Yi’ Solutions

# Microsoft Windows 11 22H2

PACKAGING TUTORIAL: Different system versions have different packaging methods. The packaging process includes: "Language pack: add, associate, delete", "Drive: add, delete", "Cumulative update: add, delete", "InBox Apps: add, Update, mark" 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

1. [Encapsulation](#_Deployment_image)
2. [Report](#_Report)
3. [Common problem](#_Common_problem)
4. [Known issues](#_Known_issues)

## Table of contents

1. Encapsulation Page 5
   * 1. Prerequisites Page 5
2. ISO tools Page 5
3. Requirements Page 5
4. System installation package Page 5
5. Language pack Page 6
   1. Learn Page 6
   2. Language pack: Download Page 9
   3. Language pack: Fixed Page9
6. InBox Apps Page 9
7. Windows Security Page 10
8. Command line Page 10
   * 1. Language Pack: Extract Page 11
9. Language pack: Ready Page 11
10. Language pack: Scheme Page 11
11. Execute the extract command Page 11
    * 1. Custom encapsulation Page 17
12. Custom encapsulation Install.wim Page 17
13. View Install.wim details Page 17
14. Specify the path to mount Install.wim Page 17
15. Start mounting Install.wim Page 17
    1. Custom encapsulation WinRE.wim Page 17
       1. View WinRE.wim details Page 17
       2. Specify the path to mount WinRE.wim Page 17
       3. Start mounting WinRE.wim Page 18
       4. Language pack: WinRE Page 18
          1. Language pack: add Page 18
          2. Offline image language: change Page 19
             1. Change default language, regional settings, and other international settings Page 19
             2. View available language settings Page 20
          3. Language packs: Removed Page 20
          4. Components: All packages installed in the image Page 22
       5. Cumulative updates Page 22
          1. Add Page 22
          2. Delete Page 22
          3. Solid update Page 22
             1. Clean components after curing and updating Page 23
       6. Drive Page 23
       7. Save image: WinRE.wim Page 23
       8. Unmount image: WinRE.wim Page 23
       9. After rebuilding WinRE.wim, the file size can be reduced Page 23
       10. Backup WinRE.wim Page 24
       11. Replace WinRE.wim within the Install.wim image Page 24

1. Language pack Page 24

* 1. Language pack: add Page 24
  2. Offline image language: change Page 30
     1. Change default language, regional settings, and other international settings Page 30
     2. View available language settings Page 30
  3. Language packs: Removed Page 31
  4. Components: All packages installed in the image Page 33

1. InBox Apps Page 33
   1. Learn Page 33
   2. InBox Apps: Pre-install the application Page 33
   3. InBox Apps: Installed Page 35
   4. Remove all installed pre-applications Page 36
   5. Region tag: Understanding Page 36
   6. Region tag: adding method Page 36
   7. InBox Apps: Install Page 39
   8. Region tag: delete Page 49
   9. InBox Apps: optimization Page 49
2. Cumulative updates Page 50
   1. Initial version Page 50
   2. Other Version Page 51
   3. Solidify Updated Page 51
      1. Clean components after curing and updating Page 51
3. Drive Page 52
4. Deployment engine: Add Page 52
5. Health Page 52
6. Replace WinRE.wim Page 52
7. Save image: Install.wim Page 52
8. Unmount image: Install.wim Page 52
9. How to batch replace WinRE.wim in all index numbers in Install.wim Page 52
   1. Obtain WimLib Page 52
   2. How to extract and update WinRE.wim in Install.wim Page 52
10. Rebuilding Install.wim reduces file size Page 53
11. Custom encapsulation boot.wim Page 54
12. View Boot.wim details Page 54
13. Specify the path to mount Boot.wim Page 54
14. Start mounting Boot.wim Page 54
15. Language pack: Boot Page 54
    1. Language pack: Add Page 54
    2. Offline image language: change Page 56
       1. Change default language, regional settings, and other international settings Page 56
       2. View available language settings Page 56
    3. Language pack: Removed Page 56
    4. Components: All packages installed in the image Page 59
    5. Language: Repair Page 59
    6. Language packs: sync to ISO installer Page 59
    7. Regenerate Lang.ini Page 59
       1. Regenerate the mounted directory lang.ini Page 59
       2. After regenerating lang.ini, sync to the installer Page 59
16. Cumulative updates Page 60
    1. Add Page 60
    2. Delete Page 60
    3. Solid update Page 60
       1. Clean components after curing and updating Page 60
17. Drive Page 60
18. Other Page 60
    1. Bypass TPM check during installation Page 60
19. Save image: Boot.wim Page 61
20. Unmount image: Boot.wim Page 61
21. Deployment engine Page 61
22. Add method Page 61
23. Deployment Engine: Advanced Page 64
    * 1. ISO Page 67
24. Generate ISO Page 67
25. Bypass TPM installation check Page 67
26. Report Page 68
27. Reports: Different versions Page 68
28. Report: Language installation packages apply to Install.wim Page 70
29. Report: Language installation packages apply to WinRE.wim Page 71
30. Report: Language installation packages apply to Boot.wim Page 73
31. Common problem Page 74
32. Clean all mounts to Page 74
33. Fix the problem of abnormal mounting Page 74
34. Clean directories Page 74
35. Known issues Page 74

## Encapsulation

### Prerequisites

#### ISO tools

Use a software that can edit ISO files, such as: PowerISO, DAEMON Tools, ISO Workshop;

#### Requirements

##### System installation package

Keywords: iteration, cross-version, major version, cumulative update, initial release

* 1. illustrate
     1. Please remake the image when each version is updated, for example, when crossing from 21H1 to 22H2, avoid other compatibility problems, and do not make the image based on the old image;
     2. The regulation has been clearly communicated to packagers in various forms by some OEMs, and direct upgrades from iterative versions are not allowed;
     3. Please use "Initial Version" and "Developer Edition" for production. There was a brief appearance in the official Microsoft documentation that the initial version must be used in production, but this sentence was later deleted in the official documentation;
     4. After installing the language pack, you must re-add the cumulative update (the same version number), and if you do not add the cumulative update, problems such as "garbled characters" and "interface flashback" will occur.
     5. Evolutionary process: Windows 11 22H2, Build 22621.382 + KB5027303 = OS Build 22621.1928
  2. Prepare to download the initial or developer version
     1. x64
        1. [en-us\_windows\_11\_business\_editions\_version\_22h2\_x64\_dvd\_17a08ce3.iso](https://files.rg-adguard.net/file/58857e8e-0da6-f4de-9d1b-048944edbd84)
        2. [en-us\_windows\_11\_consumer\_editions\_version\_22h2\_x64\_dvd\_e630fafd.iso](https://files.rg-adguard.net/file/1e7e4b34-5bda-09e8-3a06-fae7b4649406)
        3. [en-us\_windows\_11\_iot\_enterprise\_version\_22h2\_x64\_dvd\_eb39bf70.iso](https://files.rg-adguard.net/file/9f20203f-5692-dff8-6554-27cdc770cfb0)
     2. arm64
        1. [en-us\_windows\_11\_iot\_enterprise\_version\_22h2\_arm64\_dvd\_f09f5d29.iso](https://files.rg-adguard.net/file/f4a64de6-e211-11a1-d95c-95f0b9274034)
  3. After the sample download en-us\_windows\_11\_business\_editions\_version\_22h2\_x64\_dvd\_17a08ce3.iso,Unzip to: D:\en-us\_windows\_11\_business\_editions\_version\_22h2\_x64\_dvd\_17a08c

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

* 1. After decompression is complete, change the directory en-us\_windows\_11\_business\_editions\_version\_22h2\_x64\_dvd\_17a08c change to D:\OS\_11
  2. All scripts and all paths are set to D:\OS\_11 as the image source.
  3. Installation configuration
     1. Learn: [Windows Setup Edition Configuration and Product ID Files ( EI.cfg and PID.txt )](https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/windows-setup-edition-configuration-and-product-id-files--eicfg-and-pidtxt?view=windows-11)
     2. Known issues
        1. When there is no Ei.cfg, ISO boot installation will report an error when selecting certain versions, prompting: Windows cannot find the Microsoft Software License terms. Make sure the insttallation sources are valid and restart the installation.
        2. How to solve it? Add ei.cfg to D:\OS\_11\Sources and create ei.cfg:

@"

[Channel]

volume

[VL]

1

"@ | Out-File -FilePath "D:\OS\_11\sources\EI.CFG" -Encoding Ascii

* 1. Report

Check the installed image package: language packs, associated language pack components, pre-installed InBox Apps applications and other detailed information. After downloading the report, use Excel software to open the .xlsx.

* + 1. Report: Overview

Contains: ISO system installation package, image version, region, region association, associated language pack components, etc.

[Report.Overview.xlsx](https://raw.githubusercontent.com/ilikeyi/solutions/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.Overview.xlsx)

* + 1. Report: Complete

Inclusion: ISO system installation package, image version, region, region association, associated language pack components, installed InBox Apps pre-installed applications, etc.

[en-us](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.en-US.xlsx), [ar-sa](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.ar-sa.xlsx), [bg-bg](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.bg-bg.xlsx), [cs-cz](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.cs-cz.xlsx), [da-dk](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.da-dk.xlsx), [de-de](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.de-de.xlsx), [el-gr](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.el-gr.xlsx), [en-gb](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.el-gr.xlsx), [es-es](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.es-es.xlsx), [es-mx](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.es-mx.xlsx), [et-ee](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.et-ee.xlsx), [fi-fi](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.fi-FI.xlsx), [fr-ca](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.fr-ca.xlsx), [fr-fr](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.fr-fr.xlsx), [he-il](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.he-il.xlsx), [hr-hr](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.hr-hr.xlsx), [hu-hu](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.hu-hu.xlsx), [it-it](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.it-IT.xlsx), [ja-jp](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.ja-jp.xlsx), [ko-kr](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.ko-kr.xlsx), [lt-lt](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.lt-lt.xlsx), [lv-lv](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.lv-lv.xlsx), [nb-no](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.nb-no.xlsx), [nl-nl](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.nl-nl.xlsx), [pl-pl](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.pl-pl.xlsx), [pt-br](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.pt-br.xlsx), [pt-pt](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.pt-pt.xlsx), [ro-ro](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.ro-ro.xlsx), [ru-ru](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.ru-ru.xlsx), [sk-sk](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.sk-sk.xlsx), [sl-si](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.sl-si.xlsx), [sr-latn-rs](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.sr-latn-rs.xlsx), [sv-se](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.sv-se.xlsx), [th-th](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.th-th.xlsx), [tr-tr](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.tr-tr.xlsx), [uk-ua](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.uk-ua.xlsx), [zh-cn](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.zh-cn.xlsx), [zh-tw](https://github.com/ilikeyi/solutions/raw/main/_Documents/Attachment/OS.11/23H2/Expand/Install/Report/Report.zh-TW.xlsx)

##### Language pack

###### Learn

As you read, please understand the important highlights of "Blue".

* + 1. [Languages overview](https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/languages-overview?view=windows-11)
    2. [Add languages to a Windows 11 image](https://learn.microsoft.com/en-us/azure/virtual-desktop/windows-11-language-packs)
    3. [Language and region Features on Demand (FOD)](https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/features-on-demand-language-fod?view=windows-11)
       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](https://download.microsoft.com/download/7/6/0/7600F9DC-C296-4CF8-B92A-2D85BAFBD5D2/Windows-10-1809-FOD-to-LP-Mapping-Table.xlsx)" learn more.
* In "Language Pack: Extract", the automatic recognition function has been added, and you can understand the functions: Function Match\_Required\_Fonts
  + - 1. 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.

Known regional associations:

* + - * 1. Region: ar-sa, Optional associated areas: en-US, associated installation package:

Microsoft-Windows-LanguageFeatures-Basic-es-us-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~.cab

* + - * 1. Region: bg-bg, Optional associated areas: en-US, associated installation package:

Microsoft-Windows-LanguageFeatures-Basic-es-us-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~.cab

* + - * 1. Region: da-dk, Optional associated areas: en-US, associated installation package:

Microsoft-Windows-LanguageFeatures-Basic-es-us-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~.cab

* + - * 1. Region: el-gr, Optional associated areas: en-US, associated installation package:

Microsoft-Windows-LanguageFeatures-Basic-es-us-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~.cab

* + - * 1. Region: fr-ca, Optional associated areas: en-US, fr-fr, associated installation package:

Microsoft-Windows-LanguageFeatures-Basic-es-us-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-Basic-fr-fr-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-Handwriting-fi-fi-Package~31bf3856ad364e35~amd64~~.cab

* + - * 1. Region: he-il, Optional associated areas: en-US, associated installation package:

Microsoft-Windows-LanguageFeatures-Basic-es-us-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~.cab

* + - * 1. Region: ru-ru, Optional associated areas: en-US, associated installation package:

Microsoft-Windows-LanguageFeatures-Basic-es-us-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~.cab

* + - * 1. Region: th-th, Optional associated areas: en-US, associated installation package:

Microsoft-Windows-LanguageFeatures-Basic-es-us-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~.cab

* + - * 1. Region: uk-ua, Optional associated areas: en-US, associated installation package:

Microsoft-Windows-LanguageFeatures-Basic-es-us-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~.cab

* + - * 1. Region: zh-TW , Optional associated areas: zh-HK, associated installation package:

Microsoft-Windows-LanguageFeatures-Basic-zh-hk-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-Speech-zh-hk-Package~31bf3856ad364e35~amd64~~.cab

Microsoft-Windows-LanguageFeatures-TextToSpeech-zh-hk-Package~31bf3856ad364e35~amd64~~.cab

* + - 1. Other region-specific requirements

When triggering a known area, a specific "package" needs to be added.

* + - * 1. Region: zh-TW

Package: Microsoft-Windows-InternationalFeatures-Taiwan-Package~31bf3856ad364e35~amd64~~.cab

Description: Supplemental support for Taiwan date formatting requirements. Package will be provided to customers located in Taiwan.

Recommendation: Preinstall only on devices shipping to the Taiwan market. Not installing this capability on devices causes any API calls to that use the Taiwan calendar to fail.

THERE IS CONTROVERSY:

* During the test, it was found that this package was not installed in the original image of Microsoft's official original Windows 11 However, there are known issues in the recommended items. In the end, it is consistent with the official Microsoft original version, and the packager can freely choose to install it. or not.

