|  |  |
| --- | --- |
| A logo of a cube  Description automatically generated | Yi  Yi’s Solutions |

封装教程: Microsoft Windows 11 22H2

不同的系统版本，有着不同的封装方式，封装过程中包含：“语言包：添加、关联、删除”、“驱动：添加、删除”、“累积更新：添加、删除”、“InBox Appx：添加、更新、标记”等。

在这背后藏着很多的隐藏故事，想解开这些，你准备好开始尝试封装了吗？

包含内容：

1. [部署映像](#_部署映像)
2. [常见问题](#_常见问题)
3. [已知问题](#_已知问题)
4. [组件：映像中已安装的所有包，对比](#_组件：映像中已安装的所有包，对比)

作者 Feng Yi

网站 <https://fengYi.tel>

版本 1.0

发行日期 2024 / 1

1. 目录
2. [部署映像 第 6页](#_部署映像)
   * 1. [先决条件 第 6页](#_先决条件)
3. [ISO 工具 第 6页](#_ISO_工具)
4. [要求 第 6页](#_要求)
5. [系统安装包 第 6页](#_系统安装包)
6. [语言包 第 6页](#_语言包_3)
   1. [学习 第 6页](#_学习)
   2. [语言包：下载 第 7页](#_语言包：下载)
   3. [语言包：修复 第 7页](#_语言包：修复_1)
7. [命令行 第 8页](#_命令行)
   * 1. [语言包：提取 第 8页](#_语言包：提取)
8. [语言包：准备 第 8页](#_语言包：准备)
9. [语言包：提取方案 第 8页](#_语言包：提取方案)
10. [执行提取命令 第 9页](#_执行提取命令)
    * 1. [自定义部署映像 第 16页](#_自定义部署映像)
11. [自定义部署映像：Install.wim 第 16页](#_自定义部署映像：Install.wim)
12. [查看 Install.wim 详细信息 第 16页](#_查看_Install.wim_详细信息)
13. [指定挂载路径：Install.wim 第 16页](#_指定挂载_Install_路径)
14. [开始挂载 Install.wim 第 16页](#_开始挂载_Install.wim)
    1. [自定义部署映像：WinRE.wim 第 16页](#_自定义部署映像：WinRE.wim)
       1. [查看 WinRE.wim 详细信息 第 16页](#_查看_WinRE.wim_详细信息)
       2. [指定挂载路径：WinRE.wim 第 17页](#_指定挂载_WinRE.wim_路径)
       3. [开始挂载 WinRE.wim 第 17页](#_开始挂载_WinRE.wim)
       4. [语言包 第 17页](#_语言包_2)
          1. [语言包：添加 第 17页](#_语言包：添加_2)
          2. [脱机映像语言：更改 第 19页](#_脱机映像语言：更改_2)
             1. [更改默认语言、区域设置和其他国际设置 第 19页](#_更改默认语言、区域设置和其他国际设置_1)
             2. [查看可用的语言设置 第 19页](#_查看可用的语言设置_1)
          3. [语言包：删除 第 19页](#_语言包：删除，可选_2)
          4. [组件：映像中已安装的所有包 第 22页](#_组件：映像中已安装的所有包_2)
       5. [累积更新 第 22页](#_累积更新，可选_1)
          1. [添加 第 22页](#_添加_1)
          2. [删除 第 23页](#_删除_1)
          3. [固化更新 第 23页](#_固化更新，可选项_1)
             1. [固化更新后清理组件 第 23页](#_固化更新后清理组件，可选项_2)

* + 1. [驱动 第 23页](#_驱动)
    2. [保存映像 第 23页](#_保存映像_2)
    3. [卸载映像 第 23页](#_卸载映像_2)
    4. [重建 WinRE.wim 后，可缩小文件大小 第 23页](#_重建_WinRE.wim_后，可缩小文件大小)
    5. [备份 WinRE.wim 第 24页](#_备份_WinRE.wim)
    6. [替换Install.wim 映像内的WinRE.wim 第 25页](#_替换_Install.wim_映像内的)

