.

Select the edition of Windows to download. For example, Windows 10, English, and then click on **Confirm**.

Choose 64-bit Download.

Save the image to the location of your choice.

This will be an ISO image with an `install.wim` rather than an `install.esd`.

Continue to the **IMPORTANT** note below following the Windows 11 instructions.

For Windows 11

Select "Download Windows 11 Disk Image (ISO)". This option will download an ISO image that uses an install.wim file rather than an install.esd.

To see the build currently being made available, from the Windows 11 download page, choose the option called "Create Windows 11 Installation Media". Follow the same steps as outlined in the "Optional" section for Windows 10 above.

IMPORTANT: On occasion, the ISO image version on the site lags the other downloads by a few days. After you download the ISO, you can do the following to verify the version:

- 1) Mount the ISO by double-clicking it. Note the drive letter it was mounted to,
- 2) Run the following command from an elevated Deployment and Imaging Tools Environment command prompt:

```
dism /Get-WimInfo /WimFile:d:\Sources\install.wim /index:1
```

You can get the build from the **Version** and **ServicePack Build** lines. In the example below we see the build 22000.438.

```
C:\>dism /Get-WimInfo /WimFile:d:\Sources\install.wim /index:1
```

```
Deployment Image Servicing and Management tool  
Version: 10.0.22000.1
```

```
Details for image : d:\Sources\install.wim
```

```
Index : 1  
Name : Windows 11 Home  
Description : Windows 11 Home  
Size : 16,150,264,540 bytes  
WIM Bootable : No  
Architecture : x64  
Hal : <undefined>  
Version : 10.0.22000  
ServicePack Build : 438  
ServicePack Level : 0  
Edition : Core  
Installation : Client  
ProductType : WinNT  
ProductSuite : Terminal Server  
System Root : WINDOWS  
Directories : 23431  
Files : 102928  
Created : 11/4/2021 - 8:33:09 AM  
Modified : 1/18/2022 - 6:33:36 AM  
Languages :  
    en-US (Default)
```

The operation completed successfully.

Appendix F: Performing a Compact OS Install

On a system with a very small amount of storage, for example, 32 GB or 64 GB, you may want to consider performing a Compact OS installation. This causes Windows to store a number of system files in a compressed state.

To deploy Windows as a Compact OS install, in your `autounattend.xml` answer file, navigate to **Pass 1 windowsPE > Setup > ImageInstall > OSImage**. Change the value of **Compact** to **true**. On a 64-bit installation of Windows, this can save about 2.6 GB of storage space.

You can determine if the system is running a Compact OS by running this command:

```
Compact /CompactOS:Query
```

You can also turn on Compact OS after Windows is installed. Do this by opening an elevated command prompt and then running this command:

```
Compact /CompactOS:Always
```

You can disable Compact OS with this command from an elevated command prompt:

```
Compact /CompactOS:Never
```

Appendix G: Creating Partitions That Meet Microsoft Recommendations During Unattended Setup

The standard method of creating partitions during unattended setup does not allow the recovery partition to be created last, however, Microsoft's latest recommendation is that you create that partition last.

We have noted that you can omit the disk configuration section. By doing this, setup will ask where you want to install Windows. After selecting the disk to which Windows should be installed, Windows setup proceeds unattended. This will cause the recovery partition to be created last, which is not possible through the standard means of specifying the disk information.

By following the steps below, you can still have 100% unattended installation and create the recovery partition last.

METHOD 1

IMPORTANT: These steps should be followed at the point in the procedure where you were directed to this appendix.

Start by adding the component **Microsoft Windows Setup > RunSynchronous > RunSynchronousCommand** to **Pass 1 windowsPE**.

Set the following values:

Order: 1 (see the note below regarding the Order numbers)
Path: cmd.exe /c echo select disk 0 >> X:\diskpartUEFI.txt

IMPORTANT: In the above command, we specify disk 0. Make sure to select the correct disk for your system because that disk will be wiped clean! If you are not certain, you can do one of the following:

- 1) Disconnect all other disks from the system temporarily until Windows is installed. This ensures that the disk to which you are installing will be disk 0 since it is the only disk present.
- 2) Boot from your Windows install media (without an answer file) and run diskpart to determine the disk ID of the disk to which you wish to install Windows as described in [Appendix B](#).

Repeat this process by continuing to add **RunSynchronousCommand** entries, incrementing the Order each time and adding the **Path** entry noted below for each entry. Note that you are adding a total of 16 entries here.

