Yi's Solutions

Microsoft Windows Server 2022

Different system versions have different packaging methods. The packaging process includes: "Language pack: add, associate, delete", "Drive: add, delete", "Cumulative update: add, delete" etc.

There are many hidden stories hidden behind this. If you want to unlock these, are you ready to start trying to encapsulate them?

Summary

Table of contents

Chapter 1 Encapsulation

A. Prerequisites

I. Running system

1. Check if the running system is healthy

Checking the health of the running system is crucial. If the scan indicates that repairs are needed, adding cumulative updates to the offline image and performing other operations will result in errors. To check the system health, run: `sfc /scannow`

After the verification process is complete, if problems were found and corrected, you will see something similar to the following in the command prompt window:

1.1. Windows Resource Protection did not find any integrity violations.

Pending: No repair needed.

1.2. Windows Resource Protection found corrupted files but was unable to repair some of them.

Pending: Requires repair. You can try repairing it using the following command:

```
Dism /Online /Cleanup-Image /CheckHealth

Dism /Online /Cleanup-Image /ScanHealth

dism /Online /Cleanup-Image /RestoreHealth
```

Note: If the problem cannot be fixed, please reinstall the system.

2. When using DISM commands

2.1. Learning the DISM Overview

https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/what-is-dism

DISM Limitations: When using DISM to maintain Windows images, administrators must be aware of its inherent limitations. One of these is that it does not support the Resilient File System (ReFS). ReFS is a new file system for Windows Server 2022, Windows Server 2019, Windows Server 2016, Windows Server 2012 R2, and Windows Server 2012 operating systems. It scales efficiently to large datasets and provides higher data availability and performance than older file systems.

Please be aware that performing certain DISM commands after partitioning a disk and mounting an offline image to an ReFS disk partition may cause errors. Disk partitions formatted with NTFS are not affected by this issue.

When the operating system is Windows 10 or earlier than Windows 11 25H2, using the DISM command to create a higher version image may cause some unknown problems in certain situations. For example, when running the DISM command in Windows 10 to process a Windows Server 2025 offline image, you might receive an error message during the packaging process: "This app can't run on your PC." Solution:

1) Upgrade the currently running operating system or reinstall to a higher version (recommended);

2) Upgrading or installing a new version of ADK or PowerShell (not recommended)

- You can try upgrading to the latest version of PowerShell 7 or later;
- Installing the latest version of ADK and replacing the DISM command can resolve the issue of an outdated DISM version,

however: the deployment script primarily uses PowerShell commands, so we do not recommend using the above method. The best solution is to upgrade your current operating system or reinstall a higher version.

3. N ways to speed up Windows operating system

When processing packaging tasks, installing cumulative updates, installing drivers, and installing applications in InBox Apps, a large number of temporary files will be generated. The following methods can be used to speed up the system:

3.1. Turn off Windows Defender

- Turning on Windows Defender will scan files and take up a lot of CPU.
- During the test: It took 1 hour and 22 minutes before it was turned off, and 20 minutes after it was turned off.

How to turn off:

Green is the command line, hold down the Windows key and press R to launch Run.

3.1.1. Open Windows Security or run: windowsdefender:

3.1.2. Select "Virus & threat protection" or run: windowsdefender://threat

3.1.3. Find "Virus and Threat Protection Settings", click "Manage Settings" or run: windowsdefender://threatsettings. It is recommended that you turn off some features:

3.1.3.1. Real-time protection

3.1.3.2. Cloud-provided protection

3.1.3.3. Automatically submit samples

3.1.3.4. Tamper Protection

3.1.4. When not in the package, it is recommended that you turn on Windows Defender.

3.2. Disable Device/Credential Guard on your computer

Even with Windows Security Center disabled, if virtualization-based security is running, the system performance will be significantly impacted. Since Microsoft updated Windows 11 to version 24H2, Device/Credential Guard (or rather, VBS) is enabled by default with every clean installation or update. This causes problems with VMware Workstation and other virtualization software that use nested virtualization, leading to malfunctions and performance degradation.

However, Microsoft provides a tool called the Device Guard and Credential Guard hardware readiness tool to disable Device/Credential Guard. Here's how to disable it:

3.2.1. Download Device Guard and Credential Guard hardware readiness tool

Open the link: https://www.microsoft.com/en-us/download/details.aspx?id=53337 and click the download button. After downloading, you will get the dgreadiness_v3.6.zip file. Extract it to the C drive (if you extract it to another location, please replace the path when executing the program).

3.2.2. Execute the shutdown command

Open the Terminal and run the following commands in order:

```
Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope LocalMachine -Force
```

```
C:\dgreadiness_v3.6\DG_Readiness_Tool_v3.6.ps1 -Disable
```

3.2.3. After running the program, restart your computer

After restarting, Windows will enter the Credential Guard Opt-out Tool. At this point, press the Windows logo key or F3, then press Enter to continue.

3.2.4. Check status

After restarting your computer, run Msinfo32 and check the "Virtualization-based security" status in the System Summary, or check the status after running the program:

```
C:\dgreadiness_v3.6\DG_Readiness_Tool_v3.6.ps1 -Ready
```

3.2.5. How to recover

After running the command line, restart your computer for the changes to take effect.

```
C:\dgreadiness_v3.6\DG_Readiness_Tool_v3.6.ps1 -Enable
```

The Device Guard and Credential Guard hardware readiness tool cannot be disabled, or you can try using the tool from LDPlayer emulator: https://res.ldmnq.com/test/HyperV-off.exe. Download it and run it as an administrator, then restart your computer after completion.

3.3. Enable the native Windows NVMe storage stack.

Read the official Microsoft news: Announcing Native NVMe in Windows Server 2025: Ushering in a New Era of Storage Performance

Enabling the native Windows NVMe storage stack can deliver up to an 80% increase in IOPS performance. If you are using an NVMe drive, it is recommended that you enable this feature. Before enabling, please ensure you meet the following requirements: Windows Server 2025 or Windows 11 build 26100.6899 / 26200.6899 or later. This feature is only effective for NVMe SSDs; please check if your current hard drive meets the requirements.

Activation steps:

3.3.1. Ensure that the currently used storage controller is the standard NVM Express controller included with Windows. If you are currently using a driver other than the one included with Windows, it is recommended that you uninstall it.

3.3.2. Ensure that the currently used operating system version is 26100.6899 / 26200.6899 or a higher version.

3.3.3. The following two methods (from Microsoft's official website, essentially the same operation, choose either one)

3.3.3.1. Run PowerShell (as administrator), execute the following command, and then restart your computer.

```
reg add
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Policies\Microsoft\FeatureManagement\Overrides /v
1176759950 /t REG_DWORD /d 1 /f
```

3.3.3.2. Alternatively, download, extract, and install the following Group Policy MSI file, set it to enabled, and then restart your computer.

https://download.microsoft.com/download/123547b0-bff7-419d-96bad1cfee92f442/Windows%2011%2024H2,%20Windows%2011%2025H2%20and%20Windows%20Server%202025%20KB5066835%20251014_21251%20Feature%20Preview.msi

After installation, it can be found at: Group Policy > Computer Configuration > Administrative Templates > KB5066835 251014_21251 Feature Preview > Windows 11, version 24H2, 25H2

3.3.4. The method for enabling this feature in Windows 11 is slightly different from Windows Server, and requires adding and executing the following commands in PowerShell (as administrator):

```
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Policies\Microsoft\FeatureManagement\Overrides /v 1853569164 /t REG_DWORD /d 1 /f
```

```
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Policies\Microsoft\FeatureManagement\Overrides /v 156965516 /t REG_DWORD /d 1 /f
```

```
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Policies\Microsoft\FeatureManagement\Overrides /v 735209102 /t REG_DWORD /d 1 /f
```

3.3.5. If you have already used the above commands and restarted your computer but are still unable to enable the feature, you can also use the ViveTool command to force enable it.

You need ViveTool. Go to: https://github.com/thebookisclosed/ViVe/releases to download it. After downloading, run the program:

```
vivetool.exe /enable /id:48433719,49453572,5536923
```

For values above 28020, the following steps are also required: `vivetool.exe /enable /id:48613417`

3.3.6. Troubleshooting issues preventing access to Safe Mode.

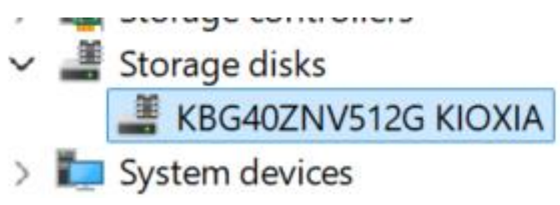
Windows Safe Mode does not load this NVMe driver by default, so you need to execute the following command to load it. After it takes effect, Safe Mode will work normally.

```
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\{75416E63-5912-4DFA-AE8F-3EFACCAFFB14}
```

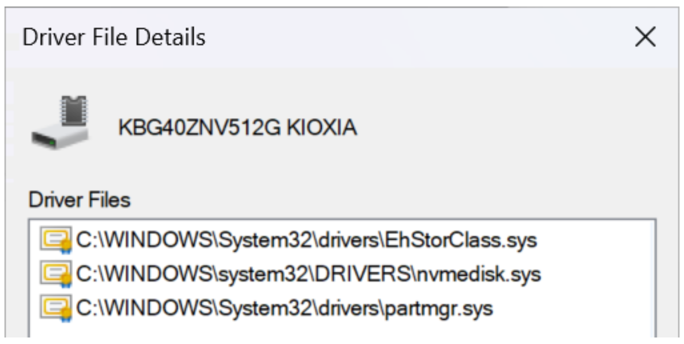
```
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\{75416E63-5912-4DFA-AE8F-3EFACCAFFB14}
```

3.3.7. The method for checking if this feature has been successfully enabled is the same as the method for disabling it.

3.3.7.1. After successfully enabling Windows native NVMe mode, your NVMe SSD will be categorized under the new "Storage disks" path in Device Manager, as shown in the image:



3.3.7.2. After successfully enabling Windows native NVMe mode, in Device Manager, right-click on your NVMe SSD and select Properties, then go to Driver > Driver Details. You will see that the driver files include the new nvmedisk.sys, as shown in the image:



Driver File Details

KBG40ZNV512G KIOXIA

Driver Files

C:\WINDOWS\System32\drivers\EhStorClass.sys
C:\WINDOWS\system32\DRIVERS\nvmedisk.sys
C:\WINDOWS\System32\drivers\partmgr.sys

3.3.7.3. Closing method

Run regedit (Registry Editor), navigate to the path HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Policies\Microsoft\FeatureManagement\Overrides, change the values of the four keys 1176759950, 156965516, 1853569164, and 735209102 to 0, or simply delete them. Then restart your computer.

Enabled using Vivetool `vivetool.exe /disable /id:48433719,49453572,5536923`

For values above 28020, the following steps are also required: `vivetool.exe /disable /id:48613417`

4. Disable security warnings for common file types

- Why is it closing? A "security warning" pops up every time I try to work with any ISO file, which is very annoying.
- After adding or restoring the default policies, remember to restart the Explorer.exe process for the changes to take effect immediately.

4.1. How to add it?

4.1.1. Step one: Select the rules.

Common file type rules and a rule specifically for *.ISO files are provided:

4.1.1.1. Common file type rules

```
$SafetyWarningsExclude = ".exe;.reg;.msi;.bat;.cmd;.com;.vbs;.hta;.scr;.pif;.js;.iso;"
```

4.1.1.2. .iso files only

```
$SafetyWarningsExclude = ".iso;"
```

4.1.2. Step two: Add it to the system policy.

```
if ((Test-Path -LiteralPath "HKCU:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Associations") -ne $true) { New-Item "HKCU:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Associations" -force -ErrorAction SilentlyContinue | Out-Null }; New-ItemProperty -LiteralPath 'HKCU:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Associations' -Name 'LowRiskFileTypes' -Value $SafetyWarningsExclude -PropertyType String -force -ErrorAction SilentlyContinue | Out-Null
```

4.2. How to restore the default policy?

```
Remove-Item -Path "HKCU:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Associations" -Force -Recurse -ErrorAction SilentlyContinue | Out-Null
```

4.3. Restart the Explorer.exe process for the changes to take effect immediately.

```
Stop-Process -ProcessName explorer -force -ErrorAction SilentlyContinue; Start-Sleep 5; $Running = Get-Process explorer -ErrorAction SilentlyContinue; if (-not ($Running)) { Start-Process "explorer.exe" }
```

5. Virtual memory disk

What is a memory disk? Memory disk, also known as virtual memory disk, is a technology that improves quick access to computer memory and files. However, the memory disk will cause data loss after the computer is shut down. The memory disk is a relatively unsafe setting.

Even so, I don't think so. During the encapsulation process, installation package files will be frequently released, logs will be generated, etc. When mounting to a virtual disk, this has many benefits, including quick formatting.

When adding language packs, cumulative updates, and InBox Apps, the installation package is stored in a memory virtual disk, which will occupy a large amount of memory. It is recommended that you store it in a non-virtual memory disk.