1. [语言包 第 25页](#_语言包_1)
   1. [语言包：添加 第 25页](#_语言包：添加_1)
   2. [脱机映像语言：更改 第 27页](#_脱机映像语言：更改_1)
      1. [更改默认语言、区域设置和其他国际设置 第 27 页](#_更改默认语言、区域设置和其他国际设置_2)
      2. [查看可用的语言设置 第 27 页](#_查看可用的语言设置_2)
   3. [语言包：删除 第 27页](#_语言包：删除，可选_1)
   4. [组件：映像中已安装的所有包 第 30页](#_组件：映像中已安装的所有包_1)
2. [InBox Apps 第 31页](#_InBox_Apps)
   1. [学习“附件：InBox Apps” 第 31页](附件：inBox_Apps#_学习)
   2. [InBox Apps：已安装 第 31页](#_InBox_Apps：已安装)
   3. [删除已安装的所有预应用程序 第 31页](#_删除已安装的所有预应用程序)
   4. [区域标记：了解 第 32页](#_区域标记：了解)
   5. [区域标记：添加方式 第 32页](#_区域标记：添加方式)
   6. [InBox Apps：安装 第 33页](#_InBox_Apps：安装)
   7. [区域标记：删除 第 42页](#_区域标记：删除)
   8. [InBox Apps：优化 第 43页](#_InBox_Apps：优化)
3. [累积更新 第 43页](#_累积更新)
   1. [初始版本 第 43页](#_初始版本)
   2. [其它版本 第 44页](#_其它版本)
   3. [固化更新 第 44页](#_固化更新，可选项)
      1. [固化更新后清理组件 第 44页](#_固化更新后清理组件，可选项_1)
4. [驱动 第 45页](#_驱动，可选_1)
5. [部署引擎：添加 第 45页](#_部署引擎：添加，可选)
6. [健康 第 45页](#_健康)
7. [替换 WinRE.wim 第 45页](#_替换_WinRE.wim)
8. [保存映像 第 45页](#_保存映像_1)
9. [卸载映像 第 45页](#_卸载映像_1)
10. [重建 Install.wim 后可缩小文件大小 第45页](#_重建_Install.wim_后可缩小文件大小)
11. [如何批量替换 Install.wim 里的所有索引号里的 WinRE.wim 第46页](#_如何批量替换_Install.wim_里的所有索引号里的)
    1. [获取 WimLib 第46页](#_获取_WimLib)
    2. [如何在 Install.wim 里提取和更新 WinRE.wim 第46 页](#_如何在_Install.wim_里提取和更新)
12. [自定义部署映像：boot.wim 第 47页](#_自定义部署映像：boot.wim)
13. [查看 Boot.wim 详细信息 第 48页](#_查看_Boot.wim_文件信息)
14. [指定挂载路径：Boot.wim 第 48页](#_指定挂载_Boot.wim_路径)
15. [开始挂载 Boot.wim 第 48页](#_开始挂载_Boot.wim)
16. [语言包 第 48页](#_语言包)
    1. [语言包：添加 第 48页](#_语言包：添加)
    2. [脱机映像语言：更改 第 50页](#_脱机映像语言：更改)
       1. [更改默认语言、区域设置和其他国际设置 第 50页](#_更改默认语言、区域设置和其他国际设置)
       2. [查看可用的语言设置 第 50页](#_查看可用的语言设置)
    3. [语言包：删除 第 50页](#_语言包：删除，可选)
    4. [组件：映像中已安装的所有包 第 53页](#_组件：映像中已安装的所有包)
    5. [语言包：修复 第 53页](#_语言包：修复)
    6. [语言包：同步到 ISO 安装程序 第 54页](#_语言包：同步到_ISO_安装程序)
    7. [重新生成 Lang.ini 第 54页](#_重新生成_Lang.ini)
       1. [重新生成已挂载目录 lang.ini 第 54页](#_重新生成已挂载目录_lang.ini)
       2. [重新生成 lang.ini 后，同步到安装程序 第 54页](#_重新生成_lang.ini_后，同步到安装程序)
17. [累积更新 第 54页](#_累积更新，可选)
    1. [添加 第 54页](#_添加)
    2. [删除 第 54页](#_删除)
    3. [固化更新 第 54页](#_累积更新，可选)
       1. [固化更新后清理组件 第 54页](#_固化更新后清理组件，可选项)
18. [驱动 第 55页](#_驱动，可选)
19. [其它 第 55页](#_其它)
    1. [绕过 TPM 安装时检查 第 55页](#_绕过_TPM_安装时检查)
20. [保存映像 第 56页](#_保存映像)
21. [卸载映像 第 56页](#_卸载映像)
22. [部署引擎 第 56页](#_部署引擎)
23. [添加方式 第 56页](#_添加方式)
24. [部署引擎：进阶 第 60页](#_部署引擎：进阶)
    * 1. [ISO 第 62页](#_ISO)
25. [生成 ISO 第 62页](#_生成_ISO)
26. [绕过 TPM 安装时检查 第 63页](#_绕过_TPM_安装检查，可选)
27. [常见问题 第 63页](#_常见问题)
28. [清理所有挂载到 第 63页](#_清理所有挂载到)
29. [修复挂载出现异常的问题 第 63页](#_修复挂载出现异常的问题)
30. [清理目录 第 64页](#_清理目录)
31. [已知问题 第 64页](#_已知问题)
32. [组件：映像中已安装的所有包，对比 第 65页](#_组件：映像中已安装的所有包，对比)

## 部署映像

### 先决条件

#### ISO 工具

准备一款可编辑 ISO 文件的软件，例如：PowerISO、DAEMON Tools、ISO Workshop；

#### 要求

学习“附件：OS”，查看“Windows 11”项。

##### 系统安装包

* 1. 准备：en-us\_windows\_11\_business\_editions\_version\_22h2\_x64\_dvd\_17a08ce3.iso
  2. 解压到：D:\en-us\_windows\_11\_business\_editions\_version\_22h2\_x64\_dvd\_17a08c
  3. 解压完成后，将目录en-us\_windows\_11\_business\_editions\_version\_22h2\_x64\_dvd\_17a08c 更改为 D:\OS\_11
  4. 所有脚本、所有路径，已默认设置为 D:\OS\_11 为映像来源。