NOTE: If you already have **RunSynchronousCommand** entries, then start the **Order** numbers with the next higher number.

Continue to add the following entries:

Order: 2
Path: cmd.exe /c echo clean >> X:\diskpartUEFI.txt

Order: 3
Path: cmd.exe /c echo convert gpt >> X:\diskpartUEFI.txt

Order: 4
Path: cmd.exe /c echo CREATE PARTITION EFI SIZE=260 >> X:\diskpartUEFI.txt

Order: 5
Path: cmd.exe /c echo format quick fs=fat32 label="System" >>
X:\diskpartUEFI.txt

Order: 6
Path: cmd.exe /c echo create partition msr size=128 >> X:\diskpartUEFI.txt

Order: 7
Path: cmd.exe /c echo create partition primary >> X:\diskpartUEFI.txt

Order: 8
Path: cmd.exe /c echo shrink desired=500 minimum=500 >> X:\diskpartUEFI.txt

Order: 9
Path: cmd.exe /c echo format quick fs=ntfs label="Windows" >>
X:\diskpartUEFI.txt

Order: 10
Path: cmd.exe /c echo assign letter="W" >> X:\diskpartUEFI.txt

Order: 11
Path: cmd.exe /c echo create partition primary >> X:\diskpartUEFI.txt

Order: 12
Path: cmd.exe /c echo format quick fs=ntfs label="Recovery tools" >>
X:\diskpartUEFI.txt

Order: 13
Path: cmd.exe /c echo set id="de94bba4-06d1-4d40-a16a-bfd50179d6ac" >>
X:\diskpartUEFI.txt

Order: 14
Path: cmd.exe /c echo gpt attributes=0x8000000000000001 >>
X:\diskpartUEFI.txt

Order: 15
Path: cmd.exe /c echo exit >> X:\diskpartUEFI.txt

Order: 16
Path: cmd.exe /c diskpart.exe /s X:\DiskPartUEFI.txt

METHOD 2

IMPORTANT: These steps should be followed AFTER you have finalized your autounattend.xml answer file. You should go back to the point in the documentation that directed you here now, then, after you get to the point where you finalize the autounattend.xml answer file, follow the below steps.

Overview of the solution

We will make some modifications to the **autounattend.xml** answer file now. Later, when you are ready to create your final media, we will instruct you on how to create your media to work with the changes made here.

End of overview

Making the changes to the autounattend.xml

Begin by removing the **Setup > DiskConfiguration** section (just the **DiskConfiguration** section, **NOT Setup**) if you created it earlier. We no longer need it because we will use a script to create the partitions.

Add **Setup > RunSynchronous > RunSynchronousCommand** to **Pass 1 windowsPE**.

Set the following values:

For MBR disk only:

Order: 1
Path: X:\CreatePartitions-BIOS.bat

For GPT disk only:

Order: 1
Path: X:\CreatePartitions-UEFI.bat

For MBR disk only:

Make no other changes. Skip to the next section (Finalizing) below.

For GPT disk only:

Go to **Setup > ImageInstall > OSImage > InstallTo** and change **PartitionID** from 4 to 3.

Either mount the Windows ISO image file or connect the Windows media that you intend to use for installation to your computer. Copy the file contents to your HD. I'm going to assume **C:\Project\ISO_Files** as the location in my examples below.

Create three additional folders:

```
C:\Project\WinPE_Mount
C:\Project\WinPE
C:\Project\Assets
```

From **C:\Project\ISO_Files\Sources** copy the file **boot.wim** to **C:\Project\WinPE**:

```
copy /B C:\Project\ISO_Files\sources\boot.wim C:\Project\WinPE
```

The **boot.wim** file contains two indices. We only care about index **2** (Windows setup) so we're simply going to export the first index without making any changes.

```
dism /Export-Image /SourceImageFile:C:\Project\WinPE\boot.wim /SourceIndex:1
/DestinationImageFile:C:\Project\Assets\boot.wim
```

Mount index 2:

```
dism /mount-image /imagefile:C:\Project\WinPE\boot.wim /index:2
/mountdir:C:\Project\WinPE_Mount
```

Copy the below batch files to the folder **C:\Project\WinPE_Mount**.

NOTE: These batch files assume that Windows will be installed to **disk 0**. If that is not the case, please modify the batch file as needed.