###### Language pack: Download

<https://software-static.download.prss.microsoft.com/dbazure/988969d5-f34g-4e03-ac9d-1f9786c66749/22621.1.220506-1250.ni_release_amd64fre_CLIENT_LOF_PACKAGES_OEM.iso>

###### Language pack: Fixed

* + 1. Select any website and open:
       1. <https://uupdump.net>
       2. <https://uup.ee>
       3. <https://osdump.com>
    2. After opening, search for keywords: 22621.382, select from the search results: Windows 11, version 22H2 (22621.382) amd64
    3. After opening, select "All files";
    4. Search the green part in the "All Files" page and download
       1. Applies to: Install.wim:
          1. MediaPlayer
       2. Applies to: WinRE.wim, none yet
       3. Applies to: Boot.wim, none yet
    5. After downloading all the files, scroll to the bottom of the page, download and run "Generate Rename Script (Windows)"
    6. Use ISO editing software, edit 22621.1.220506-1250.ni\_release\_amd64fre\_CLIENT\_LOF\_PACKAGES\_OEM.iso, and add the downloaded file to the [ISO]:\LanguagesAndOptionalFeatures directory;

##### InBox Apps

* 1. Download: <https://software-static.download.prss.microsoft.com/dbazure/888969d5-f34g-4e03-ac9d-1f9786c66749/22621.1778.230511-2102.ni_release_svc_prod3_amd64fre_InboxApps.iso>
  2. Download: <https://software-static.download.prss.microsoft.com/dbazure/988969d5-f34g-4e03-ac9d-1f9786c66749/22621.1.220506-1250.ni_release_amd64fre_InboxApps.iso> finally, Extract:
     1. Microsoft.HEVCVideoExtension
        1. Microsoft.HEVCVideoExtension\_8wekyb3d8bbwe.arm64.appx
        2. Microsoft.HEVCVideoExtension\_8wekyb3d8bbwe.arm64.xml
        3. Microsoft.HEVCVideoExtension\_8wekyb3d8bbwe.x64.appx
        4. Microsoft.HEVCVideoExtension\_8wekyb3d8bbwe.x64.xml
  3. Use the ISO editing tool to edit 22621.1778.230511-2102.ni\_release\_svc\_prod3\_amd64fre\_InboxApps.iso and add the extracted files to the [ISO]:\packages directory;

#### Windows Security

* When processing the encapsulation task, a large number of temporary files will be generated, and a large number of installation files will be released when installing the application in InBox Apps;
* Turning on Windows Security scans files and takes up a lot of CPU.
* In test: 1 hour and 22 minutes before shutdown, 20 minutes after shutdown.

How to close:

With the command line in green, hold down the Windows key and press R to launch Run.

1. Open Windows Security or run: windowsdefender:
2. Select "Virus & Threat Protection" or Run: windowsdefender://threat
3. Find "Virus & Threat Protection Settings", click "Manage Settings" or Run: windowsdefender://threatsettings, we recommend that you turn off some features:
   1. Real-time protection
   2. Cloud=delivered protection
   3. Automatic sample submission
   4. Tamper Protection
4. When you're not encapsulated, we recommend that you turn on Windows Security.

#### Command line

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

1. In this article, PS command line, green part, please copy it, paste it into the "Terminal" dialog box, press Enter and start running;
2. 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"

### Language Pack: Extract

#### Language pack: Ready

Mount 22621.1.220506-1250.ni\_release\_amd64fre\_CLIENT\_LOF\_PACKAGES\_OEM.iso or unzipped to any location;

#### Language pack: Scheme

##### Add

* 1. Language name: Simplified Chinese - China, Region: zh-CN, Scope of application: Install.Wim，Boot.Wim，WinRE.Wim

##### Delete

* 1. Language name: English - United States, Region: en-US, Scope of application: Install.Wim，Boot.Wim，WinRE.Wim

#### 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](Expand/Extract.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Extract.ps1>
* Copy the code

$ISO = "Auto"

$SaveTo = "D:\OS\_11\_Custom"

$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" ) }

)

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-Client-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-InternetExplorer-Optional-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"

"LanguagesAndOptionalFeatures\Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"

"LanguagesAndOptionalFeatures\Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~{Lang}~.cab"

"LanguagesAndOptionalFeatures\Microsoft-Windows-MediaPlayer-Package-AMD64-{Lang}.cab"

"LanguagesAndOptionalFeatures\Microsoft-Windows-MediaPlayer-Package-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-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"

"LanguagesAndOptionalFeatures\Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"

"LanguagesAndOptionalFeatures\Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~{Lang}~.cab"

"LanguagesAndOptionalFeatures\Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"

"LanguagesAndOptionalFeatures\Microsoft-Windows-WMIC-FoD-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"

"LanguagesAndOptionalFeatures\Microsoft-Windows-InternationalFeatures-{Specific}-Package~31bf3856ad364e35~amd64~~.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-appxdeployment\_{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-windowsupdate\_{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-CLIENT\_{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

}

}

}

}

ForEach ($item in $Extract\_language\_Pack) { Extract\_Language -Act $item.Act -NewLang $item.Tag -Expand $item.Scope }

### Custom encapsulation

#### Custom encapsulation Install.wim

##### View Install.wim details

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

$ViewFile = "D:\OS\_11\Sources\Install.wim"

Get-WindowsImage -ImagePath $ViewFile | Foreach-Object { Get-WindowsImage -ImagePath $ViewFile -index $\_.ImageIndex }

LOOP OPERATING AREA, START,

##### Specify the path to mount Install.wim

New-Item -Path "D:\OS\_11\_Custom\Install\Install\Mount" -ItemType directory -ea SilentlyContinue

##### Start mounting Install.wim

Default index number: 1

Mount-WindowsImage -ImagePath "D:\OS\_11\sources\install.wim" -Index "1" -Path "D:\OS\_11\_Custom\Install\Install\Mount"

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

###### 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;
* Synchronize to all index numbers to reduce the size of Install.wim, learn "How to batch replace WinRE.wim in all index numbers in Install.wim".
  + 1. View WinRE.wim details

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

$ViewFile = "D:\OS\_11\_Custom\Install\Install\Mount\Windows\System32\Recovery\WinRE.wim"

Get-WindowsImage -ImagePath $ViewFile | Foreach-Object { Get-WindowsImage -ImagePath $ViewFile -index $\_.ImageIndex }

* + 1. Specify the path to mount WinRE.wim

New-Item -Path "D:\OS\_11\_Custom\Install\WinRE\Mount" -ItemType directory -ea SilentlyContinue

* + 1. Start mounting WinRE.wim

Default index number: 1

Mount-WindowsImage -ImagePath "D:\OS\_11\_Custom\Install\Install\Mount\Windows\System32\Recovery\WinRE.wim" -Index "1" -Path "D:\OS\_11\_Custom\Install\WinRE\Mount"

* + 1. Language pack: WinRE
* 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, View the report "[Language installation package for WinRE.wim](#_Report:_Language_installation_1)".
* When adding languages, different schema versions must be corresponded, and if not, errors are reported during the addition process.
  + - 1. Language pack: add
* WinRE.Instl.lang.ps1
  + [\Expand\Install\WinRE\WinRE.Instl.lang.ps1](Expand/Install/WinRE/WinRE.Instl.lang.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/WinRE/WinRE.Instl.lang.ps1>
* Copy the code

$Mount = "D:\OS\_11\_Custom\Install\WinRE\Mount"

$Sources = "D:\OS\_11\_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"; }

@{ Match = "\*windowsupdate\*"; File = "winpe-windowsupdate\_zh-CN.cab"; }

@{ Match = "\*appxdeployment\*"; File = "winpe-appxdeployment\_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 pack 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

}

}

}

* + - 1. Offline image language: change
         1. Change default language, regional settings, and other international settings

Region: zh-CN

Dism /Image:"D:\OS\_11\_Custom\Install\WinRE\Mount" /Set-AllIntl:"zh-CN"

* + - * 1. View available language settings

Dism /Image:"D:\OS\_11\_Custom\Install\WinRE\Mount" /Get-Intl

* + - 1. Language packs: Removed
* After you add languages, if you want to deploy to a non-English locale, you can save space by removing the English language component. When you remove a language, uninstall the language components in the reverse order in which you added them.
* After adding the Chinese, delete "English - United States" in reverse, the Region tag: en-US, you need to extract the language pack in advance
* WinRE.Del.Specified.lang.Tag.ps1
  + [\Expand\Install\WinRE\WinRE.Del.Specified.lang.Tag.ps1](Expand/Install/WinRE/WinRE.Del.Specified.lang.Tag.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/WinRE/WinRE.Del.Specified.lang.Tag.ps1>
* Copy the code

$Lang = "en-US"

$Mount = "D:\OS\_11\_Custom\Install\WinRE\Mount"

$Sources = "D:\OS\_11\_Custom\Install\WinRE\Language\Del\en-US"

$Initl\_install\_Language\_Component = @()

Get-WindowsPackage -Path $Mount | ForEach-Object { $Initl\_install\_Language\_Component += $\_.PackageName }

$Language = @(

@{ Match = "\*windowsupdate\*"; File = "winpe-windowsupdate\_$($Lang).cab"; }

@{ Match = "\*appxdeployment\*"; File = "winpe-appxdeployment\_$($Lang).cab"; }

@{ Match = "\*hta\*"; File = "winpe-hta\_$($Lang).cab"; }

@{ Match = "\*opcservices\*"; File = "winpe-opcservices\_$($Lang).cab"; }

@{ Match = "\*rejuv\*"; File = "winpe-rejuv\_$($Lang).cab"; }

@{ Match = "\*WiFi\*"; File = "winpe-wifi\_$($Lang).cab"; }

@{ Match = "\*StorageWMI\*"; File = "winpe-storagewmi\_$($Lang).cab"; }

@{ Match = "\*WinPE-AppxPackaging\*"; File = "winpe-appxpackaging\_$($Lang).cab"; }

@{ Match = "\*-WMI-Package\*"; File = "winpe-wmi\_$($Lang).cab"; }

@{ Match = "\*wds-tools\*"; File = "winpe-wds-tools\_$($Lang).cab"; }

@{ Match = "\*srt\*"; File = "winpe-srt\_$($Lang).cab"; }

@{ Match = "\*srh\*"; File = "winpe-srh\_$($Lang).cab"; }

@{ Match = "\*Speech-TTS\*"; File = "winpe-speech-tts\_$($Lang).cab"; }

@{ Match = "\*scripting\*"; File = "winpe-scripting\_$($Lang).cab"; }

@{ Match = "\*Narrator\*"; File = "winpe-narrator\_$($Lang).cab"; }

@{ Match = "\*EnhancedStorage\*"; File = "winpe-enhancedstorage\_$($Lang).cab"; }

@{ Match = "\*AudioDrivers\*"; File = "winpe-audiodrivers\_$($Lang).cab"; }

@{ Match = "\*AudioCore\*"; File = "winpe-audiocore\_$($Lang).cab"; }

@{ Match = "\*ATBroker\*"; File = "winpe-atbroker\_$($Lang).cab"; }

@{ Match = "\*SecureStartup\*"; File = "winpe-securestartup\_$($Lang).cab"; }

@{ Match = "\*WinPE-LanguagePack-Package\*"; File = "lp.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 ($Component -like "\*$($Rule.Match)\*$($Lang)\*") {

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

Write-host " Language pack file: " -NoNewline; Write-host "$($Sources)\$($Rule.File)" -ForegroundColor Green

Write-Host " Deleting ".PadRight(22) -NoNewline

try {

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

Write-host "Finish" -ForegroundColor Green

} catch {

Write-host "Failed" -ForegroundColor Red

Write-host " $($\_)" -ForegroundColor Red

}

break

}

}

}

$InitlClearLanguagePackage = @()

Get-WindowsPackage -Path $Mount | ForEach-Object {

if ($\_.PackageName -like "\*$($Lang)\*") {

$InitlClearLanguagePackage += $\_.PackageName

}

}

if ($InitlClearLanguagePackage.count -gt 0) {

ForEach ($item in $InitlClearLanguagePackage) {

Write-Host "`n $($item)" -ForegroundColor Green

Write-Host " Deleting ".PadRight(22) -NoNewline

try {

Remove-AppxProvisionedPackage -Path $Mount -PackageName $item -ErrorAction SilentlyContinue | Out-Null

Write-host "Finish" -ForegroundColor Green

} catch {

Write-host "Failed" -ForegroundColor Red

Write-host " $($\_)" -ForegroundColor Red

}

}

}

* + - 1. Components: All packages installed in the image
         1. View

Get-WindowsPackage -Path "D:\OS\_11\_Custom\Install\WinRE\Mount" | Out-GridView

* + - * 1. Export to csv

$SaveTo = "D:\OS\_11\_Custom\Install\WinRE\Report.$(Get-Date -Format "yyyyMMddHHmmss").csv"

Get-WindowsPackage -Path "D:\OS\_11\_Custom\Install\WinRE\Mount" | Export-CSV -NoType -Path $SaveTo

Write-host $SaveTo -ForegroundColor Green

* + 1. Cumulative updates

To prepare the cumulative updates file available, change the example file name: KB\_WinRE.cab

* + - 1. Add

$KBPath = "D:\OS\_11\_Custom\Install\WinRE\Update\KB\_WinRE.cab"

Add-WindowsPackage -Path "D:\OS\_11\_Custom\Install\WinRE\Mount" -PackagePath $KBPath

* + - 1. Delete

$KBPath = "D:\OS\_11\_Custom\Install\WinRE\Update\KB\_WinRE.cab"

Remove-WindowsPackage -Path "D:\OS\_11\_Custom\Install\WinRE\Mount" -PackagePath $KBPath

* + - 1. Solid update

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

Dism /image:"D:\OS\_11\_Custom\Install\WinRE\Mount" /cleanup-image /StartComponentCleanup /ResetBase

* + - * 1. Clean components after curing and updating

$Mount = "D:\OS\_11\_Custom\Install\WinRE\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

}

}

* + 1. Drive
    2. Save image: WinRE.wim

Save-WindowsImage -Path "D:\OS\_11\_Custom\Install\WinRE\Mount"

* + 1. Unmount image: WinRE.wim

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

Dismount-WindowsImage -Path "D:\OS\_11\_Custom\Install\WinRE\Mount" -Discard

* + 1. After rebuilding WinRE.wim, the file size can be reduced
* WinRE.Rebuild.ps1
  + [\Install\WinRE\WinRE.Rebuild.ps1](Expand/Install/WinRE/WinRE.Rebuild.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/WinRE/WinRE.Rebuild.ps1>
* Copy the code

$FileName = "D:\OS\_11\_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 Rebuild".PadRight(28) -NoNewline

Export-WindowsImage -SourceImagePath $Filename -SourceIndex $\_.ImageIndex -DestinationImagePath "$($FileName).New" -CompressionType max

Write-Host "Finish`n" -ForegroundColor Green

}

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

}

* + 1. Backup WinRE.wim
* WinRE.Backup.ps1
  + [\Expand\Install\WinRE\WinRE.Backup.ps1](Expand/Install/WinRE/WinRE.Backup.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/WinRE/WinRE.Backup.ps1>
* Copy the code

$WimLibPath = "D:\OS\_11\_Custom\Install\Install\Update\Winlib"

$FileName = "D:\OS\_11\_Custom\Install\Install\Mount\Windows\System32\Recovery\WinRE.wim"

New-Item -Path $WimLibPath -ItemType Directory -ea SilentlyContinue

Copy-Item -Path $FileName -Destination $WimLibPath -Force

* + 1. Replace WinRE.wim within the Install.wim image
* After each mount Install.wim "[Replace WinRE.wim](#_Replace_WinRE.wim)";
* Learn "[Get all index numbers of Install.wim and replace the old WinRE.wim](#_Get_all_index)".