##### 语言包

###### 学习

阅读时，请了解“蓝色”重要突出部分。

* + 1. [语言概述](https://learn.microsoft.com/zh-cn/windows-hardware/manufacture/desktop/languages-overview?view=windows-11)
    2. [将语言添加到 Windows 11 映像](https://learn.microsoft.com/en-us/azure/virtual-desktop/windows-11-language-packs)
    3. [语言和区域按需功能 (FOD)](https://learn.microsoft.com/zh-cn/windows-hardware/manufacture/desktop/features-on-demand-language-fod?view=windows-11)
       1. 字体
* 添加语言包时，触发了对应的区域时，需添加所需的字体功能，下载“[所有可用语言 FOD 的列表](https://download.microsoft.com/download/7/6/0/7600F9DC-C296-4CF8-B92A-2D85BAFBD5D2/Windows-10-1809-FOD-to-LP-Mapping-Table.xlsx)”了解更多。
* 在“语言包：提取”时，已加入自动识别功能，可了解函数：Function Match\_Required\_Fonts
  + - 1. 区域关联

区域关联是什么？

* 映像语言仅英文版时，添加zh-HK 语言包后，映像语言不会新增，应先安装 zh-TW 后，再安装zh-HK 即可获得对应的关联。
* 可参阅微软官方原版：Windows 10、Windows 11 繁体版。

已知区域关联：

* + - * 1. 区域：zh-TW，可选关联区域：zh-HK
      1. 其它区域特定的要求

触发已知区域时，需添加特定的“程序包”。

* + - * 1. 区域：zh-TW，程序包：Microsoft-Windows-InternationalFeatures-Taiwan-Package~31bf3856ad364e35~amd64~~.cab

说明：对台湾日期格式设置要求的补充支持。 将为位于台湾的客户提供包。

建议：仅在运送到台湾市场的设备上预安装。 未在设备上安装此功能会导致对使用台湾日历的任何 API 调用失败。

存在争议：

* 在测试中发现，微软官方原版 Windows 10、Windows 11 里并未发现原版映像里安装此程序包，但是在建议项里有已知问题，到底是遵循与微软官方原版一致，由封装师自由选择安装与否。

###### 语言包：下载

[22621.1.220506-1250.ni\_release\_amd64fre\_CLIENT\_LOF\_PACKAGES\_OEM.iso](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)

###### 语言包：修复

* + 1. 任选一个网站并打开：
       1. <https://uupdump.net>
       2. <https://uup.ee>
    2. 打开后，搜索关键词：22621.382，在搜索结果选择：Windows 11, version 22H2 (22621.382) amd64
    3. 打开后，选择“全部文件”；
    4. 在“全部文件”页面里依次搜索绿色部分并下载：
       1. 适用于：Install.wim（ 1项 ）
          1. Media
       2. 适用于：WinRE.wim，暂无
       3. 适用于：Boot.wim，暂无
    5. 下载所有文件后，拉动到页面下方，下载并运行“生成重命名脚本（Windows）”；
    6. 使用ISO 编辑软件，编辑 22621.1.220506-1250.ni\_release\_amd64fre\_CLIENT\_LOF\_PACKAGES\_OEM.iso，将下载的文件添加到 [ISO]:\LanguagesAndOptionalFeatures 目录里；

##### InBox Apps

* 1. 下载：[22621.1778.230511-2102.ni\_release\_svc\_prod3\_amd64fre\_InboxApps.iso](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. 下载：[22621.1.220506-1250.ni\_release\_amd64fre\_InboxApps.iso](https://software-static.download.prss.microsoft.com/dbazure/988969d5-f34g-4e03-ac9d-1f9786c66749/22621.1.220506-1250.ni_release_amd64fre_InboxApps.iso) 后，提取以下文件后保存到桌面：
     1. Microsoft.HEVCVideoExtension\_8wekyb3d8bbwe.x64.appx
     2. Microsoft.HEVCVideoExtension\_8wekyb3d8bbwe.x64.xml
  3. 使用 ISO 编辑工具，编辑 22621.1778.230511-2102.ni\_release\_svc\_prod3\_amd64fre\_InboxApps.iso，将已提取文件添加到 [ISO]:\packages 目录里；

#### 命令行

1. 可选“Terminal ”或“PowerShell ISE”，未安装“Terminal”，请前往 <https://github.com/microsoft/terminal/releases> 后下载；
2. 以管理员身份打开“Terminal”或“PowerShell ISE”，设置 PowerShell 执行策略：绕过，PS 命令行：

Set-ExecutionPolicy -ExecutionPolicy Bypass -Force

1. 在本文中，绿色部分属于 PS 命令行，请复制后，粘贴到“Terminal”对话框，按回车键（Enter）后开始运行；
2. 有 .ps1 时，点击文件右键，选择以 PowerShell 运行，或复制路径，粘贴到“Terminal ”或“PowerShell ISE”里运行。

### 语言包：提取

#### 语言包：准备

挂载 22621.1.220506-1250.ni\_release\_amd64fre\_CLIENT\_LOF\_PACKAGES\_OEM.iso 或解压到任意位置；

#### 语言包：提取方案

##### 添加

* 1. 语言名称：简体中文 - 中国，区域：zh-CN，适用范围：Install.Wim，Boot.Wim，WinRE.Wim

##### 删除

* 1. 语言名称：英语 - 美国，区域：en-US，适用范围：Install.Wim，Boot.Wim，WinRE.Wim

#### 执行提取命令

* Auto = 自动搜索本地所有磁盘，默认；
  + - * 自定义路径，例如指定为 E 盘：$ISO = "E:\"
      * Extract.ps1
        + [\Expand\Extract.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Extract.ps1)
        + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Extract.ps1>
      * 复制代码