5.1. Software recommendation

5.1.1. Ultra RAMDisk | http://ultraramdisk.com

5.1.2. ImDisk | https://sourceforge.net/projects/imdisk-toolkit

5.1.3. AMD Radeon RAMDisk | http://www.radeonramdisk.com

5.1.4. Primo Ramdisk | https://www.romexsoftware.com/en-us/primo-ramdisk/index.html

5.1.5. SoftPerfect RAM Disk | https://www.softperfect.com/products/ramdisk

5.1.6. StarWind RAM Disk | https://www.starwindsoftware.com/high-performance-ram-disk-emulator

5.2. How to create

When creating a memory disk, you should calculate the unused rate of physical memory, open "Task Manager", "Performance", and check the remaining memory rate. Suggestions:

5.2.1. When the physical memory is 16G and the system has 10G remaining, it is recommended to divide it into: 6G memory + 40G swap file, and keep more than 4G of remaining memory;

5.2.2. When the physical memory is 32G: When the system has 26G remaining, it is recommended to divide it into: 20G memory + 40G swap file, and keep more than 6G of remaining memory;

5.2.3. When the physical memory is 64G: When the system has 54G remaining, only divide 40G memory, no need to create a swap file, and keep more than 8G of remaining memory;

5.2.4. When the physical memory is 128G: When the system has 115G remaining, divide it into 40G memory, and no need to create a swap file.

Note: Insufficient memory can cause problems during the encapsulation process.

6. Command line

6.1. Optional "Terminal" or "PowerShell ISE", if "Terminal" is not installed, please go to: https://github.com/microsoft/terminal/releases After downloading;

6.2. Open "Terminal" or "PowerShell ISE" as administrator, it is recommended to set the PowerShell execution policy: bypass, PS command line:

```
Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope LocalMachine -Force
```

6.3. In this article, PS command line, green part, please copy it, paste it into the "Terminal" dialog box, press Enter and start running;

6.4. When there is .ps1, right-click the file and select Run with PowerShell, or copy the path and paste it into Terminal to run, the path with a colon, add the & character in the command line, example: & "D:\YiSolutions_Encapsulation_SIP.ps1"

II Requirements

1. System installation package

1.1. Prepare to download the initial release or developer version

1.1.1. x64

1.1.1.1. Filename: en-us_windows_server_2022_x64_dvd_620d7eac.iso

List of files: https://files.rg-adguard.net/file/9a0f4eb7-c3a9-e46b-3fc8-cdb71289dbfb

1.2. For example, after downloading en-us_windows_server_2022_x64_dvd_620d7eac.iso, extract it to: D:\en-us_windows_server_2022_x64_dvd_620d7eac

Note: Before decompressing to disk D, you should check whether it is a ReFS partition. If it is a ReFS partition, some DISM commands will fail. Solution: Please use a disk partition in NTFS format.

1.3. After decompression, change the directory D:\en-us_windows_server_2022_x64_dvd_620d7eac to D:\ISOTEMP

1.4. All scripts and all paths have been set to D:\ISOTEMP by default as the image source.

2. Language Pack

2.1. Learn

2.1.1. Add languages to a Windows 11 image

2.1.2. Language and region Features on Demand (FOD)

2.1.2.1. Fonts

- When adding a language pack, when the corresponding region is triggered, the required font functions need to be added, download "List of all available language FODs" learn more.
- In "Language package: extract", the automatic recognition function has been added, and you can understand the functions: Function Match_Required_Fonts

2.1.2.2. Regional association

What are regional connections?

- When the image language is only in English, after adding the zh-HK language pack, the image language will not be added. You should install zh-TW first, and then install zh-HK to obtain the corresponding association.

- Please refer to Microsoft's official original version: Windows 10, Windows 11 Traditional Chinese version.

Known regional associations:

2.1.2.2.1. Region: zh-TW, Optional associated areas: zh-HK

2.2. Language pack: Download

2.2.1. Filename: https://software-download.microsoft.com/download/sg/20348.1.210507-1500.fe_release_amd64fre_SERVER_LOF_PACKAGES_OEM.iso

List of files: https://files.rg-adguard.net/file/f4a036a7-5c8e-6bd6-764a-83655c1a9ce5

B. Language package: extract

I. Language pack: Ready

Mounted 20348.1.210507-1500.fe_release_amd64fre_SERVER_LOF_PACKAGES_OEM.iso or unzip it to any location;

II. Language pack: Extract scheme

1. Add

1.1. Language name: Simplified Chinese - China, language tag: zh-CN, Scope of application: Install.Wim, Boot.Wim, WinRE.Wim

2. Delete

2.1. Language name: English - United States, language tag: en-US, Scope of application: Install.Wim, Boot.Wim, WinRE.Wim

III. Execute the extract command

- Auto = automatically search all local disks, default;
- Customize the path, for example, specify the E drive: $ISO = "E:\"
- Extract.ps1
  - \Expand\Extract.ps1
  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Extract.ps1
- Copy the code

```
$ISO = "Auto"; $SaveTo = "D:\ISOTEMP_Custom";
Function Extract_Language {
  param( $Act, $NewLang, $Expand )
  Function Match_Required_Fonts {
    param( $Lang )
    $Fonts = @( @{ Match = @("as", "ar-SA", "ar", "ar-AE", "ar-BH", "ar-DJ", "ar-DZ", "ar-EG", "ar-ER", "ar-IL", "ar-IQ", "ar-JO", "ar-KM", "ar-KW", "ar-LB", "ar-LY", "ar-MA", "ar-MR", "ar-OM", "ar-PS", "ar-QA", "ar-SD", "ar-SO", "ar-SS", "ar-SY", "ar-TD", "ar-TN", "ar-YE", "arz-Arab", "ckb-Arab", "fa", "fa-AF", "fa-IR", "glk-Arab", "ha-Arab", "ks-
Arab", "ks-Arab-IN", "ku-Arab", "ku-Arab-IQ", "mzn-Arab", "pa-Arab", "pa-Arab-PK", "pnb-Arab", "prs", "prs-AF", "prs-Arab", "ps", "ps-AF", "sd-Arab", "sd-Arab-PK", "tk-Arab", "ug", "ug-Arab", "ug-CN", "ur", "ur-IN", "ur-PK", "uz-Arab", "uz-Arab-AF"); Name = "Arab"; }; @{ Match = @("bn-IN", "as-IN", "bn", "bn-BD", "bpy-Beng"); Name = "Beng"; }; @{ Match = @("da-dk", "iu-Cans", "iu-Cans-CA"); Name = "Cans"; }; @{ Match = @("chr-Cher-US", "chr-Cher"); Name = "Cher"; }; @{ Match = @("hi-IN", "bh-Deva", "brx", "brx-Deva", "brx-IN", "hi", "ks-Deva", "mai", "mr", "mr-IN", "ne", "ne-IN", "ne-NP", "new-Deva", "pi-Deva", "sa", "sa-Deva", "sa-IN"); Name = "Deva"; }; @{ Match = @("am", "am-ET", "byn", "byn-ER", "byn-Ethi", "ti", "ti-ER", "ti-ET", "tig", "tig-ER", "tig-Ethi", "ve-Ethi", "wal", "wal-ET", "wal-Ethi"); Name = "Ethi"; }; @{ Match = @("gu", "gu-IN"); Name = "Gujr"; }; @{ Match = @("pa", "pa-IN", "pa-Guru"); Name = "Guru"; }; @{ Match = @("zh-CN", "cmn-Hans", "gan-Hans", "hak-Hans", "wuu-Hans", "yue-Hans", "zh-gan-Hans", "zh-hak-Hans", "zh-Hans", "zh-SG", "zh-wuu-Hans", "zh-yue-Hans"); Name = "Hans"; }; @{ Match = @("zh-TW", "cmn-Hant", "hak-Hant", "lzh-Hant", "zh-hak-Hant", "zh-Hant", "zh-HK", "zh-lzh-Hant", "zh-MO", "zh-yue-Hant"); Name = "Hant"; }; @{ Match = @("he", "he-IL", "yi"); Name = "Hebr"; }; @{ Match = @("ja", "ja-JP"); Name = "Jpan"; }; @{ Match = @("km", "km-KH"); Name = "Khmr"; }; @{ Match = @("kn", "kn-IN"); Name = "Knda"; }; @{ Match = @("ko", "ko-KR"); Name = "Kore"; }; @{ Match = @("de-de", "lo", "lo-LA"); Name = "Laoo"; }; @{ Match = @("ml", "ml-IN"); Name = "Mlym"; }; @{ Match = @("or", "or-IN"); Name = "Orya"; }; @{ Match = @("si", "si-LK"); Name = "Sinh"; }; @{ Match = @("tr-tr", "arc-Syrc", "syr", "syr-SY", "syr-Syrc"); Name = "Syrc"; }; @{ Match = @("ta", "ta-IN", "ta-LK", "ta-MY", "ta-SG"); Name = "Taml"; }; @{ Match = @("te", "te-IN"); Name = "Telu"; }; @{ Match = @("th", "th-TH"); Name = "Thai"; }; )
    ForEach ($item in $Fonts) { if (($item.Match) -Contains $Lang) { return $item.Name; }; }
    return "Not_matched"
  }
  Function Match_Other_Region_Specific_Requirements {
    param( $Lang )
    $RegionSpecific = @( @{ Match = @("zh-TW"); Name = "Taiwan"; }; )
    ForEach ($item in $RegionSpecific) { if (($item.Match) -Contains $Lang) { return $item.Name; }; }
    return "Skip_specific_packages"
  }
  Function Extract_Process {
    param( $Package, $Name, $NewSaveTo )
    $NewSaveTo = "$($SaveTo)\$($NewSaveTo)\Language\$($Act)\$($NewLang)"; New-Item -Path $NewSaveTo -ItemType Directory -ErrorAction SilentlyContinue | Out-Null;
    if ($ISO -eq "Auto") {
        Get-PSDrive -PSProvider FileSystem -ErrorAction SilentlyContinue | ForEach-Object {
            ForEach ($item in $Package) { $TempFilePath = Join-Path -Path $_.Root -ChildPath $item -ErrorAction SilentlyContinue; if (Test-Path $TempFilePath -PathType Leaf) { Write-host "`n   Find: " -NoNewLine; Write-host $TempFilePath -ForegroundColor Green; Write-host "   Copy to: " -NoNewLine; Write-host $NewSaveTo; Copy-Item -Path $TempFilePath -Destination $NewSaveTo -Force; }; }
        }
    } else {
        ForEach ($item in $Package) { $TempFilePath = Join-Path -Path $ISO -ChildPath $item -ErrorAction SilentlyContinue; Write-host "`n   Find: " -NoNewline; Write-host $TempFilePath -ForegroundColor Green; if (Test-Path $TempFilePath -PathType Leaf) { Write-host "   Copy to: " -NoNewLine; Write-host $NewSaveTo; Copy-Item -Path $TempFilePath -Destination $NewSaveTo -Force; } else { Write-host "   Not found"; }}
    }
    Write-host "`n   Verify the language pack file"
    ForEach ($item in $Package) { $Path = "$($NewSaveTo)\$([IO.Path]::GetFileName($item))"; if (Test-Path $Path -PathType Leaf) { Write-host "   Discover: " -NoNewLine; Write-host $Path -ForegroundColor Green; } else { Write-host "   Not found: " -NoNewLine; Write-host $Path -ForegroundColor Red; }}
  }
  $AdvLanguage = @(
    @{ Path = "Install\Install"; Rule = @( "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-Fonts-{DiyLang}-Package~31bf3856ad364e35~amd64~~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-Server-Language-Pack_x64_{Lang}.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-Basic-{Lang}-Package~31bf3856ad364e35~amd64~~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-Handwriting-{Lang}-Package~31bf3856ad364e35~amd64~~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-OCR-{Lang}-Package~31bf3856ad364e35~amd64~~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-Speech-{Lang}-Package~31bf3856ad364e35~amd64~~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-TextToSpeech-{Lang}-Package~31bf3856ad364e35~amd64~~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-MSPaint-FoD-Package~31bf3856ad364e35~amd64~{Lang}~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-MSPaint-FoD-Package~31bf3856ad364e35~wow64~{Lang}~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-Notepad-FoD-Package~31bf3856ad364e35~amd64~{Lang}~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-Notepad-FoD-Package~31bf3856ad364e35~wow64~{Lang}~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~{Lang}~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-PowerShell-ISE-FOD-
Package~31bf3856ad364e35~wow64~{Lang}~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-StepsRecorder-
Package~31bf3856ad364e35~amd64~{Lang}~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-StepsRecorder-
Package~31bf3856ad364e35~wow64~{Lang}~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-WordPad-FoD-
Package~31bf3856ad364e35~amd64~{Lang}~.cab"; "LanguagesAndOptionalFeatures\Microsoft-Windows-WordPad-FoD-
Package~31bf3856ad364e35~wow64~{Lang}~.cab"; ) }
    @{ Path = "Install\\WinRE"; Rule = @( "Windows Preinstallation Environment\x64\WinPE_OCs\WinPE-FontSupport-{Lang}.cab"; "Windows Preinstallation
Environment\x64\WinPE_OCs\{Lang}\lp.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-securestartup_{Lang}.cab"; "Windows
Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-atbroker_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-
audiocore_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-audiodrivers_{Lang}.cab"; "Windows Preinstallation
Environment\x64\WinPE_OCs\{Lang}\winpe-enhancedstorage_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-narrator_{Lang}.cab";
"Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-scripting_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-
speech-tts_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-srh_{Lang}.cab"; "Windows Preinstallation
Environment\x64\WinPE_OCs\{Lang}\winpe-srt_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-wds-tools_{Lang}.cab"; "Windows
Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-wmi_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-
appxpackaging_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-storagewmi_{Lang}.cab"; "Windows Preinstallation
Environment\x64\WinPE_OCs\{Lang}\winpe-wifi_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-rejuv_{Lang}.cab"; "Windows
Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-opcservices_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-
hta_{Lang}.cab"; )}
    @{ Path = "Boot\Boot"; Rule = @( "Windows Preinstallation Environment\x64\WinPE_OCs\WinPE-FontSupport-{Lang}.cab"; "Windows Preinstallation
Environment\x64\WinPE_OCs\{Lang}\lp.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\WinPE-Setup_{Lang}.cab"; "Windows Preinstallation
Environment\x64\WinPE_OCs\{Lang}\WINPE-SETUP-Server_{Lang}.CAB"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-
securestartup_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-atbroker_{Lang}.cab"; "Windows Preinstallation
Environment\x64\WinPE_OCs\{Lang}\winpe-audiocore_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-audiodrivers_{Lang}.cab";
"Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-enhancedstorage_{Lang}.cab"; "Windows Preinstallation
Environment\x64\WinPE_OCs\{Lang}\winpe-narrator_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-scripting_{Lang}.cab";
"Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-speech-tts_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-
srh_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-srt_{Lang}.cab"; "Windows Preinstallation
Environment\x64\WinPE_OCs\{Lang}\winpe-wds-tools_{Lang}.cab"; "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-wmi_{Lang}.cab"; ) }
  )
  $NewFonts = Match_Required_Fonts -Lang $NewLang; $SpecificPackage = Match_Other_Region_Specific_Requirements -Lang $NewLang;
  Foreach ($item in $Expand) {
    $Language = @()
    Foreach ($itemList in $AdvLanguage) {
      if ($itemList.Path -eq $item) {
        Foreach ($PrintLang in $itemList.Rule) { $Language += "$($PrintLang)".Replace("{Lang}", $NewLang).Replace("{DiyLang}", $NewFonts).Replace("{Specific}",
$SpecificPackage); }
        Extract_Process -NewSaveTo $itemList.Path -Package $Language -Name $item
      }
    }
  }
}
$Extract_language_Pack = @(
  @{ Tag = "zh-CN"; Act = "Add"; Scope = @( "Install\Install"; "Install\\WinRE"; "Boot\Boot" ) }
  @{ Tag = "en-US"; Act = "Del"; Scope = @( "Install\Install"; "Install\\WinRE"; "Boot\Boot" ) }
)
```

```
ForEach ($item in $Extract_language_Pack) { Extract_Language -Act $item.Act -NewLang $item.Tag -Expand $item.Scope }
```