Process the files in the Install.wim image and end.

##### 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, View the report "[Language installation package for Install.wim](#_component:_All_packages)".
* When adding languages, different schema versions must be corresponded, and if not, errors are reported during the addition process.

###### Language pack: add

* Install.Instl.lang.ps1
  + [\Expand\Install\Install.Instl.lang.ps1](Expand/Install/Install.Instl.lang.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/Install.Instl.lang.ps1>
* Copy the code

Function Language\_Install

{

param($Mount, $Sources, $Lang)

$Initl\_install\_Language\_Component = @()

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:Init\_Folder\_All\_File\_Match\_Done = @()

$Script:Init\_Folder\_All\_File\_Exclude = @()

$Script:Search\_File\_Order = @(

@{

Name = "Fonts"

Description = "Fonts"

Rule = @(

@{ Match\_Name = "\*Fonts\*"; IsMatch = "No"; Capability = ""; }

)

}

@{

Name = "Basic"

Description = "Basic"

Rule = @(

@{ Match\_Name = "\*LanguageFeatures-Basic\*"; IsMatch = "Yes"; Capability = "Language.Basic~~~lb-LU~0.0.1.0"; }

@{ Match\_Name = "\*Client\*Language\*Pack\*"; IsMatch = "Yes"; Capability = "Language.Basic~~~lb-LU~0.0.1.0"; }

)

}

@{

Name = "OCR"

Description = "Optical character recognition"

Rule = @(

@{ Match\_Name = "\*LanguageFeatures-OCR\*"; IsMatch = "Yes"; Capability = "Language.OCR~~~fr-FR~0.0.1.0"; }

)

}

@{

Name = "Handwriting"

Description = "Handwriting recognition"

Rule = @(

@{ Match\_Name = "\*LanguageFeatures-Handwriting\*"; IsMatch = "Yes"; Capability = "Language.Handwriting~~~fr-FR~0.0.1.0"; }

)

}

@{

Name = "TextToSpeech"

Description = "Text-to-speech"

Rule = @(

@{ Match\_Name = "\*LanguageFeatures-TextToSpeech\*"; IsMatch = "Yes"; Capability = "Language.TextToSpeech~~~fr-FR~0.0.1.0"; }

)

}

@{

Name = "Speech"

Description = "Speech recognition"

Rule = @(

@{ Match\_Name = "\*LanguageFeatures-Speech\*"; IsMatch = "Yes"; Capability = "Language.Speech~~~fr-FR~0.0.1.0"; }

)

}

@{

Name = "RegionSpecific"

Description = "Other region-specific requirements"

Rule = @(

@{ Match\_Name = "\*InternationalFeatures\*"; IsMatch = "No"; Capability = ""; }

)

}

@{

Name = "Retail"

Description = "Retail demo experience"

Rule = @(

@{ Match\_Name = "\*RetailDemo\*"; IsMatch = "Yes"; Capability = ""; }

)

}

@{

Name = "Features\_On\_Demand"

Description = "Features on demand"

Rule = @(

@{ Match\_Name = "\*InternetExplorer\*"; IsMatch = "Yes"; Capability = ""; }

@{ Match\_Name = "\*MSPaint\*amd64\*"; IsMatch = "Yes"; Capability = "Microsoft.Windows.MSPaint~~~~0.0.1.0"; }

@{ Match\_Name = "\*MSPaint\*wow64\*"; IsMatch = "Yes"; Capability = "Microsoft.Windows.MSPaint~~~~0.0.1.0"; }

@{ Match\_Name = "\*Notepad\*amd64\*"; IsMatch = "Yes"; Capability = "Microsoft.Windows.Notepad~~~~0.0.1.0"; }

@{ Match\_Name = "\*Notepad\*wow64\*"; IsMatch = "Yes"; Capability = "Microsoft.Windows.Notepad~~~~0.0.1.0"; }

@{ Match\_Name = "\*MediaPlayer\*amd64\*"; IsMatch = "Yes"; Capability = "Media.WindowsMediaPlayer~~~~0.0.12.0"; }

@{ Match\_Name = "\*MediaPlayer\*wow64\*"; IsMatch = "Yes"; Capability = "Media.WindowsMediaPlayer~~~~0.0.12.0"; }

@{ Match\_Name = "\*PowerShell-ISE-FOD-Package\*amd64\*"; IsMatch = "Yes"; Capability = "Microsoft.Windows.PowerShell.ISE~~~~0.0.1.0"; }

@{ Match\_Name = "\*PowerShell-ISE-FOD-Package\*wow64\*"; IsMatch = "Yes"; Capability = "Microsoft.Windows.PowerShell.ISE~~~~0.0.1.0"; }

@{ Match\_Name = "\*Printing\*PMCPPC\*amd64\*"; IsMatch = "Yes"; Capability = "Print.Management.Console~~~~0.0.1.0"; }

@{ Match\_Name = "\*StepsRecorder\*amd64\*"; IsMatch = "Yes"; Capability = "App.StepsRecorder~~~~0.0.1.0"; }

@{ Match\_Name = "\*StepsRecorder\*wow64\*"; IsMatch = "Yes"; Capability = "App.StepsRecorder~~~~0.0.1.0"; }

@{ Match\_Name = "\*WordPad\*amd64\*"; IsMatch = "Yes"; Capability = "Microsoft.Windows.WordPad~~~~0.0.1.0"; }

@{ Match\_Name = "\*WordPad\*wow64\*"; IsMatch = "Yes"; Capability = "Microsoft.Windows.WordPad~~~~0.0.1.0"; }

@{ Match\_Name = "\*WMIC\*FoD\*Package\*amd64\*"; IsMatch = "Yes"; Capability = "WMIC~~~~"; }

@{ Match\_Name = "\*WMIC\*FoD\*Package\*wow64\*"; IsMatch = "Yes"; Capability = "WMIC~~~~"; }

)

}

)

ForEach ($item in $Script:Search\_File\_Order) { New-Variable -Name "Init\_File\_Type\_$($item.Name)" -Value @() -Force }

ForEach ($WildCard in $Script:Init\_Folder\_All\_File) {

ForEach ($item in $Script:Search\_File\_Order) {

ForEach ($NewRule in $item.Rule) {

if ($WildCard -like "\*$($NewRule.Match\_Name)\*") {

Write-host "`n Fuzzy matching: " -NoNewline; Write-host $NewRule.Match\_Name -ForegroundColor Green

Write-host " Language pack file: " -NoNewline; Write-host $WildCard -ForegroundColor Green

$OSDefaultUser = (Get-Variable -Name "Init\_File\_Type\_$($item.Name)" -ErrorAction SilentlyContinue).Value

$TempSave = @{ Match\_Name = $NewRule.Match\_Name; Capability = $NewRule.Capability; FileName = $WildCard }

$new = $OSDefaultUser + $TempSave

if ($NewRule.IsMatch -eq "Yes") {

ForEach ($Component in $Initl\_install\_Language\_Component) {

if ($Component -like "\*$($NewRule.Match\_Name)\*") {

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

}

}

} else {

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

}

}

}

}

}

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 " Language pack file: " -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:\OS\_11\_Custom\Install\Install\Mount" -Sources "D:\OS\_11\_Custom\Install\Install\Language\Add" -Lang "zh-CN"

###### Offline image language: change

* Starting Windows 11, the [default System UI Language](https://learn.microsoft.com/en-us/windows/win32/intl/user-interface-language-management) set by DISM is left unaltered on all editions except for Home edition. For [all commercial editions](https://www.microsoft.com/windowsforbusiness/compare) the language chosen during the Out-of-Box Experience (OOBE) is set as the [System Preferred UI language](https://learn.microsoft.com/en-us/windows/win32/intl/user-interface-language-management) 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.
  + 1. Change default language, regional settings, and other international settings

Region: zh-CN

Dism /Image:"D:\OS\_11\_Custom\Install\Install\Mount" /Set-AllIntl:zh-CN

* + 1. View available language settings

Dism /Image:"D:\OS\_11\_Custom\Install\Install\Mount" /Get-Intl

###### Language packs: Removed

* After you add languages, if you want to deploy to a non-English locale, you can save space by removing the English language component. When you remove a language, uninstall the language components in the reverse order in which you added them.
* After adding the Chinese, delete "English - United States" in reverse, the Region tag: en-US, you need to extract the language pack in advance
* Install.Del.Specified.lang.Tag.ps1
  + [\Expand\Install\Install.Del.Specified.lang.Tag.ps1](Expand/Install/Install.Del.Specified.lang.Tag.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/Install.Del.Specified.lang.Tag.ps1>
* Copy the code

$Lang = "en-US"

$Mount = "D:\OS\_11\_Custom\Install\Install\Mount"

$Sources = "D:\OS\_11\_Custom\Install\Install\Language\Del\en-US"

$Initl\_install\_Language\_Component = @()

Get-WindowsPackage -Path $Mount | ForEach-Object {

$Initl\_install\_Language\_Component += $\_.PackageName

}

$Language = @(

@{ Match = "\*WordPad\*wow64\*"; File = "Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~$($Lang)~.cab"; }

@{ Match = "\*WordPad\*amd64\*"; File = "Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~$($Lang)~.cab"; }

@{ Match = "\*WMIC\*FoD\*Package\*wow64\*"; File = "Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~$($Lang)~.cab"; }

@{ Match = "\*WMIC\*FoD\*Package\*amd64\*"; File = "Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~AMD64~$($Lang)~.cab"; }

@{ Match = "\*StepsRecorder\*wow64\*"; File = "Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~$($Lang)~.cab"; }

@{ Match = "\*StepsRecorder\*amd64\*"; File = "Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~$($Lang)~.cab"; }

@{ Match = "\*Printing\*PMCPPC\*amd64\*"; File = "Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~AMD64~$($Lang)~.cab"; }

@{ Match = "\*PowerShell\*wow64\*"; File = "Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~$($Lang)~.cab"; }

@{ Match = "\*PowerShell\*amd64\*"; File = "Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~$($Lang)~.cab"; }

@{ Match = "\*MediaPlayer\*wow64\*"; File = "Microsoft-Windows-MediaPlayer-Package-wow64-$($Lang).cab"; }

@{ Match = "\*MediaPlayer\*amd64\*"; File = "Microsoft-Windows-MediaPlayer-Package-AMD64-$($Lang).cab"; }

@{ Match = "\*Notepad\*wow64\*"; File = "Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~$($Lang)~.cab"; }

@{ Match = "\*Notepad\*amd64\*"; File = "Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~AMD64~$($Lang)~.cab"; }

@{ Match = "\*InternetExplorer\*"; File = "Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~AMD64~$($Lang)~.cab"; }

@{ Match = "\*TextToSpeech\*"; File = "Microsoft-Windows-LanguageFeatures-TextToSpeech-$($Lang)-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*LanguageFeatures-Speech\*"; File = "Microsoft-Windows-LanguageFeatures-Speech-$($Lang)-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*OCR\*"; File = "Microsoft-Windows-LanguageFeatures-OCR-$($Lang)-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*Handwriting\*"; File = "Microsoft-Windows-LanguageFeatures-Handwriting-$($Lang)-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*LanguageFeatures-Basic\*"; File = "Microsoft-Windows-LanguageFeatures-Basic-$($Lang)-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*Client-LanguagePack-Package\*"; File = "Microsoft-Windows-Client-Language-Pack\_x64\_$($Lang).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 ($Component -like "\*$($Rule.Match)\*$($Lang)\*") {

Write-host " Component name: " -NoNewline

Write-host $Component -ForegroundColor Green

Write-host " Language pack file: " -NoNewline

Write-host "$($Sources)\$($Rule.File)" -ForegroundColor Green

Write-Host " Deleting ".PadRight(22) -NoNewline

try {

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

Write-host "Finish" -ForegroundColor Green

} catch {

Write-host "Failed" -ForegroundColor Red

Write-host " $($\_)" -ForegroundColor Red

}

break

}

}

}

$InitlClearLanguagePackage = @()

Get-WindowsPackage -Path $Mount | ForEach-Object {

if ($\_.PackageName -like "\*$($Lang)\*") {

$InitlClearLanguagePackage += $\_.PackageName

}

}

if ($InitlClearLanguagePackage.count -gt 0) {

ForEach ($item in $InitlClearLanguagePackage) {

Write-Host "`n $($item)" -ForegroundColor Green

Write-Host " Deleting ".PadRight(22) -NoNewline

try {

Remove-AppxProvisionedPackage -Path $Mount -PackageName $item -ErrorAction SilentlyContinue | Out-Null

Write-host "Finish" -ForegroundColor Green

} catch {

Write-host "Failed" -ForegroundColor Red

Write-host " $($\_)" -ForegroundColor Red

}

}

}

###### Components: All packages installed in the image

* + 1. View

Get-WindowsPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" | Out-GridView

* + 1. Export to csv

$SaveTo = "D:\OS\_11\_Custom\Install\Install\Report.$(Get-Date -Format "yyyyMMddHHmmss").csv"

Get-WindowsPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" | Export-CSV -NoType -Path $SaveTo

Write-host $SaveTo -ForegroundColor Green

##### InBox Apps

###### Learn

* + 1. [DISM App Package (.appx or .appxbundle) Servicing Command-Line Options](https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/dism-app-package--appx-or-appxbundle--servicing-command-line-options)

###### InBox Apps: Pre-install the application

* Group abbreviations
* Group A: Education, Professional, ProfessionalEducation, ProfessionalWorkstation, Enterprise, IoTEnterprise, ServerRdsh
* Group B: EnterpriseN, EnterpriseGN, EnterpriseSN, ProfessionalN, EducationN, ProfessionalWorkstationN, ProfessionalEducationN
* Group C: Core, CoreN, CoreSingleLanguage