$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 }

### 自定义部署映像

#### 自定义部署映像：Install.wim

##### 查看 Install.wim 详细信息

映像名称、映像描述、映像大小、架构、版本、索引号等；

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

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

循环操作区域，开始，

##### 指定挂载 Install 路径

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

##### 开始挂载 Install.wim

默认索引号：1

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

处理 Install.wim 映像内的文件，可选项，开始，

###### 自定义部署映像：WinRE.wim

注意：

* WinRE.wim 属于 Install.wim 映像内的文件；
* Install.wim 有多个索引号时，仅处理任意一个 WinRE.wim 即可；
* 同步至所有索引号即可减少 Install.wim 体积，学习“如何批量替换 Install.wim 里的所有索引号里的 WinRE.wim”。
  + 1. 查看 WinRE.wim 详细信息

映像名称、映像描述、映像大小、架构、版本、索引号等；

$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. 指定挂载 WinRE.wim 路径

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

* + 1. 开始挂载 WinRE.wim

默认索引号：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. 语言包

自动安装语言包：获取“组件：映像中已安装的所有包”后进行匹配，匹配到对应的名称后，再安装本地对应的语言包文件。

* + - 1. 语言包：添加
* WinRE.Instl.lang.ps1
  + [\Expand\Install\WinRE\WinRE.Instl.lang.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Install\WinRE\WinRE.Instl.lang.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Install/WinRE/WinRE.Instl.lang.ps1>
* 复制代码

$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. 脱机映像语言：更改
         1. 更改默认语言、区域设置和其他国际设置

区域：zh-CN

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

* + - * 1. 查看可用的语言设置

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

* + - 1. 语言包：删除，可选
* 添加语言后，如果要部署到非英语区域，可通过删除英语语言组件来节省空间。 删除语言时，请按与添加语言组件相反的顺序卸载语言组件。
* 添加中文后，反向删除“英语 - 美国”，区域：en-US，需提前提取语言包
* WinRE.Del.Specified.lang.Tag.ps1
  + [\Expand\Install\WinRE\WinRE.Del.Specified.lang.Tag.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Install\WinRE\WinRE.Del.Specified.lang.Tag.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Install/WinRE/WinRE.Del.Specified.lang.Tag.ps1>
* 复制代码

$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. 组件：映像中已安装的所有包
         1. 查看

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

* + - * 1. 导出到 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. 累积更新，可选

准备可用的累积更新文件，请更改示例文件名：KB\_WinRE.cab

* + - 1. 添加

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

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

* + - 1. 删除

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

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

* + - 1. 固化更新，可选项

固化后不可卸载，固化时将清理恢复映像并重置任何被取代的组件的基础。

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

* + - * 1. 固化更新后清理组件，可选项

$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. 驱动
    2. 保存映像

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

* + 1. 卸载映像

关闭所有可能正在访问映像中文件的应用程序，包括文件资源管理器。

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

* + 1. 重建 WinRE.wim 后，可缩小文件大小
* WinRE.Rebuild.ps1
  + [\Expand\Install\WinRE\WinRE.Rebuild.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Install\WinRE\WinRE.Rebuild.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Install/WinRE/WinRE.Rebuild.ps1>
* 复制代码

$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. 备份 WinRE.wim
* WinRE.Backup.ps1
  + [\Expand\Install\WinRE\WinRE.Backup.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Install\WinRE\WinRE.Backup.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Install/WinRE/WinRE.Backup.ps1>
* 复制代码

$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. 替换 Install.wim 映像内的 WinRE.wim
* 每次挂载 Install.wim 后“[替换 WinRE.wim](#_替换_WinRE.wim)”；
* 学习“[获取 Install.wim 所有索引号后并替换旧的 WinRE.wim](#_获取_Install.wim_所有索引号后并替换旧的)”。

处理 Install.wim 映像内的文件，结束。

##### 语言包

自动安装语言包：获取“组件：映像中已安装的所有包”后进行匹配，匹配到对应的名称后，再安装本地对应的语言包文件。

###### 语言包：添加

* Install.Instl.lang.ps1
  + [\Expand\Install\Install.Instl.lang.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Install\Install.Instl.lang.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Install/Install.Instl.lang.ps1>
* 复制代码

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

$Sources = "D:\OS\_11\_Custom\Install\Install\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)\Microsoft-Windows-LanguageFeatures-Fonts-Hans-Package~31bf3856ad364e35~amd64~~.cab"

$Language\_List = @(

@{ Match = "\*Client-LanguagePack-Package\*"; File = "Microsoft-Windows-Client-Language-Pack\_x64\_zh-CN.cab"; }

@{ Match = "\*LanguageFeatures-Basic\*"; File = "Microsoft-Windows-LanguageFeatures-Basic-zh-CN-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*Handwriting\*"; File = "Microsoft-Windows-LanguageFeatures-Handwriting-zh-CN-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*OCR\*"; File = "Microsoft-Windows-LanguageFeatures-OCR-zh-CN-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*LanguageFeatures-Speech\*"; File = "Microsoft-Windows-LanguageFeatures-Speech-zh-CN-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*TextToSpeech\*"; File = "Microsoft-Windows-LanguageFeatures-TextToSpeech-zh-CN-Package~31bf3856ad364e35~amd64~~.cab"; }

@{ Match = "\*InternetExplorer\*"; File = "Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~AMD64~zh-CN~.cab"; }

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

@{ Match = "\*Notepad\*wow64\*"; File = "Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~zh-CN~.cab"; }