C. Custom encapsulation

I. Custom encapsulation: Install.wim

1. View Install.wim details

Before mounting, you should analyze the content structure of the Windows image. It usually contains multiple different versions (such as Home, Enterprise, Education, etc.). You can use the view command to display: image name, image description, image size, architecture, version, index number, etc. Then you can customize the required "index number" for mounting.

```
$ViewFile = "D:\ISOTEMP\Sources\Install.wim"; Get-WindowsImage -ImagePath $ViewFile | Foreach-Object { Get-WindowsImage -ImagePath $ViewFile -index $_.ImageIndex };
```

CYCLIC OPERATION AREA, START,

2. Specify the path to mount install.wim

```
New-Item -Path "D:\ISOTEMP_Custom\Install\Install\Mount" -ItemType directory
```

3. Start mounting Install.wim

Default index number: 1

```
Mount-WindowsImage -ImagePath "D:\ISOTEMP\sources\install.wim" -Index "1" -Path "D:\ISOTEMP_Custom\Install\Install\Mount"
```

- Verify Mount Status

  After mounting, check the "Mounted Windows Image Information," including mount directory, image name, and status, by running the command:

  ```
  Get-WindowsImage -Mounted
  ```

PROCESS FILES INSIDE THE INSTALL.WIM IMAGE, OPTIONALLY, START

3.1. Custom encapsulation: WinRE.wim

WARNING:

- WinRE.wim is a file within the Install.wim image;
- When Install.wim has multiple index numbers, only process any WinRE.wim;
- Synchronizing to all index numbers reduces the Install.wim volume, Learn "How to extract and update WinRE.wim in Install.wim".

3.1.1. View WinRE.wim details

Image name, image description, image size, architecture, version, index number, etc.;

```
$ViewFile = "D:\ISOTEMP_Custom\Install\Install\Mount\Windows\System32\Recovery\WinRE.wim"; Get-WindowsImage -ImagePath $ViewFile | Foreach-Object { Get-WindowsImage -ImagePath $ViewFile -index $_.ImageIndex };
```

3.1.2. Specify the path to mount WinRE.wim

```
New-Item -Path "D:\ISOTEMP_Custom\Install\WinRE\Mount" -ItemType directory
```

3.1.3. Start mounting WinRE.wim

Default index number: 1

```
$FileName = "D:\ISOTEMP_Custom\Install\Install\Mount\Windows\System32\Recovery\WinRE.wim"; Mount-WindowsImage -ImagePath $FileName -Index "1" -Path "D:\ISOTEMP_Custom\Install\WinRE\Mount"
```

- Verify Mount Status

  After mounting, check the "Mounted Windows Image Information," including mount directory, image name, and status, by running the command:

  ```
  Get-WindowsImage -Mounted
  ```

3.1.4. Language pack

- Automatically install language packs: Get "Component: All installed packages in the image" and match them. After matching the corresponding names, install the local corresponding language pack files.
- When adding languages, different schema versions must be corresponded, and if not, errors are reported during the addition process.

3.1.4.1. Language pack: add

- WinRE.Instl.lang.ps1
  - \Expand\Install\WinRE\WinRE.Instl.lang.ps1
  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Install/WinRE/WinRE.Instl.lang.ps1

- Copy the code

```
$Mount = "D:\ISOTEMP_Custom\Install\WinRE\Mount"; $Sources = "D:\ISOTEMP_Custom\Install\WinRE\Language\Add\zh-CN"; $Initl_install_Language_Component = @();

Get-WindowsPackage -Path $Mount | ForEach-Object { $Initl_install_Language_Component += $_.PackageName; }

Add-WindowsPackage -Path $Mount -PackagePath "$($Sources)\WinPE-FontSupport-zh-CN.cab"

$Language_List = @(@{ Match = "*WinPE-LanguagePack-Package*"; File = "lp.cab"; }; @{ Match = "*SecureStartup*"; File = "winpe-securestartup_zh-CN.cab"; }; @{ Match = "*ATBroker*"; File = "winpe-atbroker_zh-CN.cab"; }; @{ Match = "*AudioCore*"; File = "winpe-audiocore_zh-CN.cab"; }; @{ Match = "*AudioDrivers*"; File = "winpe-audiodrivers_zh-CN.cab"; }; @{ Match = "*EnhancedStorage*"; File = "winpe-enhancedstorage_zh-CN.cab"; }; @{ Match = "*Narrator*"; File = "winpe-narrator_zh-CN.cab"; }; @{ Match = "*scripting*"; File = "winpe-scripting_zh-CN.cab"; }; @{ Match = "*Speech-TTS*"; File = "winpe-speech-tts_zh-CN.cab"; }; @{ Match = "*srh*"; File = "winpe-srh_zh-CN.cab"; }; @{ Match = "*srt*"; File = "winpe-srt_zh-CN.cab"; }; @{ Match = "*wds-tools*"; File = "winpe-wds-tools_zh-CN.cab"; }; @{ Match = "*-WMI-Package*"; File = "winpe-wmi_zh-CN.cab"; }; @{ Match = "*WinPE-AppxPackaging*"; File = "winpe-appxpackaging_zh-CN.cab"; }; @{ Match = "*StorageWMI*"; File = "winpe-storagewmi_zh-CN.cab"; }; @{ Match = "*WiFi*"; File = "winpe-wifi_zh-CN.cab"; }; @{ Match = "*rejuv*"; File = "winpe-rejuv_zh-CN.cab"; }; @{ Match = "*opcservices*"; File = "winpe-opcservices_zh-CN.cab"; }; @{ Match = "*hta*"; File = "winpe-hta_zh-CN.cab"; })

ForEach ($Rule in $Language_List) {

  Write-host "`n   Rule name: $($Rule.Match)" -ForegroundColor Yellow; Write-host "   $('-' * 80)"

  ForEach ($Component in $Initl_install_Language_Component) {

    if ($Component -like "*$($Rule.Match)*") {

      Write-host "   Component name: " -NoNewline; Write-host $Component -ForegroundColor Green; Write-host "   Language file: " -NoNewline; Write-host
```

```
"$($Sources)\$($Rule.File)" -ForegroundColor Green; Write-Host "   Installing ".PadRight(22) -NoNewline;

      try {

        Add-WindowsPackage -Path $Mount -PackagePath "$($Sources)\$($Rule.File)" | Out-Null

        Write-host "Finish" -ForegroundColor Green

      } catch { Write-host "Failed" -ForegroundColor Red; }

      break

    }

  }

}
```

3.1.4.2. Offline image language: change

3.1.4.2.1. Change default language, regional settings, and other international settings

Language Tag: zh-CN

```
Dism /Image:"D:\ISOTEMP_Custom\Install\WinRE\Mount" /Set-AllIntl:zh-CN
```

3.1.4.2.2. View available language settings

```
Dism /Image:"D:\ISOTEMP_Custom\Install\WinRE\Mount" /Get-Intl
```

3.1.4.3. Components: All packages installed in the image

3.1.4.3.1. View

```
Get-WindowsPackage -Path "D:\ISOTEMP_Custom\Install\WinRE\Mount" | Out-GridView
```

3.1.4.3.2. Export to Csv

```
$SaveTo = "D:\ISOTEMP_Custom\Install\WinRE\Report.Components.$(Get-Date -Format "yyyyMMddHHmmss").csv"; Get-WindowsPackage -Path "D:\ISOTEMP_Custom\Install\WinRE\Mount" | Export-CSV -NoType -Path $SaveTo; Write-host $SaveTo -ForegroundColor Green;
```

3.1.5. Cumulative updates

To prepare the cumulative updates file available, change the example file name: KB_WinRE.cab

3.1.5.1. Add

```
Add-WindowsPackage -Path "D:\ISOTEMP_Custom\Install\WinRE\Mount" -PackagePath "D:\ISOTEMP_Custom\Install\WinRE\Update\KB_WinRE.cab"
```

3.1.5.2. Delete

```
Remove-WindowsPackage -Path "D:\ISOTEMP_Custom\Install\WinRE\Mount" -PackagePath "D:\ISOTEMP_Custom\Install\WinRE\Update\KB_WinRE.cab"
```

3.1.5.3. Solid update

It cannot be uninstalled after curing, which cleans the recovery image and resets the basis of any superseded components.

```
Dism /image:"D:\ISOTEMP_Custom\Install\WinRE\Mount" /cleanup-image /StartComponentCleanup /ResetBase
```

3.1.5.3.1. Clean components after curing and updating

```
$Mount = "D:\ISOTEMP_Custom\Install\Install\Mount"; Get-WindowsPackage -Path $Mount -ErrorAction SilentlyContinue | ForEach-Object { if ($_.PackageState -eq "Superseded") { Write-Host "   $($_.PackageName)" -ForegroundColor Green; Remove-WindowsPackage -Path $Mount -PackageName $_.PackageName | Out-Null; }}
```

3.1.6. Drive

3.1.7. Save image: WinRE.wim

```
Save-WindowsImage -Path "D:\ISOTEMP_Custom\Install\WinRE\Mount"
```

3.1.8. Unmount image: WinRE.wim

Close any applications that may be accessing files in the image, including File Explorer.

```
Dismount-WindowsImage -Path "D:\ISOTEMP_Custom\Install\WinRE\Mount" -Discard
```

3.1.9. After rebuilding WinRE.wim, the file size can be reduced

- WinRE.Rebuild.ps1
  - \Expand\Install\WinRE\WinRE.Rebuild.ps1
  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Install/WinRE/WinRE.Rebuild.ps1

- Copy the code

```
$FileName = "D:\ISOTEMP_Custom\Install\Install\Mount\Windows\System32\Recovery\WinRE.wim"; Get-WindowsImage -ImagePath $Filename -ErrorAction SilentlyContinue | ForEach-Object { Write-Host "   Image name: " -NoNewline; Write-Host "$($_.ImageName)" -ForegroundColor Yellow; Write-Host "   The index number: " -NoNewline; Write-Host "$($_.ImageIndex)" -ForegroundColor Yellow; Write-Host "`n   Under reconstruction ".PadRight(28) -NoNewline; try { Export-WindowsImage -SourceImagePath "$($Filename)" -SourceIndex "$($_.ImageIndex)" -DestinationImagePath "$($FileName).New" -CompressionType max | Out-Null; Write-Host "Finish" -ForegroundColor Green; } catch { Write-Host $_ -ForegroundColor Yellow; Write-host $Failed -ForegroundColor Red; }}; Write-Host "`n   Rename: " -NoNewline -ForegroundColor Yellow; if (Test-Path "$($FileName).New" -PathType Leaf) { Remove-Item -Path $Filename; Move-Item -Path "$($FileName).New" -Destination $Filename; Write-Host "Finish" -ForegroundColor Green; } else { Write-host "Failed" -ForegroundColor Red; }
```

3.1.10. Backup WinRE.wim

- WinRE.Backup.ps1
  - \Expand\Install\WinRE\WinRE.Backup.ps1
  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Install/WinRE/WinRE.Backup.ps1

- Copy the code

```
$WimLibPath = "D:\ISOTEMP_Custom\Install\Install\Update\Winlib"; $FileName = "D:\ISOTEMP_Custom\Install\Install\Mount\Windows\System32\Recovery\WinRE.wim"; New-Item -Path $WimLibPath -ItemType Directory; Copy-Item -Path $FileName -Destination $WimLibPath -Force;
```

3.1.11. Replace WinRE.wim within the Install.wim image

- After each installation of Install.wim, use item "Replace the WinRE.wim";
- Learning "Get all index numbers of Install.wim and replace the old WinRE.wim".