Name Products ID Grouping

1. Microsoft.HEIFImageExtension 9PMMSR1CGPWG A, C
2. Microsoft.HEVCVideoExtension 9NMZLZ57R3T7 A, C
3. Microsoft.VP9VideoExtensions 9N4D0MSMP0PT A, C
4. Microsoft.WebpImageExtension 9PG2DK419DRG A, C
5. Microsoft.GamingApp 9MV0B5HZVK9Z A, C
6. Microsoft.ZuneMusic 9WZDNCRFJ3PT A, C
7. MicrosoftCorporationII.MicrosoftFamily 9PDJDJS743XF C
8. Microsoft.WebMediaExtensions 9N5TDP8VCMHS A, C
9. Microsoft.WindowsSoundRecorder 9WZDNCRFHWKN A, C
10. Microsoft.Xbox.TCUI 9NKNC0LD5NN6 A, C
11. Microsoft.XboxGamingOverlay 9NZKPSTSNW4P A, C
12. Microsoft.ZuneVideo 9WZDNCRFJ3P2 A, C
13. Microsoft.RawImageExtension 9NCTDW2W1BH8 A, C
14. Microsoft.UI.Xaml.2.3 A, B, C
15. Microsoft.UI.Xaml.2.4 A, B, C
16. Microsoft.UI.Xaml.2.7 A, B, C
17. Microsoft.NET.Native.Framework.2.2 A, B, C
18. Microsoft.NET.Native.Runtime.2.2 A, B, C
19. Microsoft.VCLibs.140.00 A, B, C
20. Microsoft.VCLibs.140.00.UWPDesktop A, B, C
21. Microsoft.SecHealthUI 9NK0JLZT7JD0 A, B, C
22. Microsoft.WindowsStore 9WZDNCRFJBMP A, B, C
23. Microsoft.Sticky.Notes 9NBLGGH4QGHW A, B, C
24. Microsoft.Paint 9PCFS5B6T72H A, B, C
25. Microsoft.PowerAutomateDesktop 9NFTCH6J7FHV A, B, C
26. Microsoft.ScreenSketch 9MZ95KL8MR0L A, B, C
27. Microsoft.WindowsNotepad 9MSMLRH6LZF3 A, B, C
28. Microsoft.WindowsTerminal 9N0DX20HK701 A, B, C
29. Clipchamp.Clipchamp 9P1J8S7CCWWT A, B, C
30. Microsoft.Solitaire.Collection 9WZDNCRFHWD2 A, B, C
31. Microsoft.WindowsAlarms 9WZDNCRFJ3PR A, B, C
32. Microsoft.WindowsFeedbackHub 9NBLGGH4R32N A, B, C
33. Microsoft.WindowsMaps 9WZDNCRDTBVB A, B, C
34. Microsoft.BingNews 9WZDNCRFHVFW A, B, C
35. Microsoft.DesktopAppInstaller 9NBLGGH4NNS1 A, B, C
36. Microsoft.WindowsCamera 9WZDNCRFJBBG A, B, C
37. Microsoft.Getstarted 9WZDNCRDTBJJ A, B, C
38. Microsoft.Cortana 9NFFX4SZZ23L A, B, C
39. Microsoft.BingWeather 9WZDNCRFJ3Q2 A, B, C
40. Microsoft.GetHelp 9PKDZBMV1H3T A, B, C
41. Microsoft.MicrosoftOfficeHub 9WZDNCRD29V9 A, B, C
42. Microsoft.People 9NBLGGH10PG8 A, B, C
43. Microsoft.StorePurchaseApp 9NBLGGH4LS1F A, B, C
44. Microsoft.Todos 9NBLGGH5R558 A, B, C
45. Microsoft.Windows.Photos 9WZDNCRFJBH4 A, B, C
46. Microsoft.WindowsCalculator 9WZDNCRFHVN5 A, B, C
47. Microsoft.Windows.CommunicationsApps 9WZDNCRFHVQM A, B, C
48. Microsoft.XboxGameOverlay 9NBLGGH537C2 A, B, C
49. Microsoft.XboxIdentityProvider 9WZDNCRD1HKW A, B, C
50. Microsoft.XboxSpeechToTextOverlay 9P086NHDNB9W A, B, C
51. Microsoft.YourPhone 9NMPJ99VJBWV A, B, C
52. MicrosoftCorporationII.QuickAssist 9P7BP5VNWKX5 A, B, C
53. MicrosoftWindows.Client.WebExperience 9MSSGKG348SP A, B, C

###### InBox Apps: Installed

* + 1. View

Get-AppXProvisionedPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" | Out-GridView

* + 1. Export to Csv

$SaveTo = "D:\OS\_11\_Custom\Install\Install\Report.$(Get-Date -Format "yyyyMMddHHmmss").csv"

Get-AppXProvisionedPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" | Export-CSV -NoType -Path $SaveTo

Write-host $SaveTo -ForegroundColor Green

###### Remove all installed pre-applications

* Install.InBox.Appx.Clear.all.ps1
  + [\Expand\Install\Install.InBox.Appx.Clear.all.ps1](Expand/Install/Install.InBox.Appx.Clear.all.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/Install.InBox.Appx.Clear.all.ps1>
* Copy the code

Get-AppXProvisionedPackage -path "D:\OS\_11\_Custom\Install\Install\Mount" -ErrorAction SilentlyContinue | ForEach-Object {

Write-host "`n $($\_.DisplayName)"; Write-Host " Deleting ".PadRight(22) -NoNewline

try {

Remove-AppxProvisionedPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" -PackageName $\_.PackageName -ErrorAction SilentlyContinue | Out-Null

Write-host "Finish" -ForegroundColor Green

} catch {

Write-host "Failed" -ForegroundColor Red

}

}

###### Region tag: Understanding

Before understanding zone tags, first understand the list of languages available in the ZuneMusic app: en-us, en-gb, as-in, az-latn-az, bs-latn-ba, bn-in, cs-cz, cy-gb, ar-sa, de-de, af-za, am-et, da-dk, el-gr, es-es, es-mx, ca-es, ca-es-valencia, eu-es, fi-fi, bg-bg, ga-ie, et-ee, gl-es, gu-in, gd-gb, fil-ph, fr-ca, fr-fr, hy-am, fa-ir, hu-hu, id-id, it-it, ja-jp, hr-hr, ka-ge, is-is, he-il, hi-in, kk-kz, lo-la, km-kh, lt-lt, kn-in, kok-in, ko-kr, mk-mk, lv-lv, ms-my, mi-nz, mr-in, mt-mt, ne-np, nl-nl, nb-no, ml-in, nn-no, lb-lu, or-in, ro-ro, sk-sk, ru-ru, pl-pl, quz-pe, pt-br, pt-pt, sq-al, ta-in, th-th, sv-se, pa-in, sr-cyrl-ba, sr-cyrl-rs, sr-latn-rs, ur-pk, ug-cn, tr-tr, uk-ua, tt-ru, zh-cn, vi-vn, uz-latn-uz, sl-si, zh-tw, te-in

View installed languages for offline images

Dism /Image:"D:\OS\_11\_Custom\Install\Install\Mount" /Get-Intl

View results for installed languages: en-US, zh-CN, When installing the ZuneMusic app, the installer will match "Installed Languages" and automatically filter the list of uninstalled languages, This is the area tag.

###### Region tag: adding method

* + 1. Execute "[Language Pack: Add](#_Language_pack:_add)"
    2. Install “Local Language Experience Packages (LXPs)”

Microsoft officially provides the Local Language Experience Package (LXPS) installation file for Windows 10. It will no longer be provided for Windows 11. Want to get:

* + - 1. Download using the Windows Local Language Experience Packs (LXPs) Downloader

learn: <https://github.com/ilikeyi/LXPs>

After downloading, save to: D:\OS\_11\_Custom\Install\Install\InBox.Appx

File format: LanguageExperiencePack.zh-CN.Neutral.Appx

* + - 1. Manual download
         1. Region

Region Products ID Region Products ID

en-US C9PDSCC711RVF af-za 9PDW16B5HMXR

am-et 9NGL4R61W3PL ar-sa 9N4S78P86PKX

as-in 9NTJLXMXX35J az-latn-az 9P5TFKZHQ5K8

be-by 9MXPBGNNDW3L bg-bg 9MX54588434F

bn-bd 9PH7TKVXGGM8 bn-in 9P1M44L7W84T

bs-latn-ba 9MVFKLJ10MFL ca-es 9P6JMKJQZ9S7

ca-es-valencia 9P9K3WMFSW90 chr-cher-us 9MX15485N3RK

cs-cz 9P3WXZ1KTM7C cy-gb 9NKJ9TBML4HB

da-dk 9NDMT2VKSNL1 de-de 9P6CT0SLW589

el-gr 9N586B13PBLD en-gb 9NT52VQ39BVN

es-es 9NWVGWLHPB1Z es-mx 9N8MCM1X3928

et-ee 9NFBHFMCR30L eu-es 9NMCHQHZ37HZ

fa-ir 9NGS7DD4QS21 fi-fi 9MW3PQ7SD3QK

fil-ph 9NWM2KGTDSSS fr-ca 9MTP2VP0VL92

fr-fr 9NHMG4BJKMDG ga-ie 9P0L5Q848KXT

gd-gb 9P1DBPF36BF3 gl-es 9NXRNBRNJN9B

gu-in 9P2HMSWDJDQ1 ha-latn-ng 9N1L95DBGRG3

he-il 9NB6ZFND5HCQ hi-in 9NZC3GRX8LD3

hr-hr 9NW01VND4LTW hu-hu 9MWN3C58HL87

hy-am 9NKM28TM6P67 id-id 9P4X3N4SDK8P

ig-ng 9PG4ZFJ48JSX is-is 9NTHJR7TQXX1

it-it 9P8PQWNS6VJX ja-jp 9N1W692FV4S1

ka-ge 9P60JZL05WGH kk-kz 9PHV179R97LV

km-kh 9PGKTS4JS531 kn-in 9NC6DB7N95F9

kok-in 9MV3P55CMZ6P ko-kr 9N4TXPCVRNGF

ku-arab-iq 9P1C18QL3D7H ky-kg 9P7D3JJGZM48

lb-lu 9N0ST1WBZ9D9 lo-la 9N8X352G5NZV

lt-lt 9NWWD891H6HN lv-lv 9N5CQDPH6SQT

mi-nz 9P2GDFB3JPSX mk-mk 9P1X6XB1K3RN

ml-in 9NWDTV8FFV7L mn-mn 9PG1DHC4VTZW

mr-in 9MWXCKHJVR1J ms-my 9NPXL8ZSDDQ7

mt-mt 9PDG96SQ6BN8 nb-no 9N6J0M5DHCK0

ne-np 9P7CHPLWDQVN nl-nl 9PF1C9NB5PRV

nn-no 9PK7KM3Z06KH nso-za 9NS49QLX5CDV

or-in 9NTHCXCXSJDH pa-arab-pk 9NJRL03WH6FM

pa-in 9NSNC0ZJX69B pl-pl 9NC5HW94R0LD

prs-af 9P3NGC6X5ZQC pt-br 9P8LBDM4FW35

pt-pt 9P7X8QJ7FL0X quc-latn-gt 9P2V6MNNQZ0B

quz-pe 9NHTX8NVQ04K ro-ro 9MWXGPJ5PJ3H

ru-ru 9NMJCX77QKPX rw-rw 9NFW0M20H9WG

sd-arab-pk 9NB9JSCXW9X5 si-lk 9NVF9QSLGTL0

sk-sk 9N7LSNN099WB sl-si 9NV27L34J4ST

sq-al 9MWLRGNMDGK7 sr-cyrl-ba 9MXGN7V65C7B

sr-cyrl-rs 9PPD6CCK9K5H sr-latn-rs 9NBZ0SJDPPVT

sv-se 9P0HSNX08177 sw-ke 9NFF2M19DQ55

ta-in 9PDZB1WT1B34 te-in 9PMQJJGF63FW

tg-cyrl-tj 9MZHLBPPT2HC th-th 9MSTWFRL0LR4

ti-et 9NC8C9RDNK2S tk-tm 9NKHQ4GL6VLT

tn-za 9NFSXM123DHT tr-tr 9NL1D3T5HG9R

tt-ru 9NV90Q1X1ZR2 ug-cn 9P52C5D7VL5S

uk-ua 9PPPMZRSGHR8 ur-pk 9NDWFTFW12BQ

uz-latn-uz 9P5P2T5P5L9S vi-vn 9P0W68X0XZPT

wo-sn 9NH3SW1CR90F xh-za 9NW3QWSLQD17

yo-ng 9NGM3VPPZS5V zh-cn 9NRMNT6GMZ7

zh-tw 9PCJ4DHCQ1JQ zu-za 9NNRM7KT5NB0

Download Region: zh-CN, application ID: 9NRMNT6GMZ70, Store link: https://www.microsoft.com/store/productId/9NRMNT6GMZ70

* + - * 1. Open the website: <https://store.rg-adguard.net>

Search keywords:

https://www.microsoft.com/store/productId/9NRMNT6GMZ70

Search 22621 content in the web page, search results: Microsoft.LanguageExperiencePackzh-CN\_22621.\*. neutral\_\_8wekyb3d8bbwe.appx

After downloading, save it to the

D:\OS\_11\_Custom\Install\Install\InBox.Appx directory and rename it: LanguageExperiencePack.zh-cn.Neutral.Appx

* + - 1. Execute the installation command to install the local language experience package (LXPs)

After understanding how to add zone tags, obtain LanguageExperiencePack.zh-cn.Neutral, execute the installation command:

Add-AppxProvisionedPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" -PackagePath "D:\OS\_11\_Custom\Install\Install\InBox.appx\LanguageExperiencePack.zh-cn.Neutral.appx" -SkipLicense

* + - 1. InBox Apps: An installed application package
         1. View

Get-AppXProvisionedPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" | Out-GridView

* + - * 1. Export to Csv

$SaveTo = "D:\OS\_11\_Custom\Install\Install\Report.$(Get-Date -Format "yyyyMMddHHmmss").csv"

Get-AppXProvisionedPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" | Export-CSV -NoType -Path $SaveTo

Write-host $SaveTo -ForegroundColor Green

###### InBox Apps: Install

* + 1. Mount or decompress the InBox Apps installation file

Mount 22621.1.220506-1250.ni\_release\_amd64fre\_InboxApps.iso or extract to any location;

* + 1. After executing the installation command, install InBox Apps to: Install.wim
* Auto = Automatically search all local disks, default;
* Custom path, e.g. specify F drive: $ISO = "F:\packages"
* Architecture: x64
* Install.Inst.InBox.Appx.ps1
  + [\Expand\Install\Install.Inst.InBox.Appx.ps1](Expand/Install/Install.Inst.InBox.Appx.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/Install.Inst.InBox.Appx.ps1>
* Copy the code

$ISO = "Auto"

$Mount = "D:\OS\_11\_Custom\Install\Install\Mount"

$Arch = "x64"

try {

Write-host "`n Offline image version: " -NoNewline

$Current\_Edition\_Version = (Get-WindowsEdition -Path $Mount).Edition

Write-Host $Current\_Edition\_Version -ForegroundColor Green

} catch {

Write-Host "Error" -ForegroundColor Red

Write-Host " $($\_)" -ForegroundColor Yellow

return

}

$Pre\_Config\_Rules = @{

Edition = @(

@{

Name = @( "CloudEdition"; )

Apps = @(

"Microsoft.UI.Xaml.2.3"; "Microsoft.UI.Xaml.2.4"; "Microsoft.UI.Xaml.2.7"; "Microsoft.NET.Native.Framework.2.2"; "Microsoft.NET.Native.Runtime.2.2"; "Microsoft.VCLibs.140.00"; "Microsoft.VCLibs.140.00.UWPDesktop"; "Microsoft.Services.Store.Engagement"; "Microsoft.VP9VideoExtensions"; "Clipchamp.Clipchamp"; "Microsoft.BingNews"; "Microsoft.BingWeather"; "Microsoft.DesktopAppInstaller"; "Microsoft.GetHelp"; "Microsoft.Getstarted"; "Microsoft.HEIFImageExtension"; "Microsoft.HEVCVideoExtension"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.MicrosoftStickyNotes"; "Microsoft.MinecraftEducationEdition"; "Microsoft.Paint"; "Microsoft.RawImageExtension"; "Microsoft.ScreenSketch"; "Microsoft.SecHealthUI"; "Microsoft.StorePurchaseApp"; "Microsoft.Todos"; "Microsoft.WebMediaExtensions"; "Microsoft.WebpImageExtension"; "Microsoft.Whiteboard"; "Microsoft.Windows.Photos"; "Microsoft.WindowsAlarms"; "Microsoft.WindowsCalculator"; "Microsoft.WindowsCamera"; "Microsoft.WindowsFeedbackHub"; "Microsoft.WindowsMaps"; "Microsoft.WindowsNotepad"; "Microsoft.WindowsSoundRecorder"; "Microsoft.Xbox.TCUI"; "Microsoft.XboxIdentityProvider"; "Microsoft.XboxSpeechToTextOverlay"; "Microsoft.ZuneMusic"; "Microsoft.ZuneVideo"; "MicrosoftCorporationII.QuickAssist";