@{ Match = "\*MediaPlayer\*amd64\*"; File = "Microsoft-Windows-MediaPlayer-Package-AMD64-zh-CN.cab"; }

@{ Match = "\*MediaPlayer\*wow64\*"; File = "Microsoft-Windows-MediaPlayer-Package-wow64-zh-CN.cab"; }

@{ Match = "\*PowerShell\*amd64\*"; File = "Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~zh-CN~.cab"; }

@{ Match = "\*PowerShell\*wow64\*"; File = "Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~zh-CN~.cab"; }

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

@{ Match = "\*StepsRecorder\*amd64\*"; File = "Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~zh-CN~.cab"; }

@{ Match = "\*StepsRecorder\*wow64\*"; File = "Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~zh-CN~.cab"; }

@{ Match = "\*WMIC\*FoD\*Package\*amd64\*"; File = "Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab"; }

@{ Match = "\*WMIC\*FoD\*Package\*wow64\*"; File = "Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~zh-CN~.cab"; }

@{ Match = "\*WordPad\*amd64\*"; File = "Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~zh-CN~.cab"; }

@{ Match = "\*WordPad\*wow64\*"; File = "Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~zh-CN~.cab"; }

@{ Match = "\*Client\*LanguagePack\*zh-TW\*"; File = "Microsoft-Windows-InternationalFeatures-Taiwan-Package~31bf3856ad364e35~amd64~~.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

}

}

}

###### 脱机映像语言：更改

* 从 Windows 11 开始，DISM 设置的[默认系统 UI 语言](https://learn.microsoft.com/zh-cn/windows/win32/intl/user-interface-language-management)在所有版本中保持不变（家庭版除外）。 对于[所有商业版](https://www.microsoft.com/windowsforbusiness/compare)，在开箱即用体验 (OOBE) 期间选择的语言会设置为[系统首选 UI 语言](https://learn.microsoft.com/zh-cn/windows/win32/intl/user-interface-language-management)，Windows 将以此语言显示；对于家庭版，在 OOBE 期间选择的语言将继续用作默认系统 UI 语言。
* 从 Windows 10 版本 2004 开始，如果将基于 .appx 的语言体验包 (LXP) 支持的语言作为参数传递，则该语言将设置为系统首选 UI 语言，其父语言将设置为默认系统 UI 语言。 在以前的版本中，仅支持基于 .cab 的语言包。
  + 1. 更改默认语言、区域设置和其他国际设置

区域：zh-CN

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

* + 1. 查看可用的语言设置

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

###### 语言包：删除，可选

* 添加语言后，如果要部署到非英语区域，可通过删除英语语言组件来节省空间。 删除语言时，请按与添加语言组件相反的顺序卸载语言组件。
* 添加中文后，反向删除“英语 - 美国”，区域：en-US，需提前提取语言包
* Install.Del.Specified.lang.Tag.ps1
  + [\Expand\Install\Install.Del.Specified.lang.Tag.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Install\Install.Del.Specified.lang.Tag.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Install/Install/Install.Del.Specified.lang.Tag.ps1>
* 复制代码

$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

}

}

}

###### 组件：映像中已安装的所有包

* + 1. 查看

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

* + 1. 导出到 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

###### 学习“附件：inBox Apps”

了解不同的操作系统预安装应用，应用商店连接。

###### InBox Apps：已安装

* + 1. 查看

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

* + 1. 导出到 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

###### 删除已安装的所有预应用程序

* Install.InBox.Appx.Clear.all.ps1
  + [\Expand\Install\Install.InBox.Appx.Clear.all.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Install\Install.InBox.Appx.Clear.all.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Install/Install.InBox.Appx.Clear.all.ps1>
* 复制代码

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

}

}

###### 区域标记：了解

了解区域标记前，先了解 ZuneMusic 应用程序可用的语言列表：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

查看脱机映像已安装的语言

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

查看结果已安装的语言：en-US、zh-CN，在安装 ZuneMusic 应用时，安装程序会匹配“已安装的语言”，自动过滤未安装的语言列表，这就是区域标记。

###### 区域标记：添加方式

* + 1. 执行“语言包：添加”
    2. 安装“本地语言体验包（LXPs）”

微软官方向 Windows 10 提供了本地语言体验包（LXPS）安装文件，Windows 11 不再提供，要想获取离线安装包：

* + - 1. 使用“Windows 本地语言体验包（LXPs）下载器”下载

了解：<https://github.com/ilikeyi/LXPs>

下载后保存到：D:\OS\_11\_Custom\Install\Install\InBox.Appx，文件格式：LanguageExperiencePack.zh-CN.Neutral.Appx

* + - 1. 手动下载
         1. 查看“附件：LXPs”

等待下载“简体中文 - 中国”的本地语言体验包（LXPs），区域：zh-CN，应用程序 ID：9NRMNT6GMZ70，商店连接：https://www.microsoft.com/store/productId/9NRMNT6GMZ70

* + - * 1. 打开网站：<https://store.rg-adguard.net>

搜索关键词：

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

网页内搜索 22621 内容，搜索结果：Microsoft.LanguageExperiencePackzh-CN\_22621.\*. neutral\_\_8wekyb3d8bbwe.appx

下载后保存到 D:\OS\_11\_Custom\Install\Install\InBox.Appx 目录里，重命名：LanguageExperiencePack.zh-cn.Neutral.Appx