CreatePartitions-BI
OS.bat



CreatePartitions-UE
FI.bat

NOTES: The batch files are what will now create the partitions on the HD rather than the entries that were previously specified in the answer file. This works because the **RunSynchronousCommand** we created to run the batch file is one of the first actions that setup takes. If you do not perform an unattended setup, the batch file will not affect anything so it can remain here perfectly safely.

If you examine the batch file, you will note that it simply contains a diskpart script. The magic sauce is in the fact that diskpart can shrink a partition after creating it. So, after we create the Windows partition to occupy the entire remaining space of the disk, we shrink it giving us space for the recovery tools partition which we then create.

END OF NOTES.

Cleanup the image, unmount it, mark it as being bootable, export it:

```
dism /Cleanup-Image /Image:"C:\Project\WinPE_Mount" /StartComponentCleanup
```

```
dism /Unmount-Image /MountDir:"C:\Project\WinPE_Mount" /Commit
```

```
dism /Export-Image /Bootable /SourceImageFile:C:\Project\WinPE\boot.wim  
/SourceIndex:2 /DestinationImageFile:C:\Project\Assets\boot.wim
```

You will now find that you have a file called **boot.wim** in **C:\Project\Assets**. This is the **boot.wim** with the added batch file which will be used by the modified autounattend.xml answer file. Simply copy the file back to the original location such as **C:\ISO_Files\Sources**

Done!

Appendix H - A Cleaner Windows Installation by Changing UserLocale

By setting the UserLocale to "En-001" rather than the usual (for example, "en-US"), the Microsoft Store app will be disabled. In addition, because app placeholders that are placed in start are dependent upon the region, none of these placeholders will be placed in Start. This results in an extremely clean Start screen. Once Windows has finished installation, you should do the following:

Open Settings > Time & language > Language & region. Under the "Region" section, change both the "Country or region" and "Regional format" settings to the settings that are correct for you.

The Microsoft Store app will now be enabled. You should open it and check for available updates.

Note that this procedure gives the same results as changing the setting for "Time and currency format" to "English (World)" on the first static setup screen of a manual installation. The only difference is that with the manual installation, you will see a screen during setup indicating that something went wrong, but you can simply select "skip" on that screen.

Appendix I – Using STARTNET.CMD to script operations at startup

Windows PE can startup in one of 3 ways:

- 1) Run setup.exe. This will not allow us to run any outside scripts. This requires us to insert Run commands into the answer file.
- 2) Run startnet.cmd which can be used to run scripted commands. To do this sources\setup.exe will need to be removed or renamed as noted below.
- 3) Run applications from winpeshl.ini

To use STARTNET.CMD (item 2 above), rename the SETUP.EXE. In this example, we will use HIDE_SETUP.exe.

NOTE: Find setup.exe and startnet.cmd within the boot.wim file.

Now modify the startnet.cmd, adding lines AFTER the "wpeinit". At the end, you can run the renamed setup.exe file (in this example HIDE_setup.exe) which can run your answer file. Here is an example:

```
wpeinit
for %%A in (D E F G H I J K L M N O P Q R S T U V W X Y Z) do if exist %%A:\sources
set DRIVE=%%A
diskpart /s %DRIVE%\format.txt
%DRIVE%\sources\HIDE_setup.exe /unattend:%DRIVE%\autounattend.xml
```

Credits

This document is mainly based upon the tutorials of TenForums members Kari, Cereberus, and Brink. User johngalt was also extraordinarily helpful in helping me to devise a solution to creating partitions during unattended setup that follow the latest guidance from Microsoft. A lively discussion with him on TenForums kept me pushing forward until a solution was found.

I've tried to credit below those tutorials I had in my notes, but there are surely others that I am missing from which valuable nuggets have been gleaned. When I am deep into research, I don't always properly note every resource I find so my sincere apologies for any sources that I may have missed.

Windows Monthly Updates Explained

<https://techcommunity.microsoft.com/t5/windows-it-pro-blog/windows-monthly-updates-explained/ba-p/3773544>

OSBUILDER - Offline Servicing

<https://osbuilder.osdeploy.com/docs/untitled-4/offline-servicing>

How to create install media for completely automated unattended install of Windows 10

<https://www.tenforums.com/tutorials/96683-create-media-automated-unattended-install-windows-10-a.html>

How to apply an unattended answer file to offline Windows 10 image (USB or ISO)

<https://www.tenforums.com/tutorials/131765-apply-unattended-answer-file-windows-10-install-media.html>