)

}

@{

Name = @( "CloudEditionN"; )

Apps = @(

"Microsoft.UI.Xaml.2.3"; "Microsoft.UI.Xaml.2.4"; "Microsoft.UI.Xaml.2.7"; "Microsoft.NET.Native.Framework.2.2"; "Microsoft.NET.Native.Runtime.2.2"; "Microsoft.VCLibs.140.00"; "Microsoft.VCLibs.140.00.UWPDesktop"; "Microsoft.Services.Store.Engagement"; "Microsoft.XboxSpeechToTextOverlay"; "Clipchamp.Clipchamp"; "Microsoft.BingNews"; "Microsoft.BingWeather"; "Microsoft.DesktopAppInstaller"; "Microsoft.GetHelp"; "Microsoft.Getstarted"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.MicrosoftStickyNotes"; "Microsoft.MinecraftEducationEdition"; "Microsoft.Paint"; "Microsoft.ScreenSketch"; "Microsoft.SecHealthUI"; "Microsoft.StorePurchaseApp"; "Microsoft.Whiteboard"; "Microsoft.Windows.Photos"; "Microsoft.WindowsAlarms"; "Microsoft.WindowsCalculator"; "Microsoft.WindowsCamera"; "Microsoft.WindowsFeedbackHub"; "Microsoft.WindowsMaps"; "Microsoft.WindowsNotepad"; "Microsoft.XboxIdentityProvider"; "MicrosoftCorporationII.QuickAssist";

)

}

@{

Name = @(

"Core"; "CoreN"; "CoreSingleLanguage";

)

Apps = @(

"Microsoft.UI.Xaml.2.3"; "Microsoft.UI.Xaml.2.4"; "Microsoft.UI.Xaml.2.7"; "Microsoft.NET.Native.Framework.2.2"; "Microsoft.NET.Native.Runtime.2.2"; "Microsoft.VCLibs.140.00"; "Microsoft.VCLibs.140.00.UWPDesktop"; "Microsoft.HEIFImageExtension"; "Microsoft.HEVCVideoExtension"; "Microsoft.SecHealthUI"; "Microsoft.VP9VideoExtensions"; "Microsoft.WebpImageExtension"; "Microsoft.WindowsStore"; "Microsoft.GamingApp"; "Microsoft.MicrosoftStickyNotes"; "Microsoft.Paint"; "Microsoft.PowerAutomateDesktop"; "Microsoft.ScreenSketch"; "Microsoft.WindowsNotepad"; "Microsoft.WindowsTerminal"; "Clipchamp.Clipchamp"; "Microsoft.MicrosoftSolitaireCollection"; "Microsoft.WindowsAlarms"; "Microsoft.WindowsFeedbackHub"; "Microsoft.WindowsMaps"; "Microsoft.ZuneMusic"; "Microsoft.BingNews"; "Microsoft.BingWeather"; "Microsoft.DesktopAppInstaller"; "Microsoft.WindowsCamera"; "Microsoft.Getstarted"; "Microsoft.Cortana"; "Microsoft.GetHelp"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.People"; "Microsoft.StorePurchaseApp"; "Microsoft.Todos"; "Microsoft.WebMediaExtensions"; "Microsoft.Windows.Photos"; "Microsoft.WindowsCalculator"; "Microsoft.windowscommunicationsapps"; "Microsoft.WindowsSoundRecorder"; "Microsoft.Xbox.TCUI"; "Microsoft.XboxGameOverlay"; "Microsoft.XboxGamingOverlay"; "Microsoft.XboxIdentityProvider"; "Microsoft.XboxSpeechToTextOverlay"; "Microsoft.YourPhone"; "Microsoft.ZuneVideo"; "MicrosoftCorporationII.QuickAssist"; "MicrosoftWindows.Client.WebExperience"; "Microsoft.RawImageExtension"; "MicrosoftCorporationII.MicrosoftFamily";

)

}

@{

Name = @(

"Education"; "Professional"; "ProfessionalEducation"; "ProfessionalWorkstation"; "Enterprise"; "IoTEnterprise"; "ServerRdsh";

)

Apps = @(

"Microsoft.UI.Xaml.2.3"; "Microsoft.UI.Xaml.2.4"; "Microsoft.UI.Xaml.2.7"; "Microsoft.NET.Native.Framework.2.2"; "Microsoft.NET.Native.Runtime.2.2"; "Microsoft.VCLibs.140.00"; "Microsoft.VCLibs.140.00.UWPDesktop"; "Microsoft.HEIFImageExtension"; "Microsoft.HEVCVideoExtension"; "Microsoft.SecHealthUI"; "Microsoft.VP9VideoExtensions"; "Microsoft.WebpImageExtension"; "Microsoft.WindowsStore"; "Microsoft.GamingApp"; "Microsoft.MicrosoftStickyNotes"; "Microsoft.Paint"; "Microsoft.PowerAutomateDesktop"; "Microsoft.ScreenSketch"; "Microsoft.WindowsNotepad"; "Microsoft.WindowsTerminal"; "Clipchamp.Clipchamp"; "Microsoft.MicrosoftSolitaireCollection"; "Microsoft.WindowsAlarms"; "Microsoft.WindowsFeedbackHub"; "Microsoft.WindowsMaps"; "Microsoft.ZuneMusic"; "Microsoft.BingNews"; "Microsoft.BingWeather"; "Microsoft.DesktopAppInstaller"; "Microsoft.WindowsCamera"; "Microsoft.Getstarted"; "Microsoft.Cortana"; "Microsoft.GetHelp"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.People"; "Microsoft.StorePurchaseApp"; "Microsoft.Todos"; "Microsoft.WebMediaExtensions"; "Microsoft.Windows.Photos"; "Microsoft.WindowsCalculator"; "Microsoft.windowscommunicationsapps"; "Microsoft.WindowsSoundRecorder"; "Microsoft.Xbox.TCUI"; "Microsoft.XboxGameOverlay"; "Microsoft.XboxGamingOverlay"; "Microsoft.XboxIdentityProvider"; "Microsoft.XboxSpeechToTextOverlay"; "Microsoft.YourPhone"; "Microsoft.ZuneVideo"; "MicrosoftCorporationII.QuickAssist"; "MicrosoftWindows.Client.WebExperience"; "Microsoft.RawImageExtension";

)

}

@{

Name = @(

"EnterpriseN"; "EnterpriseGN"; "EnterpriseSN"; "ProfessionalN"; "EducationN"; "ProfessionalWorkstationN"; "ProfessionalEducationN"; "CloudN"; "CloudEN"; "CloudEditionLN"; "StarterN";

)

Apps = @(

"Microsoft.UI.Xaml.2.3"; "Microsoft.UI.Xaml.2.4"; "Microsoft.UI.Xaml.2.7"; "Microsoft.NET.Native.Framework.2.2"; "Microsoft.NET.Native.Runtime.2.2"; "Microsoft.VCLibs.140.00"; "Microsoft.VCLibs.140.00.UWPDesktop"; "Microsoft.SecHealthUI"; "Microsoft.WindowsStore"; "Microsoft.MicrosoftStickyNotes"; "Microsoft.Paint"; "Microsoft.PowerAutomateDesktop"; "Microsoft.ScreenSketch"; "Microsoft.WindowsNotepad"; "Microsoft.WindowsTerminal"; "Clipchamp.Clipchamp"; "Microsoft.MicrosoftSolitaireCollection"; "Microsoft.WindowsAlarms"; "Microsoft.WindowsFeedbackHub"; "Microsoft.WindowsMaps"; "Microsoft.BingNews"; "Microsoft.BingWeather"; "Microsoft.DesktopAppInstaller"; "Microsoft.WindowsCamera"; "Microsoft.Getstarted"; "Microsoft.Cortana"; "Microsoft.GetHelp"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.People"; "Microsoft.StorePurchaseApp"; "Microsoft.Todos"; "Microsoft.Windows.Photos"; "Microsoft.WindowsCalculator"; "Microsoft.windowscommunicationsapps"; "Microsoft.XboxGameOverlay"; "Microsoft.XboxIdentityProvider"; "Microsoft.XboxSpeechToTextOverlay"; "Microsoft.YourPhone"; "MicrosoftCorporationII.QuickAssist"; "MicrosoftWindows.Client.WebExperience";

)

}

)

Rule = @(

@{ Name="Microsoft.UI.Xaml.2.3"; Match="UI.Xaml\*{ARCHTag}\*2.3";License="UI.Xaml\*{ARCHTag}\*2.3"; Dependencies=@(); }

@{ Name="Microsoft.UI.Xaml.2.4"; Match="UI.Xaml\*{ARCHTag}\*2.4";License="UI.Xaml\*{ARCHTag}\*2.4"; Dependencies=@(); }

@{ Name="Microsoft.UI.Xaml.2.7"; Match="UI.Xaml\*{ARCHTag}\*2.7";License="UI.Xaml\*{ARCHTag}\*2.7"; Dependencies=@(); }

@{ Name="Microsoft.NET.Native.Framework.2.2"; Match="Native.Framework\*{ARCHTag}\*2.2";License="Native.Framework\*{ARCHTag}\*2.2"; Dependencies=@(); }

@{ Name="Microsoft.NET.Native.Runtime.2.2"; Match="Native.Runtime\*{ARCHTag}\*2.2";License="Native.Runtime\*{ARCHTag}\*2.2"; Dependencies=@(); }

@{ Name="Microsoft.VCLibs.140.00"; Match="VCLibs\*{ARCHTag}";License="VCLibs\*{ARCHTag}"; Dependencies=@(); }

@{ Name="Microsoft.VCLibs.140.00.UWPDesktop"; Match="VCLibs\*{ARCHTag}\*Desktop";License="VCLibs\*{ARCHTag}\*Desktop"; Dependencies=@(); }

@{ Name="Microsoft.HEIFImageExtension"; Match="HEIFImageExtension";License="HEIFImageExtension\*"; Dependencies=@("Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.HEVCVideoExtension"; Match="HEVCVideoExtension\*{ARCHC}";License="HEVCVideoExtension\*{ARCHC}\*xml"; Dependencies=@("Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.SecHealthUI"; Match="SecHealthUI\*{ARCHC}";License="SecHealthUI\*{ARCHC}"; Dependencies=@("Microsoft.UI.Xaml.2.4","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.VP9VideoExtensions"; Match="VP9VideoExtensions\*{ARCHC}";License="VP9VideoExtensions\*{ARCHC}"; Dependencies=@("Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.WebpImageExtension"; Match="WebpImageExtension\*{ARCHC}";License="WebpImageExtension\*{ARCHC}"; Dependencies=@("Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.WindowsStore"; Match="WindowsStore";License="WindowsStore"; Dependencies=@("Microsoft.UI.Xaml.2.3","Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.GamingApp"; Match="GamingApp";License="GamingApp"; Dependencies=@("Microsoft.UI.Xaml.2.3","Microsoft.VCLibs.140.00","Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name="Microsoft.MicrosoftStickyNotes"; Match="Microsoft.Sticky.Notes";License="MicrosoftStickyNotes"; Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.Paint"; Match="Paint";License="Paint"; Dependencies=@("Microsoft.VCLibs.140.00","Microsoft.VCLibs.140.00.UWPDesktop","Microsoft.UI.Xaml.2.7"); }

@{ Name="Microsoft.PowerAutomateDesktop"; Match="PowerAutomateDesktop";License="PowerAutomateDesktop"; Dependencies=@("Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name="Microsoft.ScreenSketch"; Match="ScreenSketch";License="ScreenSketch"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.WindowsNotepad"; Match="WindowsNotepad";License="WindowsNotepad"; Dependencies=@("Microsoft.VCLibs.140.00","Microsoft.VCLibs.140.00.UWPDesktop","Microsoft.UI.Xaml.2.7"); }

@{ Name="Microsoft.WindowsTerminal"; Match="WindowsTerminal";License="WindowsTerminal"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name="Clipchamp.Clipchamp"; Match="Clipchamp.Clipchamp";License="Clipchamp.Clipchamp"; Dependencies=@(); }

@{ Name="Microsoft.MicrosoftSolitaireCollection"; Match="MicrosoftSolitaireCollection";License="MicrosoftSolitaireCollection"; Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.WindowsAlarms"; Match="WindowsAlarms";License="WindowsAlarms"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.WindowsFeedbackHub"; Match="WindowsFeedbackHub";License="WindowsFeedbackHub"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.WindowsMaps"; Match="WindowsMaps";License="WindowsMaps"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.ZuneMusic"; Match="ZuneMusic";License="ZuneMusic"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.VCLibs.140.00"); }

@{ Name="MicrosoftCorporationII.MicrosoftFamily"; Match="MicrosoftFamily";License="MicrosoftFamily"; Dependencies=@(); }

@{ Name="Microsoft.BingNews"; Match="BingNews";License="BingNews"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.BingWeather"; Match="BingWeather";License="BingWeather"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.DesktopAppInstaller"; Match="DesktopAppInstaller";License="DesktopAppInstaller"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name="Microsoft.WindowsCamera"; Match="WindowsCamera";License="WindowsCamera"; Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.Getstarted"; Match="Getstarted";License="Getstarted"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.Cortana"; Match="Cortana";License="Cortana"; Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00","Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name="Microsoft.GetHelp"; Match="GetHelp";License="GetHelp"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.MicrosoftOfficeHub"; Match="MicrosoftOfficeHub";License="MicrosoftOfficeHub"; Dependencies=@("Microsoft.VCLibs.140.00","Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name="Microsoft.People"; Match="People";License="People"; Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.StorePurchaseApp"; Match="StorePurchaseApp";License="StorePurchaseApp"; Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.Todos"; Match="Todos";License="Todos"; Dependencies=@("Microsoft.UI.Xaml.2.4","Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.WebMediaExtensions"; Match="WebMediaExtensions";License="WebMediaExtensions"; Dependencies=@("Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.Windows.Photos"; Match="Windows.Photos";License="Windows.Photos"; Dependencies=@("Microsoft.UI.Xaml.2.4","Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.WindowsCalculator"; Match="WindowsCalculator";License="WindowsCalculator"; Dependencies=@("Microsoft.UI.Xaml.2.4","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.windowscommunicationsapps"; Match="WindowsCommunicationsApps";License="WindowsCommunicationsApps"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.WindowsSoundRecorder"; Match="WindowsSoundRecorder";License="WindowsSoundRecorder"; Dependencies=@("Microsoft.UI.Xaml.2.3","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.Xbox.TCUI"; Match="Xbox.TCUI";License="Xbox.TCUI"; Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.XboxGameOverlay"; Match="XboxGameOverlay";License="XboxGameOverlay"; Dependencies=@("Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.XboxGamingOverlay"; Match="XboxGamingOverlay";License="XboxGamingOverlay"; Dependencies=@("Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.XboxIdentityProvider"; Match="XboxIdentityProvider";License="XboxIdentityProvider"; Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.XboxSpeechToTextOverlay"; Match="XboxSpeechToTextOverlay";License="XboxSpeechToTextOverlay"; Dependencies=@("Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.YourPhone"; Match="YourPhone";License="YourPhone"; Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00","Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name="Microsoft.ZuneVideo"; Match="ZuneVideo";License="ZuneVideo"; Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.VCLibs.140.00"); }