* + - 1. 执行安装命令安装本地语言体验包（LXPs）

了解区域标记添加方式后，获得 LanguageExperiencePack.zh-cn.Neutral 后，执行安装命令：

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：已安装的应用程序包
         1. 查看

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

* + - * 1. 导到出 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：安装

* + 1. 挂载或解压 InBox Apps 安装文件

挂载 22621.1.220506-1250.ni\_release\_amd64fre\_InboxApps.iso 或解压到任意位置；

* + 1. 执行安装命令后安装 InBox Apps 到：Install.wim
       - Auto = 自动搜索本地所有磁盘，默认；
       - 自定义路径，例如指定为 F 盘：$ISO = "F:\packages"
       - Install.Inst.InBox.Appx.ps1
       - [\Expand\Install\Install.Inst.InBox.Appx.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Install\Install.Inst.InBox.Appx.ps1)
       - <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Install/Install.Inst.InBox.Appx.ps1>
* 复制代码

$ISO = "Auto"

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

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 = @(

@{

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.Sticky.Notes"; "Microsoft.Paint"; "Microsoft.PowerAutomateDesktop"; "Microsoft.ScreenSketch"; "Microsoft.WindowsNotepad"; "Microsoft.WindowsTerminal"; "Clipchamp.Clipchamp"; "Microsoft.Solitaire.Collection"; "Microsoft.WindowsAlarms"; "Microsoft.WindowsFeedbackHub"; "Microsoft.WindowsMaps"; "Microsoft.ZuneMusic"; "Microsoft.BingNews"; "Microsoft.DesktopAppInstaller"; "Microsoft.WindowsCamera"; "Microsoft.Getstarted"; "Microsoft.Cortana"; "Microsoft.BingWeather"; "Microsoft.GetHelp"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.People"; "Microsoft.StorePurchaseApp"; "Microsoft.Todos"; "Microsoft.WebMediaExtensions"; "Microsoft.Windows.Photos"; "Microsoft.WindowsCalculator"; "Microsoft.Windows.CommunicationsApps"; "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.Sticky.Notes"; "Microsoft.Paint"; "Microsoft.PowerAutomateDesktop"; "Microsoft.ScreenSketch"; "Microsoft.WindowsNotepad"; "Microsoft.WindowsTerminal"; "Clipchamp.Clipchamp"; "Microsoft.Solitaire.Collection"; "Microsoft.WindowsAlarms"; "Microsoft.WindowsFeedbackHub"; "Microsoft.WindowsMaps"; "Microsoft.ZuneMusic"; "Microsoft.BingNews"; "Microsoft.DesktopAppInstaller"; "Microsoft.WindowsCamera"; "Microsoft.Getstarted"; "Microsoft.Cortana"; "Microsoft.BingWeather"; "Microsoft.GetHelp"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.People"; "Microsoft.StorePurchaseApp"; "Microsoft.Todos"; "Microsoft.WebMediaExtensions"; "Microsoft.Windows.Photos"; "Microsoft.WindowsCalculator"; "Microsoft.Windows.CommunicationsApps"; "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"; "CloudEditionN"; "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.Sticky.Notes"; "Microsoft.Paint"; "Microsoft.PowerAutomateDesktop"; "Microsoft.ScreenSketch"; "Microsoft.WindowsNotepad"; "Microsoft.WindowsTerminal"; "Clipchamp.Clipchamp"; "Microsoft.Solitaire.Collection"; "Microsoft.WindowsAlarms"; "Microsoft.WindowsFeedbackHub"; "Microsoft.WindowsMaps"; "Microsoft.BingNews"; "Microsoft.DesktopAppInstaller"; "Microsoft.WindowsCamera"; "Microsoft.Getstarted"; "Microsoft.Cortana"; "Microsoft.BingWeather"; "Microsoft.GetHelp"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.People"; "Microsoft.StorePurchaseApp"; "Microsoft.Todos"; "Microsoft.Windows.Photos"; "Microsoft.WindowsCalculator"; "Microsoft.Windows.CommunicationsApps"; "Microsoft.XboxGameOverlay"; "Microsoft.XboxIdentityProvider"; "Microsoft.XboxSpeechToTextOverlay"; "Microsoft.YourPhone"; "MicrosoftCorporationII.QuickAssist"; "MicrosoftWindows.Client.WebExperience";

)

}

)

$Allow\_Install\_App = @()

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

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

}

$InBoxApps = @(

@{ Name = "Microsoft.UI.Xaml.2.3"; File = "Microsoft.UI.Xaml.x64.2.3.appx"; License = ""; }

@{ Name = "Microsoft.UI.Xaml.2.4"; File = "Microsoft.UI.Xaml.x64.2.4.appx"; License = ""; }

@{ Name = "Microsoft.UI.Xaml.2.7"; File = "Microsoft.UI.Xaml.x64.2.7.appx"; License = ""; }

@{ Name = "Microsoft.NET.Native.Framework.2.2"; File = "Microsoft.NET.Native.Framework.x64.2.2.appx"; License = ""; }

@{ Name = "Microsoft.NET.Native.Runtime.2.2"; File = "Microsoft.NET.Native.Runtime.x64.2.2.appx"; License = ""; }

@{ Name = "Microsoft.VCLibs.140.00"; File = "Microsoft.VCLibs.x64.14.00.appx"; License = ""; }

@{ Name = "Microsoft.VCLibs.140.00.UWPDesktop"; File = "Microsoft.VCLibs.x64.14.00.UWPDesktop.appx"; License = ""; }