PROCESS FILES INSIDE THE INSTALL.WIM IMAGE, END

4. Language pack

- Automatically install language packs: Get "Component: All installed packages in the image" and match them. After matching the corresponding names, install the local corresponding language pack files.
- When adding languages, different schema versions must be corresponded, and if not, errors are reported during the addition process.

4.1. Language pack: add

- Install.Instl.lang.ps1
  - \Expand\Install\Install.Instl.lang.ps1
  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Install/Install.Instl.lang.ps1

- Copy the code

```
Function Language_Install {
  param($Mount, $Sources, $Lang)
  $Initl_install_Language_Component = @(); $Script:Init_Folder_All_File_Match_Done = @(); $Script:Init_Folder_All_File_Exclude = @();
  if (Test-Path $Mount -PathType Container) { Get-WindowsPackage -Path $Mount | ForEach-Object { $Initl_install_Language_Component += $_.PackageName }; } else { Write-Host "Not mounted: $($Mount)"; return; }
  $Script:Init_Folder_All_File = @()
  if (Test-Path "$($Sources)\$($Lang)" -PathType Container) {
        Get-ChildItem -Path $Sources -Recurse -Include "*.cab" -ErrorAction SilentlyContinue | ForEach-Object { $Script:Init_Folder_All_File += $_.FullName; }

        Write-Host "`n   Available language pack installation files";

        if ($Script:Init_Folder_All_File.Count -gt 0) { ForEach ($item in $Script:Init_Folder_All_File) { Write-Host "   $($item)"; } } else { Write-Host "There are no language pack files locally"; return; }

    } else { Write-Host "Path does not exist: $($Sources)\$($Lang)"; return; }

    $Script:Search_File_Order = @(

        @{ Name = "Fonts"; Description = "Fonts"; Rule = @( "*Fonts*"; ) }

        @{ Name = "Basic"; Description = "Basic"; Rule = @( "*LanguageFeatures-Basic*"; "*Client*Language*Pack*"; ) }

        @{ Name = "OCR"; Description = "Optical character recognition"; Rule = @( "*LanguageFeatures-OCR*"; ) }

        @{ Name = "Handwriting"; Description = "Handwriting recognition"; Rule = @( "*LanguageFeatures-Handwriting*"; ) }

        @{ Name = "TextToSpeech"; Description = "Text-to-speech"; Rule = @( "*LanguageFeatures-TextToSpeech*"; ); }

        @{ Name = "Speech";; Description = "Speech recognition";; Rule = @( "*LanguageFeatures-Speech*"; ) }

        @{ Name = "RegionSpecific"; Description = "Other region-specific requirements"; Rule = @( "*InternationalFeatures*"; ) }

        @{ Name = "Retail"; Description = "Retail demo experience"; Rule = @( "*RetailDemo*"; ) }

        @{ Name = "Features_On_Demand"; Description = "Features on demand"; Rule = @( "*InternetExplorer*~x86~*"; "*InternetExplorer*~x64~*"; "*InternetExplorer*~amd64~*"; "*InternetExplorer*~wow64~*"; "*InternetExplorer*~arm64.x86~*"; "*InternetExplorer*~arm64~*"; "*MSPaint*~x86~*"; "*MSPaint*~x64~*"; "*MSPaint*~amd64~*"; "*MSPaint*~wow64~*"; "*MSPaint*~arm64.x86~*"; "*MSPaint*~arm64~*"; "*Notepad-FoD-Package*~x86~*"; "*Notepad-FoD-Package*~x64~*"; "*Notepad-FoD-Package*~amd64~*"; "*Notepad-FoD-Package*~wow64~*"; "*Notepad-FoD-Package*~arm64.x86~*"; "*Notepad-FoD-Package*~arm64~*"; "*Notepad-System-FoD-Package*~x86~*"; "*Notepad-System-FoD-Package*~x64~*"; "*Notepad-System-FoD-Package*~amd64~*"; "*Notepad-System-FoD-Package*~wow64~*"; "*Notepad-System-FoD-Package*~arm64.x86~*"; "*Notepad-System-FoD-Package*~arm64~*"; "*MediaPlayer*~x86~*"; "*MediaPlayer*~x64~*"; "*MediaPlayer*~amd64~*"; "*MediaPlayer*~wow64~*"; "*MediaPlayer*~arm64.x86~*"; "*MediaPlayer*~arm64~*"; "*PowerShell*ISE*~x86~*"; "*PowerShell*ISE*~x64~*"; "*PowerShell*ISE*~amd64~*"; "*PowerShell*ISE*~wow64~*"; "*PowerShell*ISE*~arm64.x86~*"; "*PowerShell*ISE*~arm64~*"; "*StepsRecorder*~x86~*"; "*StepsRecorder*~x64~*"; "*StepsRecorder*~amd64~*"; "*StepsRecorder*~wow64~*"; "*StepsRecorder*~arm64.x86~*"; "*StepsRecorder*~arm64~*"; "*SnippingTool*~x86~*"; "*SnippingTool*~x64~*"; "*SnippingTool*~amd64~*"; "*SnippingTool*~wow64~*"; "*SnippingTool*~arm64.x86~*"; "*SnippingTool*~arm64~*"; "*WMIC*~x86~*"; "*WMIC*~x64~*"; "*WMIC*~amd64~*"; "*WMIC*~wow64~*"; "*WMIC*~arm64.x86~*"; "*WMIC*~arm64~*"; "*WordPad*~x86~*"; "*WordPad*~x64~*"; "*WordPad*~amd64~*"; "*WordPad*~wow64~*"; "*WordPad*~arm64.x86~*"; "*WordPad*~arm64~*"; "*Printing-WFS*~x86~*"; "*Printing-WFS*~x64~*"; "*Printing-WFS*~amd64~*"; "*Printing-WFS*~wow64~*"; "*Printing-WFS*~arm64.x86~*"; "*Printing-WFS*~arm64~*"; "*Printing-PMCPPC*~x86~*"; "*Printing-PMCPPC*~x64~*"; "*Printing-PMCPPC*~amd64~*"; "*Printing-PMCPPC*~wow64~*"; "*Printing-PMCPPC*~arm64.x86~*"; "*Printing-PMCPPC*~arm64~*"; "*Telnet-Client*~x86~*"; "*Telnet-Client*~x64~*"; "*Telnet-Client*~amd64~*"; "*Telnet-Client*~wow64~*"; "*Telnet-Client*~arm64.x86~*"; "*Telnet-Client*~arm64~*"; "*TFTP-Client*~x86~*"; "*TFTP-Client*~x64~*"; "*TFTP-Client*~amd64~*"; "*TFTP-Client*~wow64~*"; "*TFTP-Client*~arm64.x86~*"; "*TFTP-Client*~arm64~*"; "*VBSCRIPT*~x86~*"; "*VBSCRIPT*~x64~*"; "*VBSCRIPT*~amd64~*"; "*VBSCRIPT*~wow64~*"; "*VBSCRIPT*~arm64.x86~*"; "*VBSCRIPT*~arm64~*"; "*WinOcr-FOD-Package*~x86~*"; "*WinOcr-FOD-Package*~x64~*"; "*WinOcr-FOD-Package*~amd64~*"; "*WinOcr-FOD-Package*~wow64~*"; "*WinOcr-FOD-Package*~arm64.x86~*"; "*WinOcr-FOD-Package*~arm64~*"; "*ProjFS-OptionalFeature-FOD-Package*~x86~*"; "*ProjFS-OptionalFeature-FOD-Package*~x64~*"; "*ProjFS-OptionalFeature-FOD-Package*~amd64~*"; "*ProjFS-OptionalFeature-FOD-Package*~wow64~*"; "*ProjFS-OptionalFeature-FOD-Package*~arm64.x86~*"; "*ProjFS-OptionalFeature-FOD-Package*~arm64~*"; "*ServerCoreFonts-NonCritical-Fonts-BitmapFonts-FOD-Package*~x86~*"; "*ServerCoreFonts-NonCritical-Fonts-BitmapFonts-FOD-Package*~x64~*"; "*ServerCoreFonts-NonCritical-Fonts-BitmapFonts-FOD-Package*~amd64~*"; "*ServerCoreFonts-NonCritical-Fonts-BitmapFonts-FOD-Package*~wow64~*"; "*ServerCoreFonts-NonCritical-Fonts-BitmapFonts-FOD-Package*~arm64.x86~*"; "*ServerCoreFonts-NonCritical-Fonts-BitmapFonts-FOD-Package*~arm64~*"; "*ServerCoreFonts-NonCritical-Fonts-Support-FOD-Package*~x86~*"; "*ServerCoreFonts-NonCritical-Fonts-Support-FOD-Package*~x64~*"; "*ServerCoreFonts-NonCritical-Fonts-Support-FOD-Package*~amd64~*"; "*ServerCoreFonts-NonCritical-Fonts-Support-FOD-Package*~wow64~*"; "*ServerCoreFonts-NonCritical-Fonts-Support-FOD-Package*~arm64.x86~*"; "*ServerCoreFonts-NonCritical-Fonts-Support-FOD-Package*~arm64~*"; "*ServerCoreFonts-NonCritical-Fonts-TrueType-FOD-Package*~x86~*"; "*ServerCoreFonts-NonCritical-Fonts-TrueType-FOD-Package*~x64~*"; "*ServerCoreFonts-NonCritical-Fonts-TrueType-FOD-Package*~amd64~*"; "*ServerCoreFonts-NonCritical-Fonts-TrueType-FOD-Package*~wow64~*"; "*ServerCoreFonts-NonCritical-Fonts-TrueType-FOD-Package*~arm64.x86~*"; "*ServerCoreFonts-NonCritical-Fonts-TrueType-FOD-Package*~arm64~*"; "*SimpleTCP-FOD-Package*~x86~*"; "*SimpleTCP-FOD-Package*~x64~*"; "*SimpleTCP-FOD-Package*~amd64~*"; "*SimpleTCP-FOD-Package*~wow64~*"; "*SimpleTCP-FOD-Package*~arm64.x86~*"; "*SimpleTCP-FOD-Package*~arm64~*"; "*VirtualMachinePlatform-Client-Disabled-FOD-Package*~x86~*"; "*VirtualMachinePlatform-Client-Disabled-FOD-Package*~x64~*"; "*VirtualMachinePlatform-Client-Disabled-FOD-Package*~amd64~*";
"*VirtualMachinePlatform-Client-Disabled-FOD-Package*~wow64~*"; "*VirtualMachinePlatform-Client-Disabled-FOD-Package*~arm64.x86~*"; "*VirtualMachinePlatform-Client-Disabled-FOD-Package*~arm64~*"; "*DirectoryServices-ADAM-Client-FOD-Package*~x86~*"; "*DirectoryServices-ADAM-Client-FOD-Package*~x64~*"; "*DirectoryServices-ADAM-Client-FOD-Package*~amd64~*"; "*DirectoryServices-ADAM-Client-FOD-Package*~wow64~*"; "*DirectoryServices-ADAM-Client-FOD-Package*~arm64.x86~*"; "*DirectoryServices-ADAM-Client-FOD-Package*~arm64~*"; "*EnterpriseClientSync-Host-FOD-Package*~x86~*"; "*EnterpriseClientSync-Host-FOD-Package*~x64~*"; "*EnterpriseClientSync-Host-FOD-Package*~amd64~*"; "*EnterpriseClientSync-Host-FOD-Package*~wow64~*"; "*EnterpriseClientSync-Host-FOD-Package*~arm64.x86~*"; "*EnterpriseClientSync-Host-FOD-Package*~arm64~*"; "*SenseClient-FoD-Package*~x86~*"; "*SenseClient-FoD-Package*~x64~*"; "*SenseClient-FoD-Package*~amd64~*"; "*SenseClient-FoD-Package*~wow64~*"; "*SenseClient-FoD-Package*~arm64.x86~*"; "*SenseClient-FoD-Package*~arm64~*"; "*SmbDirect-FOD-Package*~x86~*"; "*SmbDirect-FOD-Package*~x64~*"; "*SmbDirect-FOD-Package*~amd64~*"; "*SmbDirect-FOD-Package*~wow64~*"; "*SmbDirect-FOD-Package*~arm64.x86~*"; "*SmbDirect-FOD-Package*~arm64~*"; "*TerminalServices-AppServer-Client-FOD-Package*~x86~*"; "*TerminalServices-AppServer-Client-FOD-Package*~x64~*"; "*TerminalServices-AppServer-Client-FOD-Package*~amd64~*"; "*TerminalServices-AppServer-Client-FOD-Package*~wow64~*"; "*TerminalServices-AppServer-Client-FOD-Package*~arm64.x86~*"; "*TerminalServices-AppServer-Client-FOD-Package*~arm64~*"; )}

    )

    ForEach ($item in $Script:Search_File_Order) { New-Variable -Name "Init_File_Type_$($item.Name)" -Value @() -Force }

    $ExcludeNameItem = @( "Fonts"; "RegionSpecific"; )

    ForEach ($WildCard in $Script:Init_Folder_All_File) {

        ForEach ($item in $Script:Search_File_Order) {

            ForEach ($NewRule in $item.Rule) {

                if ($WildCard -like "*$($NewRule)*") {

                    Write-Host "`n   Fuzzy matching: " -NoNewline; Write-Host $NewRule -ForegroundColor Green; Write-Host "   File name: " -NoNewline; Write-Host $WildCard -ForegroundColor Green;

                    $OSDefaultUser = (Get-Variable -Name "Init_File_Type_$($item.Name)" -ErrorAction SilentlyContinue).Value

                    $TempSave = @{ Match_Name = $NewRule; FileName = $WildCard }

                    $new = $OSDefaultUser + $TempSave

                    if ($item.name -Contains $ExcludeNameItem) { Write-Host "   Do not match, install directly" -ForegroundColor Yellow; New-Variable -Name "Init_File_Type_$($item.Name)" -Value $new -Force; $Script:Init_Folder_All_File_Match_Done += $WildCard; } else {

                        ForEach ($Component in $Initl_install_Language_Component) { if ($Component -like "*$($NewRule)*") { Write-Host "   Component name: " -NoNewline; Write-Host $Component -ForegroundColor Green; New-Variable -Name "Init_File_Type_$($item.Name)" -Value $new -Force; $Script:Init_Folder_All_File_Match_Done += $WildCard; break; }}

                    }

                }

            }

        }

    }

    Write-Host "`n   Grouping is complete, pending installation" -ForegroundColor Yellow; Write-Host "   $('-' * 80)";

    ForEach ($WildCard in $Script:Search_File_Order) {

        $OSDefaultUser = (Get-Variable -Name "Init_File_Type_$($WildCard.Name)" -ErrorAction SilentlyContinue).Value

        Write-Host "`n   $($WildCard.Description) ( $($OSDefaultUser.Count) item )"

        if ($OSDefaultUser.Count -gt 0) {  ForEach ($item in $OSDefaultUser) { Write-Host "   $($item.FileName)" -ForegroundColor Green; } } else { Write-Host "   Not available" -
```

```
ForegroundColor Red; }
    }
    Write-Host "`n   Not matched, no longer installed" -ForegroundColor Yellow; Write-Host "   $('-' * 80)";
    ForEach ($item in $Script:Init_Folder_All_File) { if ($Script:Init_Folder_All_File_Match_Done -notcontains $item) { $Script:Init_Folder_All_File_Exclude += $item; Write-Host "   $($item)" -ForegroundColor Red; }}
    Write-Host "`n   Install" -ForegroundColor Yellow; Write-Host "   $('-' * 80)";
    ForEach ($WildCard in $Script:Search_File_Order) {
        $OSDefaultUser = (Get-Variable -Name "Init_File_Type_$($WildCard.Name)" -ErrorAction SilentlyContinue).Value
        Write-Host "`n   $($WildCard.Description) ( $($OSDefaultUser.Count) item )"; Write-Host "   $('-' * 80)";
        if ($OSDefaultUser.Count -gt 0) {
            ForEach ($item in $OSDefaultUser) {
                Write-Host "   File name: " -NoNewline; Write-Host $item.FileName -ForegroundColor Green; Write-Host "   Installing ".PadRight(22) -NoNewline;
                if (Test-Path $item.FileName -PathType Leaf) {
                    try {
                        Add-WindowsPackage -Path $Mount -PackagePath $item.FileName | Out-Null
                        Write-Host "Finish`n" -ForegroundColor Green
                    } catch { Write-Host "Failed" -ForegroundColor Red; Write-Host "   $($_)" -ForegroundColor Red; }
                } else { Write-Host "Does not exist`n"; }
            }
        } else { Write-Host "   Not available`n" -ForegroundColor Red; }
    }
}
Language_Install -Mount "D:\ISOTemp_Custom\Install\Install\Mount" -Sources "D:\ISOTemp_Custom\Install\Install\Language\Add" -Lang "zh-CN"
```