@{ Name="MicrosoftCorporationII.QuickAssist"; Match="QuickAssist";License="QuickAssist"; Dependencies=@(); }

@{ Name="MicrosoftWindows.Client.WebExperience"; Match="WebExperience";License="WebExperience"; Dependencies=@("Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.MinecraftEducationEdition"; Match="MinecraftEducationEdition";License="MinecraftEducationEdition"; Dependencies=@("Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name="Microsoft.Whiteboard"; Match="Whiteboard";License="Whiteboard"; Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.RawImageExtension"; Match="RawImageExtension"; License="RawImageExtension"; Dependencies=@(); }

)

}

$Allow\_Install\_App = @()

ForEach ($item in $Pre\_Config\_Rules.Edition) {

if ($item.Name -contains $Current\_Edition\_Version) {

Write-host "`n Match to: "-NoNewline; Write-host $Current\_Edition\_Version -ForegroundColor Green

$Allow\_Install\_App = $item.Apps

break

}

}

Write-host "`n The app to install ( $($Allow\_Install\_App.Count) item )" -ForegroundColor Yellow

Write-host " $('-' \* 80)"

ForEach ($item in $Allow\_Install\_App) {

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

}

Function Match\_InBox\_Apps\_Install\_Pack

{

param ( $NewPath )

$NewArch = $Arch

$NewArchC = $Arch.Replace("AMD64", "x64")

$NewArchCTag = $Arch.Replace("AMD64", "x64")

if ($Arch -eq "arm64") { $NewArchCTag = "arm" }

if ($Pre\_Config\_Rules.Rule.Count -gt 0) {

ForEach ($itemInBoxApps in $Pre\_Config\_Rules.Rule){

$InstallPacker = ""

$InstallPackerCert = ""

$SearchNewStructure = $itemInBoxApps.Match.Replace("{ARCH}", $NewArch).Replace("{ARCHC}", $NewArchC).Replace("{ARCHTag}", $NewArchCTag)

$SearchNewLicense = $itemInBoxApps.License.Replace("{ARCH}", $NewArch).Replace("{ARCHC}", $NewArchC).Replace("{ARCHTag}", $NewArchCTag)

Get-ChildItem -Path $NewPath -Filter "\*$($SearchNewStructure)\*" -Include "\*.appx", "\*.appxbundle", "\*.msixbundle" -Recurse -Force -ErrorAction SilentlyContinue | ForEach-Object {

if (Test-Path -Path $\_.FullName -PathType Leaf) {

$InstallPacker = $\_.FullName

Get-ChildItem -Path $NewPath -Filter "\*$($SearchNewLicense)\*" -Include \*.xml -Recurse -Force -ErrorAction SilentlyContinue | ForEach-Object {

$InstallPackerCert = $\_.FullName

}

$Script:InBoxAppx += @{

Name = $itemInBoxApps.Name;

Depend = $itemInBoxApps.Dependencies;

Search = $SearchNewStructure;

InstallPacker = $InstallPacker;

Certificate = $InstallPackerCert

CertificateRule = $SearchNewLicense

}

return

}

}

}

}

}

Write-host "`n InBox Apps: Installation packages, automatic search for full disk or specified paths" -ForegroundColor Yellow

Write-host " $('-' \* 80)"

$Script:InBoxAppx = @()

if ($ISO -eq "Auto") {

Get-PSDrive -PSProvider FileSystem -ErrorAction SilentlyContinue | ForEach-Object {

$AppPath = Join-Path -Path $\_.Root -ChildPath "packages" -ErrorAction SilentlyContinue

Match\_InBox\_Apps\_Install\_Pack -NewPath $AppPath

}

} else {

Match\_InBox\_Apps\_Install\_Pack -NewPath $ISO

}

Write-host " Search Complete" -ForegroundColor Green

Write-host "`n InBox Apps: Installer Match Results" -ForegroundColor Yellow

Write-host " $('-' \* 80)"

if ($Script:InBoxAppx.Count -gt 0) {

Write-host " Match successful" -ForegroundColor Green

} else {

Write-host " Failed match" -ForegroundColor Red

return

}

Write-host "`n InBox Apps: Details of the application to be installed ( $($Script:InBoxAppx.Count) item )" -ForegroundColor Yellow

Write-host " $('-' \* 80)"

ForEach ($Rule in $Script:InBoxAppx) {

Write-host " Apps name: " -NoNewline; Write-host $Rule.Name -ForegroundColor Yellow

Write-host " Apps installer: " -NoNewline; Write-host $Rule.InstallPacker -ForegroundColor Yellow

Write-host " License: " -NoNewline; Write-host $Rule.Certificate -ForegroundColor Yellow

Write-host ""

}

Write-host "`n InBox Apps: Installation" -ForegroundColor Yellow

Write-host " $('-' \* 80)"

ForEach ($Rule in $Script:InBoxAppx) {

Write-host " Name: " -NoNewline; Write-host $Rule.Name -ForegroundColor Yellow

Write-host " $('-' \* 80)"

if($Allow\_Install\_App -contains $Rule.Name) {

Write-host " Search for apps: " -NoNewline; Write-host $Rule.InstallPacker -ForegroundColor Yellow

Write-host " Search for License: " -NoNewline; Write-host $Rule.Certificate -ForegroundColor Yellow

if (Test-Path -Path $Rule.InstallPacker -PathType Leaf) {

if (Test-Path -Path $Rule.Certificate -PathType Leaf) {

Write-host " License: " -NoNewline

Write-host $Rule.Certificate -ForegroundColor Yellow

Write-host " With License".PadRight(22) -NoNewline -ForegroundColor Green

Write-host " Installing".PadRight(22) -NoNewline

try {

Add-AppxProvisionedPackage -Path $Mount -PackagePath $Rule.InstallPacker -LicensePath $Rule.Certificate -ErrorAction SilentlyContinue | Out-Null

Write-Host "Done`n" -ForegroundColor Green

} catch {

Write-Host "Failed" -ForegroundColor Red

Write-Host " $($\_)`n" -ForegroundColor Red

}

} else {

Write-host " No License".PadRight(22) -NoNewline -ForegroundColor Red

Write-host " Installing".PadRight(22) -NoNewline

try {

Add-AppxProvisionedPackage -Path $Mount -PackagePath $Rule.InstallPacker -SkipLicense -ErrorAction SilentlyContinue | Out-Null

Write-Host "Done`n" -ForegroundColor Green

} catch {

Write-Host "Failed" -ForegroundColor Red

Write-Host " $($\_)`n" -ForegroundColor Red

}

}

} else {

Write-host " The installation package does not exist" -ForegroundColor Red

}

} else {

Write-host " Skip the installation`n" -ForegroundColor Red

}

}

###### Region tag: delete

* After completing the installation of the InBox Apps application, the Region tag is no longer important and can be deleted or not deleted. Delete "Local Language Experience Package (LXPs), Simplified Chinese - China)" and uniquely identify the Region tag: zh-CN, can be changed to other Region tag.
* Install.Clear.Flag.ps1
  + [\Expand\Install\Install.Clear.Flag.ps1](Expand/Install/Install.Clear.Flag.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/Install.Clear.Flag.ps1>
* Copy the code

$Lang = "zh-CN"

Get-AppXProvisionedPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" | Foreach-object {

if ($\_.DisplayName -Like "\*LanguageExperiencePack\*$($Lang)\*") {

Write-host " $($\_.DisplayName)"; Write-Host " Deleting ".PadRight(22) -NoNewline

try {

Remove-AppxProvisionedPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" -PackageName $\_.PackageName -ErrorAction SilentlyContinue | Out-Null

Write-host "Finish" -ForegroundColor Green

} catch {

Write-host "Failed" -ForegroundColor Red

}

}

}

###### InBox Apps: optimization

After the app is installed, provisioning the Appx package should be optimized to reduce the app's disk usage by replacing identical files with hard links, only for offline images.

Dism /Image:"D:\OS\_11\_Custom\Install\Install\Mount" /Optimize-ProvisionedAppxPackages

##### Cumulative updates

* + - When upgrading different versions or old versions to the latest version, you need to add the "Function Enablement Package" first before adding the latest cumulative update;
    - After adding a language pack, you can install the same cumulative update as the initial version to resolve a known issue where the "Components: All packages installed in the image" status is not refreshed after installation;
    - To stay up to date, it is recommended that you download the latest version.

###### Initial version

* + 1. How to download

Cumulative update KB5016632 is no longer searchable from <https://www.catalog.update.microsoft.com/Search.aspx?q=KB5016632>

* + - 1. Select any website and open:
         1. <https://uupdump.net>
         2. <https://uup.ee>
         3. <https://osdump.com>
      2. After opening, search for keywords: 22621.382, select from the search results: Windows 11, version 22H2 (22621.382) amd64 and Windows 11, version 22H2 (22621.382) arm64
      3. After opening, select "All files";
      4. Search for: KB5016632 in the "All Files" page, download:
* Windows11.0-KB5016632-x64.cab
* Windows11.0-KB5016632-x64.psf
  + - 1. Cumulative updates: Merging
         1. Go to <https://github.com/abbodi1406/WHD/tree/master/scripts> After downloading [PSFX\_Repack\_6.zip](https://github.com/abbodi1406/WHD/raw/master/scripts/PSFX_Repack_6.zip)
         2. After downloading, unzip it to: D:\PSFX\_Repack\_6
         3. Copy Windows11.0-KB5016632-x64.cab, Windows11.0-KB5016632-x64.psf to: D:\PSFX\_Repack\_6
         4. After running: psfx2cab\_GUI.cmd, you will get a new file:

Windows11.0-KB5016632-x64-full\_psfx.cab

* + - * 1. Save Windows11.0-KB5016632-x64-full\_psfx.cab to:

D:\OS\_11\_Custom\Install\Install\Update\Windows11.0-KB5016632-x64-full\_psfx.cab

* + 1. Add

$KBPath = "D:\OS\_11\_Custom\Install\Install\Update\Windows11.0-KB5016632-x64-full\_psfx.cab"

Add-WindowsPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" -PackagePath $KBPath

###### Other Version

Check "[Windows 11 version information](https://learn.microsoft.com/en-us/windows/release-health/windows11-release-information)", for example, download cumulative update: KB5035853, Version: 22621.3296, go to the download page: <https://www.catalog.update.microsoft.com/Search.aspx?q=KB5035853>, download and save to: D:\OS\_11\_Custom\Install\Install\Update, or download through direct connection, select download according to the architecture:

* + 1. x64, default
* Direct download

<https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/594b22d5-84c3-4665-bdc7-3167c91759b9/public/windows11.0-kb5035853-x64_8ca1a9a646dbe25c071a8057f249633a61929efa.msu>

* Add to

$KBPath = "D:\OS\_11\_Custom\Install\Install\Update\windows11.0-kb5035853-x64\_8ca1a9a646dbe25c071a8057f249633a61929efa.msu"

Add-WindowsPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" -PackagePath $KBPath

* + 1. Arm64
* Direct download

<https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/72ba502a-7982-471c-aad2-355efddbff61/public/windows11.0-kb5035853-arm64_076c009bca80cfa85272bd50d65951c2f663fcc5.msu>

* Add to

$KBPath = "D:\OS\_11\_Custom\Install\Install\Update\windows11.0-kb5035853-arm64\_076c009bca80cfa85272bd50d65951c2f663fcc5.msu"

Add-WindowsPackage -Path "D:\OS\_11\_Custom\Install\Install\Mount" -PackagePath $KBPath

###### Solidify Updated

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

Dism /Image:"D:\OS\_11\_Custom\Install\Install\Mount" /cleanup-image /StartComponentCleanup /ResetBase

* + 1. Clean components after curing and updating

$Mount = "D:\OS\_11\_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

}

}

##### Drive

##### 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.

##### Health

Before saving, check whether it is damaged. If the health status is abnormal, stop saving.

Repair-WindowsImage -Path "D:\OS\_11\_Custom\Install\Install\Mount" -ScanHealth

##### Replace WinRE.wim

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

$WinRE = "D:\OS\_11\_Custom\Install\Install\Update\Winlib\WinRE.wim"

$CopyTo = "D:\OS\_11\_Custom\Install\Install\Mount\Windows\System32\Recovery"

Copy-Item -Path $WinRE -Destination $CopyTo -Force

##### Save image: Install.wim

Save-WindowsImage -Path "D:\OS\_11\_Custom\Install\Install\Mount"

##### Unmount image: Install.wim

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

Dismount-WindowsImage -Path "D:\OS\_11\_Custom\Install\Install\Mount" -Discard

LOOP OPERATING AREA, END.

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

###### Obtain WimLib

After going to the official website of <https://wimlib.net>, select a different version: [arm64](https://wimlib.net/downloads/wimlib-1.14.3-windows-aarch64-bin.zip), [x64](https://wimlib.net/downloads/wimlib-1.14.3-windows-x86_64-bin.zip), [x86](https://wimlib.net/downloads/wimlib-1.14.3-windows-i686-bin.zip), and extract it to: D:Wimlib after downloading

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

* + 1. Extract the WinRE.wim file Install.wim from Install.wim
* Install.WinRE.Extract.ps1
  + [\Expand\Install\Install.WinRE.Extract.ps1](Expand/Install/Install.WinRE.Extract.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/Install.WinRE.Extract.ps1>
* Copy the code

$Arguments = @(

"extract",

"D:\OS\_11\sources\install.wim", "1",

"\Windows\System32\Recovery\Winre.wim",

"--dest-dir=""D:\OS\_11\_Custom\Install\Install\Update\Winlib"""

)

New-Item -Path "D:\OS\_11\_Custom\Install\Install\Update\Winlib" -ItemType Directory -ea SilentlyContinue

Start-Process -FilePath "d:\wimlib\wimlib-imagex.exe" -ArgumentList $Arguments -wait -nonewwindow

* + 1. 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](Expand/Install/Install.WinRE.Replace.wim.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/Install.WinRE.Replace.wim.ps1>
* Copy the code

Get-WindowsImage -ImagePath "D:\OS\_11\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:\OS\_11\sources\install.wim", $\_.ImageIndex,

"--command=""add 'D:\OS\_11\_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

}

##### Rebuilding Install.wim reduces file size

* Install.Rebuild.wim.ps1
  + [\Expand\Install\Install.Rebuild.wim.ps1](Expand/Install/Install.Rebuild.wim.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/Install.Rebuild.wim.ps1>
* Copy the code

$InstallWim = "D:\OS\_11\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

}

#### Custom encapsulation boot.wim

##### View Boot.wim details

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

$ViewFile = "D:\OS\_11\Sources\Boot.wim"

Get-WindowsImage -ImagePath $ViewFile | Foreach-Object { Get-WindowsImage -ImagePath $ViewFile -index $\_.ImageIndex }

##### Specify the path to mount Boot.wim

New-Item -Path "D:\OS\_11\_Custom\Boot\Boot\Mount" -ItemType directory -ea SilentlyContinue

##### Start mounting Boot.wim

Default index number: 2

Mount-WindowsImage -ImagePath "D:\OS\_11\sources\boot.wim" -Index "2" -Path "D:\OS\_11\_Custom\Boot\Boot\Mount"

##### Language pack: Boot

* 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, View the report "[Language installation package for Boot.wim](#_Report:_Language_installation_3)".
* When adding languages, different schema versions must be corresponded, and if not, errors are reported during the addition process.