@{ Name = "Microsoft.WindowsStore"; File = "Microsoft.WindowsStore\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsStore\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.HEIFImageExtension"; File = "Microsoft.HEIFImageExtension\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.HEIFImageExtension\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.HEVCVideoExtension"; File = "Microsoft.HEVCVideoExtension\_8wekyb3d8bbwe.x64.appx"; License = "Microsoft.HEVCVideoExtension\_8wekyb3d8bbwe.x64.xml"; }

@{ Name = "Microsoft.SecHealthUI"; File = "Microsoft.SecHealthUI\_8wekyb3d8bbwe.x64.appx"; License = "Microsoft.SecHealthUI\_8wekyb3d8bbwe.x64.xml"; }

@{ Name = "Microsoft.VP9VideoExtensions"; File = "Microsoft.VP9VideoExtensions\_8wekyb3d8bbwe.x64.appx"; License = "Microsoft.VP9VideoExtensions\_8wekyb3d8bbwe.x64.xml"; }

@{ Name = "Microsoft.WebpImageExtension"; File = "Microsoft.WebpImageExtension\_8wekyb3d8bbwe.x64.appx"; License = "Microsoft.WebpImageExtension\_8wekyb3d8bbwe.x64.xml"; }

@{ Name = "Microsoft.GamingApp"; File = "Microsoft.GamingApp\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.GamingApp\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.Sticky.Notes"; File = "Microsoft.MicrosoftStickyNotes\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.MicrosoftStickyNotes\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.Paint"; File = "Microsoft.Paint\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.Paint\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.PowerAutomateDesktop"; File = "Microsoft.PowerAutomateDesktop\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.PowerAutomateDesktop\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.ScreenSketch"; File = "Microsoft.ScreenSketch\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.ScreenSketch\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.WindowsNotepad"; File = "Microsoft.WindowsNotepad\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsNotepad\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.WindowsTerminal"; File = "Microsoft.WindowsTerminal\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsTerminal\_8wekyb3d8bbwe.xml"; }

@{ Name = "Clipchamp.Clipchamp"; File = "Clipchamp.Clipchamp\_yxz26nhyzhsrt.msixbundle"; License = "Clipchamp.Clipchamp\_yxz26nhyzhsrt.xml"; }

@{ Name = "Microsoft.Solitaire.Collection"; File = "Microsoft.MicrosoftSolitaireCollection\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.MicrosoftSolitaireCollection\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.WindowsAlarms"; File = "Microsoft.WindowsAlarms\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsAlarms\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.WindowsFeedbackHub"; File = "Microsoft.WindowsFeedbackHub\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsFeedbackHub\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.WindowsMaps"; File = "Microsoft.WindowsMaps\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsMaps\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.ZuneMusic"; File = "Microsoft.ZuneMusic\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.ZuneMusic\_8wekyb3d8bbwe.xml"; }

@{ Name = "MicrosoftCorporationII.MicrosoftFamily"; File = "MicrosoftCorporationII.MicrosoftFamily\_8wekyb3d8bbwe.msixbundle"; License = "MicrosoftCorporationII.MicrosoftFamily\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.BingNews"; File = "Microsoft.BingNews\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.BingNews\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.DesktopAppInstaller"; File = "Microsoft.DesktopAppInstaller\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.DesktopAppInstaller\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.WindowsCamera"; File = "Microsoft.WindowsCamera\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsCamera\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.Getstarted"; File = "Microsoft.Getstarted\_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.Getstarted\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.Cortana"; File = "Microsoft.CortanaApp\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.CortanaApp\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.BingWeather"; File = "Microsoft.BingWeather\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.BingWeather\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.GetHelp"; File = "Microsoft.GetHelp\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.GetHelp\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.MicrosoftOfficeHub"; File = "Microsoft.MicrosoftOfficeHub\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.MicrosoftOfficeHub\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.People"; File = "Microsoft.People\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.People\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.StorePurchaseApp"; File = "Microsoft.StorePurchaseApp\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.StorePurchaseApp\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.Todos"; File = "Microsoft.Todos\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.Todos\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.WebMediaExtensions"; File = "Microsoft.WebMediaExtensions\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.WebMediaExtensions\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.Windows.Photos"; File = "Microsoft.Windows.Photos\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.Windows.Photos\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.WindowsCalculator"; File = "Microsoft.WindowsCalculator\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.WindowsCalculator\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.Windows.CommunicationsApps"; File = "Microsoft.WindowsCommunicationsApps\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.WindowsCommunicationsApps\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.WindowsSoundRecorder"; File = "Microsoft.WindowsSoundRecorder\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.WindowsSoundRecorder\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.Xbox.TCUI"; File = "Microsoft.Xbox.TCUI\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.Xbox.TCUI\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.XboxGameOverlay"; File = "Microsoft.XboxGameOverlay\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.XboxGameOverlay\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.XboxGamingOverlay"; File = "Microsoft.XboxGamingOverlay\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.XboxGamingOverlay\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.XboxIdentityProvider"; File = "Microsoft.XboxIdentityProvider\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.XboxIdentityProvider\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.XboxSpeechToTextOverlay"; File = "Microsoft.XboxSpeechToTextOverlay\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.XboxSpeechToTextOverlay\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.YourPhone"; File = "Microsoft.YourPhone\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.YourPhone\_8wekyb3d8bbwe.xml"; }