How to Create a Bootable ISO file Containing Multiple Windows 10 Images

<https://www.tenforums.com/tutorials/133098-dism-create-bootable-iso-multiple-windows-10-images.html>

Personalization of Windows in a Hyper-V VM

<https://www.tenforums.com/virtualization/142357-personalization-possible-pre-activation-vmware-not-hyper-v-why.html>

How to create a Windows 10 ISO image for clean, in-place upgrade and repair install

<https://www.tenforums.com/tutorials/72031-create-windows-10-iso-image-existing-installation.html>

How to Create and Customize a Windows 10 Image in Audit Mode with Sysprep Tool

<https://www.tenforums.com/tutorials/3020-customize-windows-10-image-audit-mode-sysprep.html>

Create bootable USB installer if install.wim is greater than 4GB

<https://www.tenforums.com/tutorials/118137-create-bootable-usb-installer-if-install-wim-greater-than-4gb.html>

Available languages for Windows

<https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/available-language-packs-for-windows>

List of Generic Product Keys to Install Windows 10 Editions

<https://www.tenforums.com/tutorials/95922-generic-product-keys-install-windows-10-editions.html>

Windows Media Creation Tool

<https://www.microsoft.com/en-us/software-download/windows10>

Download and install the Windows ADK

<https://docs.microsoft.com/en-us/windows-hardware/get-started/adk-install>

Microsoft Update Catalog

<https://www.catalog.update.microsoft.com>

Windows 10 Update History

<https://support.microsoft.com/en-us/help/4529964/windows-10-update-history>

Windows 11 Update History

<https://support.microsoft.com/en-us/topic/windows-11-update-history-a19cd327-b57f-44b9-84e0-26ced7109ba9>

List of Windows Servicing Stack Updates (SSU)

<https://portal.msrc.microsoft.com/en-us/security-guidance/advisory/ADV990001>

Unattended Windows Setup Reference

<https://docs.microsoft.com/en-us/windows-hardware/customize/desktop/unattend/>

DISM Command-Line Options

<https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/deployment-image-servicing-and-management--dism--command-line-options>

Oscdimg Command-Line Options

<https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/oscimg-command-line-options>

BCDEdit Command-Line Options

<https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/bcdedit-command-line-options>

Microsoft Blog: Updating Windows 10 media with Dynamic Update packages

IMPORTANT: This blog is updated by the info in the next link

<https://techcommunity.microsoft.com/t5/windows-it-pro-blog/updating-windows-10-media-with-dynamic-update-packages/ba-p/982477>

Update Windows installation media with Dynamic Update

<https://learn.microsoft.com/en-us/windows/deployment/update/media-dynamic-update>

Update Windows installation media with Dynamic Update

<https://docs.microsoft.com/en-us/windows/deployment/update/media-dynamic-update>

The benefits of Windows 10 Dynamic Update

<https://techcommunity.microsoft.com/t5/windows-it-pro-blog/the-benefits-of-windows-10-dynamic-update/ba-p/467847>

Deploy Windows SSUs and LCUs together with one cumulative update

<https://techcommunity.microsoft.com/t5/windows-it-pro-blog/deploy-windows-ssus-and-lcus-together-with-one-cumulative-update/ba-p/1967887>

Create a Hardware Independent System Image

<https://www.tenforums.com/tutorials/2113-system-image-create-hardware-independent-system-image.html>

UEFI/GPT-based hard drive partitions

<https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/configure-uefigpt-based-hard-drive-partitions>

How to Native Boot to VHD of Hyper-V Virtual Machine

<https://www.tenforums.com/tutorials/53256-hyper-v-native-boot-vhd.html>

How to Create VHD of Windows 10 Installation and Use in Hyper-V

<https://www.tenforums.com/tutorials/2206-hyper-v-create-use-vhd-windows-10-disk2vhd.html>

Sample deployment scripts

<https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/windows-deployment-sample-scripts-sxs>

Windows 10 Unattended install media

<https://win10.guru/windows-10-unattended-install-media-part-1-basics/>

Dual Boot the easy way

<https://www.tenforums.com/general-support/5361-windows-videos-126.html#post1250189>

Intel Microcode Update Information

<https://www.tenforums.com/windows-10-news/168114-kb4589212-intel-microcode-updates-windows-10-v2004-v20h2-nov-10-a.html>

Apply Windows Image using DISM Instead of Clean Install

<https://www.tenforums.com/tutorials/84331-apply-windows-image-using-dism-instead-clean-install.html>