###### Language pack: Add

* Boot.Instl.lang.ps1
  + [\Expand\Boot\Boot.Instl.lang.ps1](Expand/Boot/Boot.Instl.lang.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Boot/Boot.Instl.lang.ps1>
* Copy the code

$Mount = "D:\OS\_11\_Custom\Boot\Boot\Mount"

$Sources = "D:\OS\_11\_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\_List = @(

@{ Match = "\*WinPE\*Setup\*Client\*Package\*"; File = "WINPE-SETUP-CLIENT\_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\_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 pack 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

}

}

}

###### Offline image language: change

* + 1. Change default language, regional settings, and other international settings

Region: zh-CN

Dism /Image:"D:\OS\_11\_Custom\Boot\Boot\Mount" /Set-AllIntl:"zh-CN"

* + 1. View available language settings

Dism /Image:"D:\OS\_11\_Custom\Boot\Boot\Mount" /Get-Intl

###### Language pack: Removed

* After you add languages, if you want to deploy to a non-English locale, you can save space by removing the English language component. When you remove a language, uninstall the language components in the reverse order in which you added them.
* After adding the Chinese, delete "English - United States" in reverse, the Region tag: en-US, you need to extract the language pack in advance
* Boot.Del.Specified.lang.Tag.ps1
  + [\Expand\Boot\Boot.Del.Specified.lang.Tag.ps1](Expand/Boot/Boot.Del.Specified.lang.Tag.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Boot/Boot.Del.Specified.lang.Tag.ps1>
* Copy the code

$Lang = "en-US"

$Mount = "D:\OS\_11\_Custom\Boot\Boot\Mount"

$Sources = "D:\OS\_11\_Custom\Boot\Boot\Language\Del\en-US"

$Initl\_install\_Language\_Component = @()

Get-WindowsPackage -Path $Mount | ForEach-Object { $Initl\_install\_Language\_Component += $\_.PackageName }

$Language = @(

@{ Match = "\*WordPad\*wow64\*"; File = "Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~$($Lang)~.cab"; }

@{ Match = "\*WordPad\*amd64\*"; File = "Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~$($Lang)~.cab"; }

@{ Match = "\*WMIC\*FoD\*Package\*wow64\*"; File = "Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~$($Lang)~.cab"; }

@{ Match = "\*WMIC\*FoD\*Package\*amd64\*"; File = "Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~AMD64~$($Lang)~.cab"; }

@{ Match = "\*StepsRecorder\*wow64\*"; File = "Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~$($Lang)~.cab"; }

@{ Match = "\*StepsRecorder\*amd64\*"; File = "Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~$($Lang)~.cab"; }

@{ Match = "\*Printing\*PMCPPC\*amd64\*"; File = "Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~AMD64~$($Lang)~.cab"; }

@{ Match = "\*PowerShell\*wow64\*"; File = "Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~$($Lang)~.cab"; }

@{ Match = "\*PowerShell\*amd64\*"; File = "Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~$($Lang)~.cab"; }

@{ Match = "\*MediaPlayer\*wow64\*"; File = "Microsoft-Windows-MediaPlayer-Package-wow64-$($Lang).cab"; }

@{ Match = "\*MediaPlayer\*amd64\*"; File = "Microsoft-Windows-MediaPlayer-Package-AMD64-$($Lang).cab"; }

@{ Match = "\*Notepad\*wow64\*"; File = "Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~$($Lang)~.cab"; }

@{ Match = "\*Notepad\*amd64\*"; File = "Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~AMD64~$($Lang)~.cab"; }

@{ Match = "\*InternetExplorer\*"; File = "Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~AMD64~$($Lang)~.cab"; }

@{ Match = "\*TextToSpeech\*"; File = "Microsoft-Windows-LanguageFeatures-TextToSpeech-$($Lang)-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*LanguageFeatures-Speech\*"; File = "Microsoft-Windows-LanguageFeatures-Speech-$($Lang)-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*OCR\*"; File = "Microsoft-Windows-LanguageFeatures-OCR-$($Lang)-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*Handwriting\*"; File = "Microsoft-Windows-LanguageFeatures-Handwriting-$($Lang)-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*LanguageFeatures-Basic\*"; File = "Microsoft-Windows-LanguageFeatures-Basic-$($Lang)-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*Client-LanguagePack-Package\*"; File = "Microsoft-Windows-Client-Language-Pack\_x64\_$($Lang).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 ($Component -like "\*$($Rule.Match)\*$($Lang)\*") {

Write-host " Component name: " -NoNewline

Write-host $Component -ForegroundColor Green

Write-host " Language pack file: " -NoNewline

Write-host "$($Sources)\$($Rule.File)" -ForegroundColor Green

Write-Host " Deleting ".PadRight(22) -NoNewline

try {

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

Write-host "Finish" -ForegroundColor Green

} catch {

Write-host "Failed" -ForegroundColor Red

Write-host " $($\_)" -ForegroundColor Red

}

break

}

}

}

$InitlClearLanguagePackage = @()

Get-WindowsPackage -Path $Mount | ForEach-Object {

if ($\_.PackageName -like "\*$($Lang)\*") {

$InitlClearLanguagePackage += $\_.PackageName

}

}

if ($InitlClearLanguagePackage.count -gt 0) {

ForEach ($item in $InitlClearLanguagePackage) {

Write-Host "`n $($item)" -ForegroundColor Green

Write-Host " Deleting ".PadRight(22) -NoNewline

try {

Remove-AppxProvisionedPackage -Path $Mount -PackageName $item -ErrorAction SilentlyContinue | Out-Null

Write-host "Finish" -ForegroundColor Green

} catch {

Write-host "Failed" -ForegroundColor Red

Write-host " $($\_)" -ForegroundColor Red

}

}

}

###### Components: All packages installed in the image

* + 1. View

Get-WindowsPackage -Path "D:\OS\_11\_Custom\Boot\Boot\Mount" | Out-GridView

* + 1. Export to csv

$SaveTo = "D:\OS\_11\_Custom\Boot\Boot\Report.$(Get-Date -Format "yyyyMMddHHmmss").csv"

Get-WindowsPackage -Path "D:\OS\_11\_Custom\Boot\Boot\Mount" | Export-CSV -NoType -Path $SaveTo

Write-host $SaveTo -ForegroundColor Green

###### Language: Repair

* + 1. Extract

Open: D:\OS\_11\_Custom\Install\Install\Language\Add\zh-CN\Microsoft-Windows-Client-Language-Pack\_x64\_zh-CN.cab, enter the directory: Setup\sources\zh-cn\cli, and copy the following files to the deskto:

* + - 1. arunres.dll.mui
      2. spwizres.dll.mui
      3. w32uires.dll.mui
    1. Copy

Copy the extracted files to: D:\OS\_11\_Custom\Boot\Boot\Mount\sources\zh-CN

###### Language packs: sync to ISO installer

Copy-Item -Path "D:\OS\_11\_Custom\Boot\Boot\Mount\sources\zh-CN" -Destination "D:\OS\_11\sources\zh-CN" -Recurse -Force

###### 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.

* + 1. Regenerate the mounted directory lang.ini

Regenerated Lang.ini file location: D:\OS\_11\_Custom\Boot\Boot\Mount\Sources\lang.ini

Dism /image:"D:\OS\_11\_Custom\Boot\Boot\Mount" /gen-langini /distribution:"D:\OS\_11\_Custom\Boot\Boot\Mount"

* + 1. After regenerating lang.ini, sync to the installer

Regenerated Lang.ini file location: D:\OS\_11\Sources\lang.ini

Dism /image:"D:\OS\_11\_Custom\Boot\Boot\Mount" /gen-langini /distribution:"D:\OS\_11"

##### Cumulative updates

To prepare the cumulative updates file available, change the example file name: KB\_Boot.cab

###### Add

$KBPath = "D:\OS\_11\_Custom\Boot\Boot\Update\KB\_Boot.cab"

Add-WindowsPackage -Path "D:\OS\_11\_Custom\Boot\Boot\Mount" -PackagePath $KBPath

###### Delete

$KBPath = "D:\OS\_11\_Custom\Boot\Boot\Update\KB\_Boot.cab"

Remove-WindowsPackage -Path "D:\OS\_11\_Custom\Boot\Boot\Mount" -PackagePath $KBPath

###### Solid update

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

Dism /image:"D:\OS\_11\_Custom\Boot\Boot\Mount" /cleanup-image /StartComponentCleanup /ResetBase

* + 1. Clean components after curing and updating

$Mount = "D:\OS\_11\_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

}

}

##### Drive

##### Other

###### Bypass TPM check during installation

* Boot.Bypass.TPM.ps1
  + [\Expand\Boot\Boot.Bypass.TPM.ps1](Expand/Boot/Boot.Bypass.TPM.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Boot/Boot.Bypass.TPM.ps1>
* Copy the code

$RegSystem = "D:\OS\_11\_Custom\Boot\Boot\Mount\Windows\System32\Config\SYSTEM"

$RandomGuid = [guid]::NewGuid()

Write-Host " HKLM:\$($RandomGuid)"

New-PSDrive -PSProvider Registry -Name OtherTasksTPM -Root HKLM -ErrorAction SilentlyContinue | Out-Null

Start-Process reg -ArgumentList "Load ""HKLM\$($RandomGuid)"" ""$($RegSystem)""" -Wait -WindowStyle Hidden -ErrorAction SilentlyContinue

New-Item "HKLM:\$($RandomGuid)\Setup\LabConfig" -force -ea SilentlyContinue | Out-Null

New-ItemProperty -LiteralPath "HKLM:\$($RandomGuid)\Setup\LabConfig" -Name "BypassCPUCheck" -Value 1 -PropertyType DWord -Force -ea SilentlyContinue | Out-Null

New-ItemProperty -LiteralPath "HKLM:\$($RandomGuid)\Setup\LabConfig" -Name "BypassStorageCheck" -Value 1 -PropertyType DWord -Force -ea SilentlyContinue | Out-Null

New-ItemProperty -LiteralPath "HKLM:\$($RandomGuid)\Setup\LabConfig" -Name "BypassRAMCheck" -Value 1 -PropertyType DWord -Force -ea SilentlyContinue | Out-Null

New-ItemProperty -LiteralPath "HKLM:\$($RandomGuid)\Setup\LabConfig" -Name "BypassTPMCheck" -Value 1 -PropertyType DWord -Force -ea SilentlyContinue | Out-Null

New-ItemProperty -LiteralPath "HKLM:\$($RandomGuid)\Setup\LabConfig" -Name "BypassSecureBootCheck" -Value 1 -PropertyType DWord -Force -ea SilentlyContinue | Out-Null

[gc]::collect()

Start-Process reg -ArgumentList "unload ""HKLM\$($RandomGuid)""" -Wait -WindowStyle Hidden -ErrorAction SilentlyContinue

Remove-PSDrive -Name OtherTasksTPM

##### Save image: Boot.wim

Save-WindowsImage -Path "D:\OS\_11\_Custom\Boot\Boot\Mount"

##### Unmount image: Boot.wim

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

Dismount-WindowsImage -Path "D:\OS\_11\_Custom\Boot\Boot\Mount" -Discard

#### 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](https://learn.microsoft.com/en-us/windows-hardware/customize/desktop/unattend/)", Intervene in the installation process by leaving it unattended.

##### Add method

###### Add to ISO installation media

* + 1. Unattended
       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:\OS\_11\Autounattend.xml

Copy-Item "D:\Multilingual\\_Learn\Unattend\Mul.Unattend.xml" -Destination "D:\OS\_11\Autounattend.xml" -Force

* + - 1. 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:\OS\_11\Sources\Unattend.xml

Copy-Item "D:\Multilingual\\_Learn\Unattend\Mul.Unattend.xml" -Destination "D:\OS\_11\Sources\Unattend.xml" -Force

* + - 1. 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. Create $OEM$ path

New-Item -Path "D:\OS\_11\sources\`$OEM$\`$$\Panther" -ItemType Directory

* + - * 1. Copy

Copy D:\Multilingual\\_Learn\Unattend\Mul.Unattend.xml to D:\OS\_11\Sources\$OEM$\Panther\Unattend.xml

Copy-Item "D:\Multilingual\\_Learn\Unattend\Mul.Unattend.xml" -Destination "D:\OS\_11\sources\`$OEM$\`$$\Panther\Unattend.xml" -Force

* + 1. Deployment engine: add

Add "Automatically add installed languages for Windows systems" to D:\OS\_11\sources\$OEM$\$1\Yi\Engine in the directory.

* + - 1. Deployment Engine: Copy

Copy D:\Multilingual\Engine to D:\OS\_11\Sources\$OEM$\$1\Yi\Engine

Copy-Item "D:\Multilingual\Engine" -Destination "D:\OS\_11\sources\`$OEM$\`$1\Yi\Engine" -Recurse -Force

* + - 1. 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:\OS\_11\sources\`$OEM$\`$1\Yi\Engine\Deploy\Allow" -ItemType Directory -ErrorAction SilentlyContinue | Out-Null

Out-File -FilePath "D:\OS\_11\sources\`$OEM$\`$1\Yi\Engine\Deploy\Allow\$($item)" -Encoding utf8 -ErrorAction SilentlyContinue

}

###### Add to mounted

Through "[Custom encapsulation: Install.wim](#_Custom_encapsulation_Install.wim)", execute "[Start mounting Install.wim](#_Start_mounting_Install.wim)" and mount to: D:\OS\_11\_Custom\Install\Install\Mount

* + 1. Unattended

Copy D:\Multilingual\\_Learn\Unattend\Mul.Unattend.xml to D:\OS\_11\_Custom\Install\Install\Mount\Panther\Unattend.xml

Copy-Item "D:\Multilingual\\_Learn\Unattend\Mul.Unattend.xml" -Destination "D:\OS\_11\_Custom\Install\Install\Mount\Panther" -Force

* + 1. Deployment engine: add

Add "Automatically add languages installed on Windows systems" to the D:\OS\_11\_Custom\Install\Install\Mount\Yi\Engine directory.