4.2. Offline image language: change

- Starting Windows 11, the default System UI Language set by DISM is left unaltered on all editions except for Home edition. For all commercial editions the language chosen during the Out-of-Box Experience (OOBE) is set as the System Preferred UI language and Windows will be displayed in this language and for Home edition the language chosen at OOBE will continue to be the default System UI Language.
- As of Windows 10, version 2004, if an .appx-based Language Experience Pack (LXP) backed language is passed as an argument then the language will be set as the System Preferred UI language and its parent language will be set as the Default System UI language. In prior versions only .cab based language packs were supported.

4.2.1. Change default language, regional settings, and other international settings

Language Tag: zh-CN

```
Dism /Image:"D:\ISOTEMP_Custom\Install\Install\Mount" /Set-AllIntl:zh-CN
```

4.2.2. View available language settings

```
Dism /Image:"D:\ISOTEMP_Custom\Install\Install\Mount" /Get-Intl
```

4.3. Components: All packages installed in the image

4.3.1. View

```
Get-WindowsPackage -Path "D:\ISOTEMP_Custom\Install\Install\Mount" | Out-GridView
```

4.3.2. Export to Csv

```
$SaveTo = "D:\ISOTEMP_Custom\Install\Install\Report.Components.$(Get-Date -Format "yyyyMMddHHmmss").csv"; Get-WindowsPackage -Path "D:\ISOTEMP_Custom\Install\Install\Mount" | Export-CSV -NoType -Path $SaveTo; Write-host $SaveTo -ForegroundColor Green;
```

5. Cumulative updates

5.1. Download

Check the "Windows Server 2022 Update History ", for example, install the cumulative update: KB5030216

Go to the download page: https://www.catalog.update.microsoft.com/Search.aspx?q=Kb5030216 Or "Direct download" (If you cannot download, please go to the download page), save to

D:\ISOTEMP_Custom\Install\Install\Update\windows10.0-kb5030216-x64_cbe587155f9818548b75f65d5cd41d341ed2fc61.msu

5.2. Add

```
Add-WindowsPackage -Path "D:\ISOTEMP_Custom\Install\Install\Mount" -PackagePath "D:\ISOTEMP_Custom\Install\Install\Update\windows10.0-kb5030216-x64_cbe587155f9818548b75f65d5cd41d341ed2fc61.msu"
```

5.3. Solid update

It cannot be uninstalled after curing, which cleans the recovery image and resets the basis of any superseded components.

```
Dism /Image:"D:\ISOTEMP_Custom\Install\Install\Mount" /cleanup-image /StartComponentCleanup /ResetBase
```

5.3.1. Clean up components after curing updates

- Install.Update.Curing.ps1
  - \Expand\Install\Install.Update.Curing.ps1
  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Install/Install.Update.Curing.ps1

- Copy the code

```
$Mount = "D:\ISOTEMP_Custom\Install\Install\Mount"; Get-WindowsPackage -Path $Mount -ErrorAction SilentlyContinue | ForEach-Object { if ($_.PackageState -eq "Superseded") { Write-Host "  $($_.PackageName)" -ForegroundColor Green; Remove-WindowsPackage -Path $Mount -PackageName $_.PackageName | Out-Null; }}
```

6. Drive

7. Deployment engine: Add

   - Learn "Deployment engine", if added to ISO installation media, can skip adding to mounted.
   - After adding the deployment engine, continue at the current location.

8. Health

   Check whether there is any damage before saving. When the health status is abnormal, abort saving

   ```
   Repair-WindowsImage -Path "D:\ISOTEMP_Custom\Install\Install\Mount" -ScanHealth
   ```

9. Replace the WinRE.wim

   WinRE.wim in all index numbers in Install.wim has been replaced in batches. Please skip this step.

   ```
   $WinRE = "D:\ISOTEMP_Custom\Install\Install\Update\Winlib\WinRE.wim"; $CopyTo = "D:\ISOTEMP_Custom\Install\Install\Mount\Windows\System32\Recovery"; Copy-Item -Path $WinRE -Destination $CopyTo -Force;
   ```

10. Save image: Install.wim

    ```
    Save-WindowsImage -Path "D:\ISOTEMP_Custom\Install\Install\Mount"
    ```

11. Unmount image: Install.wim

    Close any applications that may be accessing files in the image, including File Explorer.

    ```
    Dismount-WindowsImage -Path "D:\ISOTEMP_Custom\Install\Install\Mount" -Discard
    ```

CYCLIC OPERATION AREA, END.

12. How to batch replace WinRE.wim in all index numbers in Install.wim

    12.1. Get WimLib

    After going to the official website of https://wimlib.net, select a different version: arm64, x64, x86, and extract it to: D:Wimlib after downloading.

    12.2. How to extract and update WinRE.wim in Install.wim

    12.2.1. Extract the WinRE.wim file from Install.wim

    - Install.WinRE.Extract.ps1

- \Expand\Install\Install.WinRE.Extract.ps1
- https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Install/Install.WinRE.Extract.ps1

- Copy the code

```
$Arguments = @( "extract", "D:\ISOTEMP\sources\install.wim", "1", "\Windows\System32\Recovery\Winre.wim", "--dest-dir=""D:\ISOTEMP_Custom\Install\Install\Update\Winlib""" ); New-Item -Path "D:\ISOTEMP_Custom\Install\Install\Update\Winlib" -ItemType Directory; Start-Process -FilePath "d:\wimlib\wimlib-imagex.exe" -ArgumentList $Arguments -wait -nonewwindow;
```

12.2.2. Get all index numbers of Install.wim and replace the old WinRE.wim

- Install.WinRE.Replace.wim.ps1
  - \Expand\Install\Install.WinRE.Replace.wim.ps1
  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Install/Install.WinRE.Replace.wim.ps1

- Copy the code

```
Get-WindowsImage -ImagePath "D:\ISOTEMP\sources\install.wim" -ErrorAction SilentlyContinue | ForEach-Object { Write-Host "   Image name: " -NoNewline; Write-Host $_.ImageName -ForegroundColor Yellow; Write-Host "   The index number: " -NoNewline; Write-Host $_.ImageIndex -ForegroundColor Yellow; Write-Host "`n   Replacement "; $Arguments = @( "update", "D:\ISOTEMP\sources\install.wim", $_.ImageIndex, "--command=""add 'D:\ISOTEMP_Custom\Install\Install\Update\Winlib\WinRE.wim' '\Windows\System32\Recovery\WinRe.wim'""" ); Start-Process -FilePath "d:\wimlib\wimlib-imagex.exe" -ArgumentList $Arguments -wait -nonewwindow; Write-Host "   Finish`n" -ForegroundColor Green; }
```

13. Rebuilding Install.wim reduces file size

- Install.Rebuild.wim.ps1
  - \Expand\Install\Install.Rebuild.wim.ps1
  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Install/Install.Rebuild.wim.ps1

- Copy the code

```
$InstallWim = "D:\ISOTEMP\sources\install.wim"; Get-WindowsImage -ImagePath $InstallWim -ErrorAction SilentlyContinue | ForEach-Object { Write-Host "   Image name: " -NoNewline; Write-Host $_.ImageName -ForegroundColor Yellow; Write-Host "   The index number: " -NoNewline; Write-Host $_.ImageIndex -ForegroundColor Yellow; Write-Host "`n   Under reconstruction".PadRight(28) -NoNewline; Export-WindowsImage -SourceImagePath $InstallWim -SourceIndex $_.ImageIndex -DestinationImagePath "$($InstallWim).New" -CompressionType max | Out-Null; Write-Host "Finish`n" -ForegroundColor Green; }; if (Test-Path "$($InstallWim).New" -PathType Leaf) { Remove-Item -Path $InstallWim; Move-Item -Path "$($InstallWim).New" -Destination $InstallWim; Write-Host "Finish" -ForegroundColor Green; } else { Write-host "Failed" -ForegroundColor Red; }
```

14. Split, merge, compress, and convert

Solid compression is in ESD file format. If the file exceeds 4GB, it cannot be split and cannot be copied to a FAT32 disk. This is a disadvantage.

Using FAT32 format to store Windows installation boot is the best solution. If the Install.wim file exceeds 4GB and cannot be copied to a FAT32 disk, you need to split the Install.wim file and copy it to a FAT32 disk after the file size is less than 4GB.

It is particularly important to learn how to split and merge, solid compression and conversion.

14.1. Splitting and merging

14.1.1. Splitting

After splitting Install.wim into 4GB file sizes and getting new file names Install.*.swm, delete the old Install.wim.

- Install.Split.ps1
  - \Expand\Install\Install.Split.ps1
  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Install/Install.Split.ps1

- Copy the code

```
Write-host "Split Install.wim into Install.*.swm"; Write-host "Splitting" -NoNewline; Split-WindowsImage -
ImagePath "D:\ISOTEMP\sources\install.wim" -SplitImagePath "D:\ISOTEMP\sources\install.swm" -FileSize "4096"
-CheckIntegrity -ErrorAction SilentlyContinue | Out-Null; Write-Host "Split Complete`n" -ForegroundColor Green;
Write-host "`nVerify completion and delete old files"; if (Test-Path -Path "D:\ISOTEMP\sources\install.swm" -
PathType leaf) { Remove-Item -Path "D:\ISOTEMP\sources\install.wim"; Write-Host "Done" -ForegroundColor
Green; } else { Write-Host "Failed" -ForegroundColor Red; }
```

14.1.2. Merge

After merging all Install.*.swm into Install.wim, delete the old Install.*.swm.