* + - 1. Deployment Engine: Copy

Copy D:\Multilingual\Engine to D:\OS\_11\_Custom\Install\Install\Mount\Yi\Engine

Copy-Item "D:\Multilingual\Engine" -Destination "D:\OS\_11\_Custom\Install\Install\Mount\Yi\Engine" -Recurse -Force

* + - 1. 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:\OS\_11\sources\`$OEM$\`$1\Yi\Engine\Deploy\Allow" -ItemType Directory -ErrorAction SilentlyContinue | Out-Null

Out-File -FilePath "D:\OS\_11\sources\`$OEM$\`$1\Yi\Engine\Deploy\Allow\$($item)" -Encoding utf8 -ErrorAction SilentlyContinue

}

##### Deployment Engine: Advanced

###### Deployment engine: adding process

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

###### Unattended solution

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

* D:\OS\_11\Autounattend.xml
* D:\OS\_11\Sources\Unattend.xml
* D:\OS\_11\sources\$OEM$\$$\Panther\unattend.xml
* D:\OS\_11\_Custom\Install\Install\Mount\Panther\Unattend.xml
  + 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.

* + - 1. Multi-language

<UILanguage>%OSDUILanguage%</UILanguage>

<InputLocale>%OSDInputLocale%</InputLocale>

<SystemLocale>%OSDSystemLocale%</SystemLocale>

<UILanguage>%OSDUILanguage%</UILanguage>

<UILanguageFallback>%OSDUILanguageFallback%</UILanguageFallback>

<UserLocale>%OSDUserLocale%</UserLocale>

* + - 1. Monolingual

A single language needs to specify a Region tag, for example, specify a Region 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>

* + 1. 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.

* + - 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>

* + - 1. Custom user

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

* + - * 1. Delete

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

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

* + - * 1. Replace

From the beginning <OOBE> to </OOBE>

<OOBE>

<ProtectYourPC>3</ProtectYourPC>

<HideEULAPage>true</HideEULAPage>

<HideWirelessSetupInOOBE>true</HideWirelessSetupInOOBE>

</OOBE>

### ISO

#### 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. x64

<https://github.com/ilikeyi/solutions/raw/main/_Software/Oscdimg/amd64/oscdimg.exe>

* 1. x86

<https://github.com/ilikeyi/solutions/raw/main/_Software/Oscdimg/x86/oscdimg.exe>

* 1. arm64

<https://github.com/ilikeyi/solutions/raw/main/_Software/Oscdimg/arm64/oscdimg.exe>

1. Use the oscdimg command line to generate an ISO file and save it to: D:\OS\_11.iso

* ISO.ps1
  + [\Expand\ISO.ps1](Expand/ISO.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/ISO.ps1>
* Copy the code

$Oscdimg = "D:\Oscdimg.exe"

$ISO = "D:\OS\_11"

$Volume = "OS\_11"

$SaveTo = "D:\OS\_11.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

#### Bypass TPM installation check

1. Learn about: <https://github.com/AveYo/MediaCreationTool.bat/tree/main/bypass11> and download: Quick\_11\_iso\_esd\_wim\_TPM\_toggle.bat
2. Drag the D:\OS\_11.iso file to Quick\_11\_iso\_esd\_wim\_TPM\_toggle.bat, and "add" or "delete" the TPM installation check function in reverse order.

## Report

* Group abbreviations:
* Group A: Windows 11 Education, Windows 11 Enterprise, Windows 11 Pro, Windows 11 Pro Education, Windows 11 Pro for Workstations, Windows 11 IoT Enterprise
* Group B: Windows 11 Education N, Windows 11 Enterprise N, Windows 11 Pro N, Windows 11 Pro Education N, Windows 11 Pro N for Workstations
* Group C: Windows 11 Home, Windows 11 Home Single Language
* Group D: Windows 11 Home N

### Reports: Different versions

1. Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~~10.0.22621.382 A, C

Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.317

1. Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~~10.0.22621.1 A, C

Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1

1. Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B
2. Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1
3. Microsoft-OneCore-ApplicationModel-Sync-Desktop-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.317 A, B, C, D
4. Microsoft-OneCore-DirectX-Database-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
5. Microsoft-Windows-Client-LanguagePack-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.382 A, B, C, D
6. Microsoft-Windows-Ethernet-Client-Intel-E1i68x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
7. Microsoft-Windows-Ethernet-Client-Intel-E2f68-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
8. Microsoft-Windows-Ethernet-Client-Realtek-Rtcx21x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
9. Microsoft-Windows-Ethernet-Client-Vmware-Vmxnet3-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
10. Microsoft-Windows-FodMetadata-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
11. Microsoft-Windows-Foundation-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
12. Microsoft-Windows-Hello-Face-Package~31bf3856ad364e35~amd64~~10.0.22621.317 A, B, C, D
13. Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~~11.0.22621.1 A, B, C, D

Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~en-US~11.0.22621.1 A, B, C, D

1. Microsoft-Windows-Kernel-LA57-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.382 A, B, C, D
2. Microsoft-Windows-LanguageFeatures-Basic-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
3. Microsoft-Windows-LanguageFeatures-Handwriting-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
4. Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
5. Microsoft-Windows-LanguageFeatures-Speech-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
6. Microsoft-Windows-LanguageFeatures-TextToSpeech-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
7. Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D

Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 A, B, C, D

1. Microsoft-Windows -Notepad-System-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1 A, B, C, D

Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 A, B, C, D

1. Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D

Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 A, B, C, D

1. Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~~10.0.22621.1 A, B, C, D

Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 A, B, C, D

1. Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D

Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 A, B, C, D

1. Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~~10.0.22621.1 A, B, C, D

Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 A, B, C, D

1. Microsoft-Windows-TabletPCMath-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
2. Microsoft-Windows-Wallpaper-Content-Extended-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
3. Microsoft-Windows-Wifi-Client-Broadcom-Bcmpciedhd63-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
4. Microsoft-Windows-Wifi-Client-Broadcom-Bcmwl63a-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
5. Microsoft-Windows-Wifi-Client-Broadcom-Bcmwl63al-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
6. Microsoft-Windows-Wifi-Client-Intel-Netwbw02-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
7. Microsoft-Windows-Wifi-Client-Intel-Netwew00-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
8. Microsoft-Windows-Wifi-Client-Intel-Netwew01-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
9. Microsoft-Windows-Wifi-Client-Intel-Netwlv64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
10. Microsoft-Windows-Wifi-Client-Intel-Netwns64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
11. Microsoft-Windows-Wifi-Client-Intel-Netwsw00-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
12. Microsoft-Windows-Wifi-Client-Intel-Netwtw02-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
13. Microsoft-Windows-Wifi-Client-Intel-Netwtw04-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
14. Microsoft-Windows-Wifi-Client-Intel-Netwtw06-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
15. Microsoft-Windows-Wifi-Client-Intel-Netwtw08-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
16. Microsoft-Windows-Wifi-Client-Intel-Netwtw10-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
17. Microsoft-Windows-Wifi-Client-Marvel-Mrvlpcie8897-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
18. Microsoft-Windows-Wifi-Client-Qualcomm-Athw8x-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
19. Microsoft-Windows-Wifi-Client-Qualcomm-Athwnx-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
20. Microsoft-Windows-Wifi-Client-Qualcomm-Qcamain10x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
21. Microsoft-Windows-Wifi-Client-Ralink-Netr28x-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
22. Microsoft-Windows-Wifi-Client-Realtek-Rtl8187se-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
23. Microsoft-Windows-Wifi-Client-Realtek-Rtl8192se-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
24. Microsoft-Windows-Wifi-Client-Realtek-Rtl819xp-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
25. Microsoft-Windows-Wifi-Client-Realtek-Rtl85n64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
26. Microsoft-Windows-Wifi-Client-Realtek-Rtwlane-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
27. Microsoft-Windows-Wifi-Client-Realtek-Rtwlane01-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
28. Microsoft-Windows-Wifi-Client-Realtek-Rtwlane13-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
29. Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.382 A, B, C, D

Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621 A, B, C, D

1. Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1 A, B, C, D

Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 A, B, C, D

1. Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.317 A, B, C, D

Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 A, B, C, D

1. Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1 A, B, C, D

Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 A, B, C, D

1. OpenSSH-Client-Package~31bf3856ad364e35~amd64~~10.0.22621.1 A, B, C, D
2. Package\_for\_DotNetRollup\_481~31bf3856ad364e35~amd64~~10.0.9065.6 A, B, C, D
3. Package\_for\_RollupFix~31bf3856ad364e35~amd64~~22621.382.1.13 A, B, C, D
4. Package\_for\_ServicingStack\_378~31bf3856ad364e35~amd64~~22621.378.1.0 A, B, C, D

### Report: Language installation packages apply to Install.wim

1. Microsoft-Windows-MediaPlayer-Package-AMD64-zh-CN.cab A, C

Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. Microsoft-Windows-MediaPlayer-Package-wow64-zh-CN.cab A,C

Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1

1. Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab A, B

Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. Microsoft-Windows-LanguageFeatures-Fonts-Hans-Package~31bf3856ad364e35~AMD64~~.cab A, B, C, D

Microsoft-Windows-LanguageFeatures-Fonts-Hans-Package~31bf3856ad364e35~amd64~~10.0.22621.1

1. Microsoft-Windows-Client-Language-Pack\_x64\_zh-CN.cab A, B, C, D

Microsoft-Windows-Client-LanguagePack-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~AMD64~zh-CN~.cab A, B, C, D

Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~zh-CN~11.0.22621.1

1. Microsoft-Windows-LanguageFeatures-Basic-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab A, B, C, D

Microsoft-Windows-LanguageFeatures-Basic-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1

1. Microsoft-Windows-LanguageFeatures-Handwriting-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab A, B, C, D

Microsoft-Windows-LanguageFeatures-Handwriting-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1

1. Microsoft-Windows-LanguageFeatures-OCR-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab A, B, C, D

Microsoft-Windows-LanguageFeatures-OCR-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1

1. Microsoft-Windows-LanguageFeatures-Speech-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab A, B, C, D

Microsoft-Windows-LanguageFeatures-Speech-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1

1. Microsoft-Windows-LanguageFeatures-TextToSpeech-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab A, B, C, D

Microsoft-Windows-LanguageFeatures-TextToSpeech-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1

1. Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab A, B, C, D

Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~zh-CN~.cab A, B, C, D

Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1

1. Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab A, B, C, D

Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~zh-CN~.cab

Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1

1. Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~AMD64~zh-CN~.cab A, B, C, D

Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~zh-CN~.cab A, B, C, D

Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1

1. Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab A, B, C, D

Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~zh-CN~.cab A, B, C, D

Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1

1. Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab A, B, C, D

Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~zh-CN~.cab A, B, C, D

Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1

### Report: Language installation packages apply to WinRE.wim

1. WinPE-FontSupport-zh-CN.cab

Microsoft-Windows-WinPE-FontSupport-ZH-CN-Package~31bf3856ad364e35~amd64~~10.0.22621.1

1. lp.cab

Microsoft-Windows-WinPE-LanguagePack-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-appxdeployment\_zh-CN.cab

WinPE-AppxDeployment-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-appxpackaging\_zh-CN.cab

WinPE-AppxPackaging-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-atbroker\_zh-CN.cab

WinPE-ATBroker-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-audiocore\_zh-CN.cab

WinPE-AudioCore-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-audiodrivers\_zh-CN.cab

WinPE-AudioDrivers-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-enhancedstorage\_zh-CN.cab

WinPE-EnhancedStorage-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-hta\_zh-CN.cab

WinPE-HTA-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-narrator\_zh-CN.cab

WinPE-Narrator-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-opcservices\_zh-CN.cab

WinPE-OpcServices-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-rejuv\_zh-CN.cab

WinPE-Rejuv-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-scripting\_zh-CN.cab

WinPE-Scripting-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-securestartup\_zh-CN.cab

WinPE-SecureStartup-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-speech-tts\_zh-CN.cab

WinPE-Speech-TTS-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-srh\_zh-CN.cab

WinPE-SRH-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-srt\_zh-CN.cab

WinPE-SRT-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-storagewmi\_zh-CN.cab

WinPE-StorageWMI-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-wds-tools\_zh-CN.cab

WinPE-WDS-Tools-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-wifi\_zh-CN.cab

WinPE-WiFi-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-windowsupdate\_zh-CN.cab

WinPE-WindowsUpdate-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-wmi\_zh-CN.cab

WinPE-WMI-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

### Report: Language installation packages apply to Boot.wim

1. WinPE-FontSupport-zh-CN.cab

Microsoft-Windows-WinPE-FontSupport-ZH-CN-Package~31bf3856ad364e35~amd64~~10.0.22621.1

1. lp.cab

Microsoft-Windows-WinPE-LanguagePack-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-atbroker\_zh-CN.cab

WinPE-ATBroker-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-audiocore\_zh-CN.cab

WinPE-AudioCore-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-audiodrivers\_zh-CN.cab

WinPE-AudioDrivers-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-enhancedstorage\_zh-CN.cab

WinPE-EnhancedStorage-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-narrator\_zh-CN.cab

WinPE-Narrator-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-scripting\_zh-CN.cab

WinPE-Scripting-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-securestartup\_zh-CN.cab

WinPE-SecureStartup-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. WINPE-SETUP-CLIENT\_zh-CN.CAB

WinPE-Setup-Client-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. WinPE-Setup\_zh-CN.cab

WinPE-Setup-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-speech-tts\_zh-CN.cab

WinPE-Speech-TTS-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-srh\_zh-CN.cab

WinPE-SRH-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-srt\_zh-CN.cab

WinPE-SRT-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-wds-tools\_zh-CN.cab

WinPE-WDS-Tools-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

1. winpe-wmi\_zh-CN.cab

WinPE-WMI-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1

## Common problem

### Clean all mounts to

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

Dismount-WindowsImage -Path "D:\OS\_11\_Custom\Install\Install\Mount" -Discard

Dismount-WindowsImage -Path "D:\OS\_11\_Custom\Install\WinRE\Mount" -Discard

Dismount-WindowsImage -Path "D:\OS\_11\_Custom\Boot\Boot\Mount" -Discard

### Fix the problem of abnormal mounting

1. View mounted

Get-WindowsImage -Mounted

1. Delete the DISM mount record saved in the registry

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

1. Delete all resources associated with the corrupted mounted image

Clear-WindowsCorruptMountPoint

Dism /cleanup-wim

### Clean directories

Remove-Item "D:\OS\_11\_Custom" -Force -Recurse

Remove-Item "D:\OS\_11\Sources\`$OEM$" -Force -Recurse

Remove-Item -Path "D:\OS\_11\Autounattend.xml" -Force

Remove-Item -Path "D:\OS\_11\Sources\Unattend.xml" -Force

## Known issues

None yet

Yi’ 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](mailto:775159955@qq.com), [ilikeyi@outlook.com](mailto:ilikeyi@outlook.com)

Document version: 1.1

Documentation model: Full version

Translation: Chinese to English version

Updated: 2024 - 6

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