- Install.Merging.ps1
  - \Expand\Install\Install.Merging.ps1
  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Install/Install.Merging.ps1

- Copy the code

```
Write-host "Merge all Install.*.swm files into Install.wim"; Get-WindowsImage -ImagePath
"D:\ISOTEMP\Sources\install.swm" -ErrorAction SilentlyContinue | ForEach-Object { Write-Host "Image Name: " -
NoNewline; Write-Host $_.ImageName -ForegroundColor Yellow; Write-Host "Index Number: " -NoNewline;
    Write-Host $_.ImageIndex -ForegroundColor Yellow; Write-Host "Exporting".PadRight(28) -NoNewline; dism
/export-image /SourceImageFile:"D:\ISOTEMP\Sources\install.swm" /swmfile:"D:\ISOTEMP\sources\install*.swm"
/SourceIndex:"$($_.ImageIndex)" /DestinationImageFile:"D:\ISOTEMP\Sources\install.wim" /Compress:"Max"
/CheckIntegrity; Write-Host "Export Complete`n" -ForegroundColor Green; }; Write-host "`nVerify completion and
delete old files"; if (Test-Path -Path "D:\ISOTEMP\Sources\install.wim" -PathType leaf) { Get-ChildItem -Path
"D:\ISOTEMP\sources" -Recurse -include "*.swm" | ForEach-Object { Write-Host "Delete: $($_.Fullname)" -
ForegroundColor Green; Remove-Item -Path $_.Fullname; }; Write-Host "Done" -ForegroundColor Green; } else
{ Write-Host "Falied" -ForegroundColor Green; }
```

14.2. Solid compressed ESD format and WIM format conversion

14.2.1. Solid compression

After solid compression, you can edit version information and application files, etc.; you cannot mount images, etc. After obtaining the new file install.esd, delete the old Install.wim.

- Install.Compress.ps1
    - \Expand\Install\Install.Compress.ps1
    - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Install/Install.Compress.ps1
- Copy the code

```
Write-host "Solid compressed Install.wim"; Get-WindowsImage -ImagePath "D:\ISOTEMP\Sources\install.wim" -ErrorAction SilentlyContinue | ForEach-Object { Write-Host "Image Name: " -NoNewline; Write-Host $_.ImageName -ForegroundColor Yellow; Write-Host "Index Number: " -NoNewline; Write-Host $_.ImageIndex -ForegroundColor Yellow; Write-Host "Compressing".PadRight(28) -NoNewline; dism /export-image /SourceImageFile:"D:\ISOTEMP\Sources\install.wim" /SourceIndex:"$($_.ImageIndex)" /DestinationImageFile:"D:\ISOTEMP\Sources\install.esd" /Compress:recovery /CheckIntegrity; Write-Host "Compression completed`n" -ForegroundColor Green; }; Write-host "`nVerify completion and delete old files"; if (Test-Path -Path "D:\ISOTEMP\Sources\install.esd" -PathType leaf) { Remove-Item -Path "D:\ISOTEMP\Sources\install.wim"; Write-Host "Done" -ForegroundColor Green; } else { Write-Host "Falied" -ForegroundColor Green; }
```

14.2.2. Convert compressed files to WIM file format

- Install.Convert.ps1
    - \Expand\Install\Install.Convert.ps1
    - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Install/Install.Convert.ps1
- Copy the code

```
Write-host "Convert ESD to WIM"; Get-WindowsImage -ImagePath "D:\ISOTEMP\Sources\install.esd" -ErrorAction SilentlyContinue | ForEach-Object { Write-Host "Image Name: " -NoNewline; Write-Host $_.ImageName -ForegroundColor Yellow; Write-Host "Index Number: " -NoNewline; Write-Host $_.ImageIndex -ForegroundColor Yellow; Write-Host "Exporting".PadRight(28) -NoNewline; try { Export-WindowsImage -SourceImagePath "D:\ISOTEMP\Sources\install.esd" -SourceIndex $_.ImageIndex -DestinationImagePath "D:\ISOTEMP\Sources\install.wim" -CompressionType "Max" -CheckIntegrity -ErrorAction SilentlyContinue | Out-Null; Write-Host "Done`n" -ForegroundColor Green; } catch { Write-Host $_ -ForegroundColor Yellow; Write-host "Falied`n" -ForegroundColor Red; }}; Write-host "`nVerify completion and delete old files"; if (Test-Path -Path "D:\ISOTEMP\Sources\install.wim" -PathType leaf) { Remove-Item -Path "D:\ISOTEMP\Sources\install.esd"; Write-Host "Done" -ForegroundColor Green; } else { Write-Host "Falied" -ForegroundColor Green; }
```

II. Custom encapsulation: boot.wim

1. View Boot.wim details

Image name, image description, image size, architecture, version, index number, etc.;

```
$ViewFile = "D:\ISOTEMP\Sources\Boot.wim"; Get-WindowsImage -ImagePath $ViewFile | Foreach-Object { Get-WindowsImage -ImagePath $ViewFile -index $_.ImageIndex };
```

2. Specify the path to mount Boot.wim

```
New-Item -Path "D:\ISOTEMP_Custom\Boot\Boot\Mount" -ItemType directory
```

3. Start mounting Boot.wim

   Default index number: 2

```
Mount-WindowsImage -ImagePath "D:\ISOTEMP\sources\boot.wim" -Index "2" -Path "D:\ISOTEMP_Custom\Boot\Boot\Mount"
```

- Verify Mount Status

  After mounting, check the "Mounted Windows Image Information," including mount directory, image name, and status, by running the command:

```
Get-WindowsImage -Mounted
```

4. Language pack

- Automatically install language packs: Get "Component: All installed packages in the image" and match them. After matching the corresponding names, install the local corresponding language pack files.
- When adding languages, different schema versions must be corresponded, and if not, errors are reported during the addition process.

4.1. Language pack: add

- Boot.Instl.lang.ps1
  - \Expand\Boot\Boot.Instl.lang.ps1
  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/Boot/Boot.Instl.lang.ps1

- Copy the code

```
$Mount = "D:\ISOTEMP_Custom\Boot\Boot\Mount"; $Sources = "D:\ISOTEMP_Custom\Boot\Boot\Language\Add\zh-CN"; $Initl_install_Language_Component = @();

Get-WindowsPackage -Path $Mount | ForEach-Object { $Initl_install_Language_Component += $_.PackageName; }

Add-WindowsPackage -Path $Mount -PackagePath "$($Sources)\WinPE-FontSupport-zh-CN.cab"

$Language = @( @{ Match = "*WinPE*Setup*Server_zh*Package*"; File = "WINPE-SETUP-Server_zh-CN.CAB"; }; @{ Match = "*WinPE*Setup*Package*"; File = "WinPE-Setup_zh-CN.cab"; }; @{ Match = "*WinPE-LanguagePack-Package*"; File = "lp.cab"; }; @{ Match = "*SecureStartup*"; File = "winpe-securestartup_zh-CN.cab"; }; @{ Match = "*ATBroker*"; File = "winpe-atbroker_zh-CN.cab"; }; @{ Match = "*AudioCore*"; File = "winpe-audiocore_zh-CN.cab"; }; @{ Match = "*AudioDrivers*"; File = "winpe-audiodrivers_zh-CN.cab"; }; @{ Match = "*EnhancedStorage*"; File = "winpe-enhancedstorage_zh-CN.cab"; }; @{ Match = "*Narrator*"; File = "winpe-narrator_zh-CN.cab"; }; @{ Match = "*scripting*"; File = "winpe-scripting_zh-CN.cab"; }; @{ Match = "*Speech-TTS*"; File = "winpe-speech-tts_zh-CN.cab"; }; @{ Match = "*srh*"; File = "winpe-srh_zh-CN.cab"; }; @{ Match = "*srt*"; File = "winpe-srt_zh-CN.cab"; }; @{ Match = "*wds-tools*"; File = "winpe-wds-tools_zh-CN.cab"; }; @{ Match = "*-WMI-Package*"; File = "winpe-wmi_zh-CN.cab"; })

ForEach ($Rule in $Language) {
  Write-host "`n  Rule name: $($Rule.Match)" -ForegroundColor Yellow; Write-host "  $('-' * 80)";

  ForEach ($Component in $Initl_install_Language_Component) {

    if ($Component -like "*$($Rule.Match)*") {

      Write-host "  Component name: " -NoNewline; Write-host $Component -ForegroundColor Green; Write-host "  Language file: " -NoNewline; Write-host "$($Sources)\$($Rule.File)" -ForegroundColor Green; Write-Host "  Installing ".PadRight(22) -NoNewline;

      try {

        Add-WindowsPackage -Path $Mount -PackagePath "$($Sources)\$($Rule.File)" | Out-Null

        Write-host "Finish" -ForegroundColor Green

      } catch { Write-host "Failed" -ForegroundColor Red; }; break; }

  }

}
```

4.2. Offline image language: change

4.2.1. Change default language, regional settings, and other international settings

Language Tag: zh-CN

```
Dism /Image:"D:\ISOTEMP_Custom\Boot\Boot\Mount" /Set-AllIntl:zh-CN
```

4.2.2. View available language settings

```
Dism /Image:"D:\ISOTEMP_Custom\Boot\Boot\Mount" /Get-Intl
```

4.3. Components: All packages installed in the image

4.3.1. View

```
Get-WindowsPackage -Path "D:\ISOTEMP_Custom\Boot\Boot\Mount" | Out-GridView
```

4.3.2. Export to Csv

```
$SaveTo = "D:\ISOTEMP_Custom\Boot\Boot\Report.Components.$(Get-Date -Format "yyyyMMddHHmmss").csv"; Get-WindowsPackage -Path "D:\ISOTEMP_Custom\Boot\Boot\Mount" | Export-CSV -NoType -Path $SaveTo; Write-host $SaveTo -ForegroundColor Green;
```

4.4. Language packs: sync to ISO installer

```
Copy-Item -Path "D:\ISOTEMP_Custom\Boot\Boot\Mount\sources\zh-CN" -Destination "D:\ISOTEMP\sources\zh-CN" -Recurse -Force
```

4.5. Regenerate Lang.ini

After regeneration, you can adjust the "Installation Interface", the order when selecting "Language", open lang.ini, the default preferred value = 3, non-default value = 2.

4.5.1. Regenerate the mounted directory lang.ini

Re-generated Lang.ini file location: D:\ISOTEMP_Custom\Boot\Boot\Mount\Sources\lang.ini

```
Dism /image:"D:\ISOTEMP_Custom\Boot\Boot\Mount" /gen-langini /distribution:"D:\ISOTEMP_Custom\Boot\Boot\Mount"
```

4.5.2. After regenerating lang.ini, synchronize to the installer

Re-generated Lang.ini file location: D:\ISOTEMP\Sources\lang.ini

```
Dism /image:"D:\ISOTEMP_Custom\Boot\Boot\Mount" /gen-langini /distribution:"D:\ISOTEMP"
```

5. Cumulative updates

To prepare the cumulative updates file available, change the example file name: KB_Boot.cab

5.1. Add

```
Add-WindowsPackage -Path "D:\ISOTEMP_Custom\Boot\Boot\Mount" -PackagePath "D:\ISOTEMP_Custom\Boot\Boot\Update\KB_Boot.cab"
```

5.2. Delete

```
Remove-WindowsPackage -Path "D:\ISOTEMP_Custom\Boot\Boot\Mount" -PackagePath "D:\ISOTEMP_Custom\Boot\Boot\Update\KB_Boot.cab"
```

5.3. Solid update

It cannot be uninstalled after curing, which cleans the recovery image and resets the basis of any superseded components.

```
Dism /image:"D:\ISOTEMP_Custom\Boot\Boot\Mount" /cleanup-image /StartComponentCleanup /ResetBase
```

5.3.1. Clean components after curing and updating

```
$Mount = "D:\ISOTEMP_Custom\Boot\Boot\Mount"; Get-WindowsPackage -Path $Mount -ErrorAction SilentlyContinue | ForEach-Object { if ($_.PackageState -eq "Superseded") { Write-Host "   $($_.PackageName)" -ForegroundColor Green; Remove-WindowsPackage -Path $Mount -PackageName $_.PackageName | Out-Null; }}
```

6. Drive

7. Save image: Boot.wim

```
Save-WindowsImage -Path "D:\ISOTEMP_Custom\Boot\Boot\Mount"
```

8. Unmount image: Boot.wim

Close any applications that may be accessing files in the image, including File Explorer.

```
Dismount-WindowsImage -Path "D:\ISOTEMP_Custom\Boot\Boot\Mount" -Discard
```

III. Deployment engine

- Learn about "Automatically Adding Languages Installed in Windows Systems", learn: https://github.com/ilikeyi/Multilingual, how to download:
  - After entering the website, click "Code", "Download Compressed Package", and after the download is completed, you will get the main.zip compressed package file.
  - Go to the https://github.com/ilikeyi/Multilingual/releases download page, select the available version: 1.1.1.1, select the download source code format: zip, and get the Multilingual-1.1.1.1.zip compressed package file after the download is completed;
- Unzip the downloaded main.zip or Multilingual-1.1.1.1.zip to: D:\Multilingual-1.1.1.1, and rename: D:\Multilingual
- Learn "Unattended Windows Setup Reference", Intervene in the installation process by leaving it unattended.

1. Add method

1.1. Add to ISO installation media

1.1.1. Unattended

1.1.1.1. Add to: [ISO]:\Autounattend.xml

Autounattend.xml interferes with the WinPE installer when booting an ISO installation.

Copy D:\Multilingual_Learn\Unattend\Mul.Unattend.xml to D:\ISOTEMP\Autounattend.xml

```
Copy-Item "D:\Multilingual\_Learn\Unattend\Mul.Unattend.xml" -Destination
"D:\ISOTEMP\Autounattend.xml" -Force
```

1.1.1.2. Add to: [ISO]:\Sources\Unattend.xml

When mounting or unpacking an ISO, after running the [ISO]:\Setup.exe installer, [ISO]:\Sources\Unattend.xml will intervene in the installation process.

Copy D:\Multilingual_Learn\Unattend\Mul.Unattend.xml to D:\ISOTEMP\Sources\Unattend.xml

```
Copy-Item "D:\Multilingual\_Learn\Unattend\Mul.Unattend.xml" -Destination
"D:\ISOTEMP\Sources\Unattend.xml" -Force
```

1.1.1.3. Add to: [ISO]:\sources\$OEM$\$$\Panther\unattend.xml

Copy it to the system disk during the installation process, copy to: {system disk}:\Windows\Panther\unattend.xml

1.1.1.3.1. Create $OEM$ path

```
New-Item -Path "D:\ISOTEMP\sources\`$OEM$\`$$\Panther" -ItemType Directory
```

1.1.1.3.2. Copy

Copy D:\Multilingual_Learn\Unattend\Mul.Unattend.xml to D:\ISOTEMP\Sources\OEM\Panther\Unattend.xml

```
Copy-Item "D:\Multilingual\_Learn\Unattend\Mul.Unattend.xml" -Destination
"D:\ISOTEMP\sources\`$OEM$\`$$\Panther\Unattend.xml" -Force
```

1.1.2. Deployment engine: add

Add "Automatically add installed languages for Windows systems" to D:\ISOTEMP\sources\$OEM$\$1\Yi\Engine in the directory.

1.1.2.1. Deployment Engine: Copy

Copy D:\Multilingual\Engine to D:\ISOTEMP\Sources\$OEM$\$1\Yi\Engine

```
Copy-Item "D:\Multilingual\Engine" -Destination "D:\ISOTEMP\sources\`$OEM$\`$1\Yi\Engine" -Recurse -
Force
```

1.1.2.2. Deployment engine: custom deployment tags

```
$Flag = @(
  "Is_Mark_Sync" # Allow global search and synchronization of deployment tags

  # Prerequisite deployment

#   "Auto_Update" # Allow automatic updates

#   "Use_UTF8" # Beta: Global language support using Unicode UTF-8

  "Disable_Network_Location_Wizard" # Network Location Wizard

  "Disable_Cleanup_Appx_Tasks" # Appx Cleanup and maintenance tasks

  "Disable_Cleanup_On_Demand_Language" # Prevent cleanup of unused on-demand feature language
packs

  "Disable_Cleanup_Unsed_Language"   # Prevent cleaning of unused language packs

  "Prerequisites_Reboot" # Restart your computer

  # Complete first deployment

#   "Popup_Engine" # Allow the deployment engine main interface to pop up for the first time

#   "Allow_First_Pre_Experience" # Allow first preview, as planned

  "Reset_Execution_Policy" # Restore PowerShell execution policy: Restricted

  "Clear_Solutions" # Delete the entire solution

  "Clear_Engine" # Delete the deployment engine and keep the others

#   "First_Experience_Reboot" # Restart your computer

)

ForEach ($item in $Flag) {

  Write-host "   $($item)" -ForegroundColor Green
```

```
    New-Item -Path "D:\ISOTEMP\sources\`$OEM$\`$1\Yi\Engine\Deploy\Allow" -ItemType Directory -ErrorAction SilentlyContinue | Out-Null

    Out-File -FilePath "D:\ISOTEMP\sources\`$OEM$\`$1\Yi\Engine\Deploy\Allow\$($item)" -Encoding utf8 -ErrorAction SilentlyContinue

}
```

1.2. Add to mounted

Through "Custom encapsulation: Install.wim", execute "Start mounting Install.wim" and mount to: D:\ISOTEMP_Custom\Install\Install\Mount

1.2.1. Unattended

Copy D:\Multilingual_Learn\Unattend\Mul.Unattend.xml to D:\ISOTEMP_Custom\Install\Install\Mount\Panther\Unattend.xml

```
Copy-Item "D:\Multilingual\_Learn\Unattend\Mul.Unattend.xml" -Destination "D:\ISOTEMP_Custom\Install\Install\Mount\Panther" -Force
```

1.2.2. Deployment engine: add

Add "Automatically add languages installed on Windows systems" to the D:\ISOTEMP_Custom\Install\Install\Mount\Yi\Engine directory.

1.2.2.1. Deployment Engine: Copy

Copy D:\Multilingual\Engine to D:\ISOTEMP_Custom\Install\Install\Mount\Yi\Engine

```
Copy-Item "D:\Multilingual\Engine" -Destination "D:\ISOTEMP_Custom\Install\Install\Mount\Yi\Engine" -Recurse -Force
```

1.2.2.2. Deployment engine: custom deployment tags

```
$Flag = @(

    "Is_Mark_Sync" # Allow global search and synchronization of deployment tags

    # Prerequisite deployment

#    "Auto_Update" # Allow automatic updates

#    "Use_UTF8" # Beta: Global language support using Unicode UTF-8

    "Disable_Network_Location_Wizard" # Network Location Wizard

    "Disable_Cleanup_Appx_Tasks" # Appx Cleanup and maintenance tasks

    "Disable_Cleanup_On_Demand_Language" # Prevent cleanup of unused on-demand feature language packs

    "Disable_Cleanup_Unsed_Language"    # Prevent cleaning of unused language packs
```

```
   "Prerequisites_Reboot" # Restart your computer

   # Complete first deployment

#   "Popup_Engine" # Allow the deployment engine main interface to pop up for the first time

#   "Allow_First_Pre_Experience" # Allow first preview, as planned

   "Reset_Execution_Policy" # Restore PowerShell execution policy: Restricted

   "Clear_Solutions" # Delete the entire solution

   "Clear_Engine" # Delete the deployment engine and keep the others

#   "First_Experience_Reboot" # Restart your computer

)

ForEach ($item in $Flag) {

   Write-host "   $($item)" -ForegroundColor Green

   New-Item -Path "D:\ISOTEMP\sources\`$OEM$\`$1\Yi\Engine\Deploy\Allow" -ItemType Directory -ErrorAction SilentlyContinue | Out-Null

   Out-File -FilePath "D:\ISOTEMP\sources\`$OEM$\`$1\Yi\Engine\Deploy\Allow\$($item)" -Encoding utf8 -ErrorAction SilentlyContinue

}
```

2. Deployment Engine: Advanced

2.1. Deployment engine: adding process

After copying the deployment engine, you can add deployment tags to intervene in the installation process.

2.2. Unattended solution

When the customization is unattended, please modify it simultaneously if the following files exist:

- D:\ISOTEMP\Autounattend.xml
- D:\ISOTEMP\Sources\Unattend.xml
- D:\ISOTEMP\sources\$OEM$\$$\Panther\unattend.xml
- D:\ISOTEMP_Custom\Install\Install\Mount\Panther\Unattend.xml

2.2.1. Multilingual or monolingual

In multi-language and monolingual, you can switch between each other. When replacing, please replace all the same ones in the file.

2.2.1.1. Multi-language

```
<UILanguage>%OSDUILanguage%</UILanguage>
```

```
<InputLocale>%OSDInputLocale%</InputLocale>
<SystemLocale>%OSDSystemLocale%</SystemLocale>
<UILanguage>%OSDUILanguage%</UILanguage>
<UILanguageFallback>%OSDUILanguageFallback%</UILanguageFallback>
<UserLocale>%OSDUserLocale%</UserLocale>
```

2.2.1.2. Monolingual

A single language needs to specify a language tag, for example, specify a language tag: zh-CN

```
<UILanguage>zh-CN</UILanguage>
<InputLocale>zh-CN</InputLocale>
<SystemLocale>zh-CN</SystemLocale>
<UILanguage>zh-CN</UILanguage>
<UILanguageFallback>zh-CN</UILanguageFallback>
<UserLocale>zh-CN</UserLocale>
```

2.2.2. User plan

By default, the self-created user Administrator is used and logged in automatically. It can be switched by modifying the following configuration: self-created or customized user.

2.2.2.1. Self-created user Administrator

By default, the self-created user: Administrator is used and logged in automatically, inserted between <OOBE> and </OOBE>.

```
<UserAccounts>
  <LocalAccounts>
    <LocalAccount wcm:action="add">
      <Password>
        <Value></Value>
        <PlainText>true</PlainText>
      </Password>
      <Description>Administrator</Description>
      <DisplayName>Administrator</DisplayName>
      <Group>Administrators</Group>
      <Name>Administrator</Name>
    </LocalAccount>
  </LocalAccounts>
</UserAccounts>
<AutoLogon>
  <Password>
    <Value></Value>
    <PlainText>true</PlainText>
  </Password>
  <Enabled>true</Enabled>
  <Username>Administrator</Username>
</AutoLogon>
```

2.2.2.2. Custom user

After setting up a custom user and installing the system, in OOBE, you can choose settings such as local and online users.

2.2.2.3. Delete

Username: Removed from start <UserAccounts> to </UserAccounts>

Autologin: Remove from start <AutoLogon> to </AutoLogon>

2.2.2.4. Replace

From the beginning <OOBE> to </OOBE>

```
<OOBE>
  <ProtectYourPC>3</ProtectYourPC>
  <HideEULAPage>true</HideEULAPage>
  <HideWirelessSetupInOOBE>true</HideWirelessSetupInOOBE>
</OOBE>
```

D. Generate ISO

1. Download OScdimg

Select the Oscdimg version according to the architecture, and save it to: D:\ after downloading. To save in other paths, please enter the absolute path of OScdimg.exe;

1.1. x64

https://github.com/ilikeyi/Solutions/raw/refs/heads/main/_Encapsulation/Modules/AIO/Oscdimg/amd64/oscdimg.exe

1.2. x86

https://github.com/ilikeyi/Solutions/raw/refs/heads/main/_Encapsulation/Modules/AIO/Oscdimg/x86/oscdimg.exe

1.3. arm64

https://github.com/ilikeyi/Solutions/raw/refs/heads/main/_Encapsulation/Modules/AIO/Oscdimg/arm64/oscdimg.exe

2. Use the oscdimg command line to generate an ISO file and save it to: D:\ISOTEMP.iso

- ISO.ps1
  - \Expand\ISO.ps1
  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.2022/Expand/ISO.ps1

- Copy the code

```
$Oscdimg = "D:\Oscdimg.exe"; $ISO = "D:\ISOTemp"; $Volume = "ISOTemp"; $SaveTo = "D:\ISOTemp.iso"; $Arguments = @("-m", "-o", "-u2", "-udfver102", "-l""$($Volume)""", "-bootdata:2#p0,e,b""$($ISO)\boot\etfsboot.com""#pEF,e,b""$($ISO)\efi\microsoft\boot\efisys.bin""", """$($ISO)""", """$($SaveTo)"""); Start-Process -FilePath $Oscdimg -ArgumentList $Arguments -wait -nonewwindow;
```

Chapter 2 Deploy

A. Precautions before deployment

1. When choosing the location to install Windows 11

First select "Disk Partition", then select "Format Partition", and then click "Next". If the disk is not formatted, an overwrite installation will cause known problems:

- Using the Administrator user:
  - The application icon will display a UAC security warning;
  - Right-clicking and running PS1 as administrator does not work;

2. When running the installer

- You must not run Setup.exe from the ISO on a system you are currently using to enter the installer, otherwise it will cause an error during the copying of the OEM directory. This problem is a bug in the installer.
- When installing the system onto a physical device, you can only access the installation program using a USB flash drive, CD-ROM, PE, or network installation (PXE boot), and a "clean install" is required.

B. Deploying the operating system to the physical device

Before the physical device can boot into the system, you must select one of the following options in "Preparing prerequisites for booting the installation program": "Create a bootable physical storage medium, CD-ROM, or install over a network (PXE boot)" and complete the process.

1. Prerequisites for preparing the installation program

When deploying system installation files to physical storage devices, you should prepare a removable drive or CD-ROM to store the Windows operating system installation files.

1.1. Create a bootable installation physical storage medium

If you plan to store more than 16GB of storage in a portable hard drive or USB drive, when purchasing the portable drive, you should choose one with dual interfaces (Type-C and USB 3.1). The advantages of choosing a drive with both Type-C and USB 3.1 interfaces are:

- If drivers are missing during system installation: You can download them via your mobile phone, then plug in a Type-C removable drive for file management and copy the downloaded drivers to the removable drive.
- In daily use, you can use the Type-C connection to your phone to store temporary files and back up data.

1.1.1. Disk Partitioning

- For installing Windows on macOS (excluding M series) or PC, storing the Windows installation files in FAT32 format is the best solution.
- If the storage device is less than 16GB, it is recommended that you create a single partition.
- For storage devices larger than 32GB, it is recommended that you partition them into 3 partitions. Partitioning scheme:

  Partition 1, after allocating 6 GB to Partition 2, the remaining disk space is used for: storing Windows system installation files (customizable deployment after extracting Yi.Deploy.Rule.iso), and for daily operations such as storing temporary files and backup data.

  Partition 2: Allocate 6GB of disk space for storing the PE system. Recommended.

  ○ Sergei Strelec | https://sergeistrelec.name/winpe-10-8-sergei-strelec-english

  ○ Hirens BootCD | https://www.hirensbootcd.org

1.1.2. Copy the system installation files to the disk partition.

To format the partition, select Fat32, then copy all files from the ISO to the root directory of the USB drive to complete the creation process.

1.2. CD-ROM

1.2.1. Prepare a CD/DVD burner.

1.2.2. Prepare a blank CD.

1.2.3. After selecting "ISO file", right-click and select the "burn" function, then click Start burning and wait for it to complete.

1.3. Install via network (PXE boot)

Each software has a different usage method; please learn it before use. You can choose from the following:

1.3.1. Serva | https://www.vercot.com/~serva

1.3.2. TinyPXE Server | http://labalec.fr/erwan/?page_id=958

1.3.3. iventory | https://www.iventoy.com

2. Physical device system installation guide

- When powering on, press different keys depending on the motherboard to enter the "Boot Menu" and then select disk boot from the BIOS menu. Common BIOS boot hotkeys include: F2, F8, F9, F11, F12, and ESC.
- Choose the appropriate menu based on the boot medium: CD-ROM, PXE, or select the partition that has been recognized by the USB drive.

C. Deploy to a system that is currently in use, and add the native boot VHD to the existing boot menu

1. Create VHD/VHDX files

1.1. Interactive Disk Management

Open "Disk Management (diskmgmt.msc)", select "Actions", select "Create VHD", and the "Create or Attach Virtual Disk" dialog box will pop up:

○ Set "Location: D:\OS.vhdx"

○ Set "Virtual disk size: 120, selection: GB"

○ Select "Virtual Disk Format: VHDX"

○ Select "Dynamically Expand"

After clicking "OK", a new disk will be added to the disk area, in the following order:

○ Select "Disk 2" (please ensure you select the correct disk before selecting), then right-click and select "Initialize Disk".

○ After selecting "Disk 2 Partition", right-click and select "New Simple Volume Wizard" to complete.

1.2. Command Line Creation

Quickly create (save to: D:\OS.vhdx, virtual disk size: 120GB, dynamically expandable, drive letter assigned: Q), command line:

○ Type Diskpart and press Enter. In this dialog box, run the following commands in sequence:

```
Create Vdisk File="D:\OS.vhdx" Maximum=122880 Type=expandable

Select Vdisk file="D:\OS.vhdx"

Attach Vdisk

Create Partition Primary

Format Fs=NTFS Label="VOS" Quick

Assign Letter=Q

Exit
```

2. Apply the system from Install.wim to the VHD/VHDX file.

After completing the "Create VHD/VHDX file" step, you can apply the index number specified in Install.wim to the specified drive letter. The settings are: Image file: D:\ISOTemp\Sources\Install.wim, Index number: 1, Apply to drive letter: Q, Command line:

```
Expand-WindowsImage -ImagePath "D:\ISOTemp\Sources\install.wim" -ApplyPath "Q:\" -Index 1
```

3. Add native boot VHD to the existing Windows 10/11 boot menu

3.1. Backup BCD

Use the BCDedit tool with the /export option to back up your BCD storage. At the command prompt, run: `bcdedit /export c:\bcdbackup`

3.2. Copy an existing Windows 10/11 installation boot entry. Then modify the copy to use it as the VHD boot entry. At the command prompt, run:

```
bcdedit /copy {default} /d "VHD New Windows 11"
```

When the BCDedit command completes successfully, it returns {GUID} as output in the command prompt window.

3.3. 3.3. Locate {GUID} in the command prompt output of the previous command. Copy the GUID (including the curly braces) for use in subsequent steps.

3.4. Configure the device and operating system device options for the VHD boot entry; you must replace {GUID}. At the command prompt, run:

```
bcdedit /set {default} device vhd="[D:]\OS.vhdx"
```

```
bcdedit /set {default} osdevice vhd="[D:]\OS.vhdx"
```

3.5. Optional:

3.5.1. Set VHD as the default boot option. After the computer restarts, the boot menu will display all Windows installations on the computer, and VHD will start after the operating system selection countdown ends. At the command prompt, type:

```
bcdedit /default {guid}
```

3.5.2. Some x86-based systems require kernel boot configuration options to detect certain hardware information and successfully boot locally from the VHD. At the command prompt, type:

```
bcdedit /set {guid} detecthal on
```

For more information on how to use the BCDedit tool, please see this Microsoft website.

D. Deploy to virtual machine

Common examples include Windows' built-in Hyper-V, VMware Workstation Pro, and VirtualBox.

1. Enable Hyper-V

- Ensure your system meets the requirements: Windows 10/11 Professional or Enterprise edition, equipped with a 64-bit processor with Second-Level Address Translation (SLAT), and at least 4 GB of memory. Hyper-V is not available for Windows Home edition.
- Once Hyper-V is enabled, you can use Hyper-V Manager or PowerShell commands to begin creating virtual machines.
- Hyper-V is a virtualization platform built into Windows that allows you to create and manage virtual machines. Here's how to enable Hyper-V on your system.

1.1. Using PowerShell

Open Windows PowerShell as administrator: Press Win + S, type "PowerShell", right-click, and select "Run as administrator". Run the following command to enable Hyper-V:

```
Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Hyper-V -All
```

When prompted by the system, type Y to restart your computer to complete the installation. When prompted by the system again, restart your computer.

1.2. Select "Turn Windows features on or off"

1.2.1. Open the "Turn Windows features on or off" dialog box

1.2.1.1. After running OptionalFeatures, a "Turn Windows features on or off" message pops up.

1.2.1.2. Use Control Panel

- ○ Open Control Panel: Press Win + S, type "Control Panel", and then open it.
- ○ Navigate to Programs > Programs and Features > Turn Windows features on or off.

1.2.2. After opening, check the Hyper-V checkbox and then click OK.

1.2.3. Restart your computer to complete the installation.

2. VMware Workstation Pro

Official website | https://www.vmware.com/products/desktop-hypervisor/workstation-and-fusion

3. VirtualBox

Official website | https://www.virtualbox.org

E. Advanced deployment

1. Fully automatic installation

- To achieve fully automated installation during batch deployment, it is necessary to modify the pre-configuration.
- Multiple hard drives

  Batch installation prioritizes determining the number of hard drives and initializing them, then implements different solutions based on the different hard drive requirements.

2. Deployment Engine

If you've added a deployment engine (Multilingual, YiSuite), you can customize the deployment process. Download the template: Yi.Engine.Deploy.Rule.iso. After downloading, extract it to any disk, or mount the ISO or modify its contents during the initial deployment. Learn more:

- Multilingual | https://github.com/ilikeyi/Multilingual
- YiSuite | https://github.com/ilikeyi/YiSuite

2.1. Custom Office Deployment

After downloading the template Yi.Engine.Deploy.Rule.iso, extract it to any drive (example: D:\Yi), select the version, and then run the download script.

2.1.1. Office 365

```
& "D:\Yi\Office\365\amd64\Download.Office.amd64.ps1"

& "D:\Yi\Office\365\amd64\Download.Office.x86.ps1"
```

2.1.2. Office 2024

```
& "D:\Yi\Office\2024\amd64\Download.Office.amd64.ps1"

& "D:\Yi\Office\2024\amd64\Download.Office.x86.ps1"
```

2.1.3. Office 2021

```
& "D:\Yi\Office\2021\amd64\Download.Office.amd64.ps1"

& "D:\Yi\Office\2021\amd64\Download.Office.x86.ps1"
```

Before installation, please download the Office installation package. After downloading, search in this order: 365, 2024, 2021;

If a usable installation package is found: the first deployment will search for the Yi.ps1 file and execute the Office installation process;

To run manually: navigate to D:\Yi\Office\{ 365 | 2024 | 2021 }\{ amd64 | x86 }\Office\Install.Office.ps1 to start the installation process.

Sustainability:

Custom deployment: By using a portable USB drive, you can save the Yi directory to a partition on the USB drive. Keep the USB drive connected throughout the system installation process, and the custom deployment strategy will run automatically.

2.2. User solutions should be provided in advance.

The default setting uses the self-created user Administrator and logs in automatically. You can switch between self-created and custom users by modifying the following configuration.

2.2.1. Self-created user Administrator

The default user is Administrator, who will log in automatically. This user name is inserted between <OOBE> and </OOBE>.

```
<UserAccounts>
  <LocalAccounts>
    <LocalAccount wcm:action="add">
      <Password>
        <Value></Value>
        <PlainText>true</PlainText>
      </Password>
      <Description>Administrator</Description>
      <DisplayName>Administrator</DisplayName>
      <Group>Administrators</Group>
      <Name>Administrator</Name>
    </LocalAccount>
  </LocalAccounts>
</UserAccounts>
<AutoLogon>
  <Password>
    <Value></Value>
    <PlainText>true</PlainText>
  </Password>
  <Enabled>true</Enabled>
  <Username>Administrator</Username>
</AutoLogon>
```

2.2.2. OOBE interactive creation of new users

After setting up a custom user and completing the system installation, you can select local or online user settings in OOBE.

2.2.2.1. Delete

Username: Delete from the beginning of <UserAccounts> to </UserAccounts>

Automatic Login: Delete from the beginning of <AutoLogon> to </AutoLogon>

2.2.2.2. Replace

From the beginning <OOBE> to </OOBE>

```
<OOBE>
  <ProtectYourPC>3</ProtectYourPC>
  <HideEULAPage>true</HideEULAPage>
  <HideWirelessSetupInOOBE>true</HideWirelessSetupInOOBE>
</OOBE>
```

Chapter 3 Common problem

A. Clean all mounts to

Close any applications that may be accessing files in the image, including File Explorer.

```
Dismount-WindowsImage -Path "D:\ISOTEMP_Custom\Install\Install\Mount" -Discard

Dismount-WindowsImage -Path "D:\ISOTEMP_Custom\Install\WinRE\Mount" -Discard

Dismount-WindowsImage -Path "D:\ISOTEMP_Custom\Boot\Boot\Mount" -Discard
```

B. Fix the problem of abnormal mounting

1. View mounted

```
Get-WindowsImage -Mounted
```

2. Delete the DISM mount record saved in the registry

```
Remove-Item -Path "HKLM:\SOFTWARE\Microsoft\WIMMount\Mounted Images\*" -Force -Recurse -ErrorAction SilentlyContinue
```

3. Delete all resources associated with the corrupted mounted image

```
Clear-WindowsCorruptMountPoint

Dism /cleanup-wim
```

C. Clean up

A large number of temporary files will be generated during the packaging process. Installation files will be temporarily released when installing InBox Apps applications, installing cumulative updates, and installing language packs. Therefore, unscheduled cleaning of outdated ones will occupy a large amount of disk space for a long time. It is recommended that you try the following methods to achieve this. Cleanup plan to free up more space:

1. Common logs

1.1. Clean using the command line

```
$TempPaths = @( $env:Temp; "$($env:SystemRoot)\Logs\DISM"; ); foreach ($TempPath in $TempPaths) { if (Test-Path -Path $TempPath) { write-host "   $($TempPath)" -ForegroundColor Green; Get-ChildItem -Path $TempPath -Recurse -Force | ForEach-Object { try { Remove-Item $_.FullName -Force -Recurse -ErrorAction SilentlyContinue | Out-Null; } catch { write-host $_ -ForegroundColor Red; } }}}
```

1.2. Manual deletion

1.2.1. DISM log

Using the "Disk Cleanup" function, the logs generated by DISM cannot be cleaned and need to be deleted manually. Path: {system disk}:\Windows\Logs\DISM

1.2.2. Temporary directory

Using the "Disk Cleanup" function, files in the temporary directory cannot be cleaned and manual operation is required. Run: %Temp% to quickly locate and open the temporary directory. Path: {system disk}:\Users\{username}\AppData\Local\ Temp

1.2.3. Clear the command line records of "Terminal"

```
Remove-Item -Path (Get-PSReadlineOption).HistorySavePath -ErrorAction SilentlyContinue
```

After cleaning up command line records, you need to restart the "Terminal" to take effect.

2. Disk cleanup

Run cleanmgr, selecting the disks and types to clean.

Chapter 4 Known issues

1. Add Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~zh-CN~.cab to Windows Server 2022 Standard Core, Windows Server

2022 Datacenter Core will add Microsoft-Windows-PowerShell-ISE-FOD -Package~31bf3856ad364e35~amd64~zh-CN~10.0.20348.1, an error will be reported when deleting it, and the operation is not recommended for the time being.

Yi's Solutions

This copy packaging tutorial is part of Yi's Solutions content, learn more:

- Yi's official website | https://fengyi.tel/solutions
- Github | https://github.com/ilikeyi/solutions

Author: Yi

EMail: 775159955@qq.com, ilikeyi@outlook.com

Document version: 4.5

Translation: Chinese to English version

Initial public offering time: 4 / 2023

All scripts included in the document, last tested: 1 / 2026

Document last updated: 1 / 2026

Suggestions or feedback: https://github.com/ilikeyi/solutions/issues