@{ Name = "Microsoft.ZuneVideo"; File = "Microsoft.ZuneVideo\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.ZuneVideo\_8wekyb3d8bbwe.xml"; }

@{ Name = "MicrosoftCorporationII.QuickAssist"; File = "MicrosoftCorporationII.QuickAssist\_8wekyb3d8bbwe.appxbundle"; License = "MicrosoftCorporationII.QuickAssist\_8wekyb3d8bbwe.xml"; }

@{ Name = "MicrosoftWindows.Client.WebExperience"; File = "MicrosoftWindows.Client.WebExperience\_cw5n1h2txyewy.appxbundle"; License = "MicrosoftWindows.Client.WebExperience\_cw5n1h2txyewy.xml"; }

@{ Name = "Microsoft.RawImageExtension"; File = "Microsoft.RawImageExtension\_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.RawImageExtension\_8wekyb3d8bbwe.xml"; }

)

Function Install\_Appx

{

param($File, $License)

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

Write-host " Installing: " -NoNewline; Write-host $File -ForegroundColor Yellow

if (Test-Path -Path $File -PathType Leaf) {

if (Test-Path -Path $License -PathType Leaf) {

Write-host " License: " -NoNewline

Write-host $License -ForegroundColor Yellow

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

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

try {

Add-AppxProvisionedPackage -Path $Mount -PackagePath $File -LicensePath $License -ErrorAction SilentlyContinue | Out-Null

Write-Host "Done" -ForegroundColor Green

} catch {

Write-Host "Failed" -ForegroundColor Red

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

}

} else {

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

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

try {

Add-AppxProvisionedPackage -Path $Mount -PackagePath $File -SkipLicense -ErrorAction SilentlyContinue | Out-Null

Write-Host "Done" -ForegroundColor Green

} catch {

Write-Host "Failed" -ForegroundColor Red

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

}

}

} else {

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

}

}

ForEach ($Rule in $InBoxApps) {

Write-host "`n 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.File -ForegroundColor Yellow

Write-host " Search for License: " -NoNewline

Write-host $Rule.File -ForegroundColor Yellow

if ($ISO -eq "Auto") {

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

$AppPath = Join-Path -Path $\_.Root -ChildPath "packages\$($Rule.File)" -ErrorAction SilentlyContinue

$LicensePath = Join-Path -Path $\_.Root -ChildPath "packages\$($Rule.License)" -ErrorAction SilentlyContinue

if (Test-Path $AppPath -PathType Leaf) {

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

Write-host " Discover apps: " -NoNewLine; Write-host $AppPath -ForegroundColor Green

if (Test-Path $LicensePath -PathType Leaf) {; Write-host " Discover License: " -NoNewLine

Write-host $LicensePath -ForegroundColor Green

} else {

Write-host " License: " -NoNewLine

Write-host "Not found" -ForegroundColor Red

}

Install\_Appx -File $AppPath -License $LicensePath

return

}

}

} else {

Install\_Appx -File "$($ISO)\$($Rule.File)" -License "$($ISO)\$($Rule.License)"

}

} else {

Write-host " Skip the installation" -ForegroundColor Red

}

}

###### 区域标记：删除

* 完成安装 InBox Apps 应用后，区域标记已不再重要，可删除或不删除，删除“本地语言体验包（LXPs），简体中文 - 中国）”，区域标记： zh-CN，可更改为其它区域标记。
* Install.Clear.Flag.ps1
  + [\Expand\Install\Install.Clear.Flag.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Install\Install.Clear.Flag.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Install/Install.Clear.Flag.ps1>
* 复制代码

$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：优化

在安装应用后，应优化预配 Appx 包，通过用硬链接替换相同的文件来减少应用的磁盘使用量，仅针对脱机映像。

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

##### 累积更新

推荐你下载最新版，提供初始版本的累积更新，解决同一版本号重新安装后刷新“组件：映像中已安装的所有包”状态。

###### 初始版本

* + 1. 如何下载

累积更新 KB5016632 已无法从 <https://www.catalog.update.microsoft.com/Search.aspx?q=KB5016632> 里搜索到。

* + - 1. 任选一个网站并打开：
         1. <https://uupdump.net>
         2. <https://uup.ee>
      2. 打开后，搜索关键词：22621.382，在搜索结果选择：Windows 11, version 22H2 (22621.382) amd64
      3. 打开后，选择“全部文件”；
      4. 在“全部文件”页面里搜索：KB5016632，下载：
* Windows11.0-KB5016632-x64.cab
* Windows11.0-KB5016632-x64.psf
  + - 1. 累积更新：合并
         1. 前往 <https://github.com/abbodi1406/WHD/tree/master/scripts> 后下载[PSFX\_Repack\_6.zip](https://github.com/abbodi1406/WHD/raw/master/scripts/PSFX_Repack_6.zip)
         2. 下载后解压到：D:\PSFX\_Repack\_6
         3. 复制 Windows11.0-KB5016632-x64.cab、Windows11.0-KB5016632-x64.psf 到：D:\PSFX\_Repack\_6
         4. 运行：psfx2cab\_GUI.cmd 后，将得到一个新的文件：

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

* + - * 1. 将 Windows11.0-KB5016632-x64-full\_psfx.cab 保存到：

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

* + 1. 添加

$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

###### 其它版本

* + 1. 如何下载

查阅“[Windows 11 版本信息](https://learn.microsoft.com/en-us/windows/release-health/windows11-release-information)”，例如下载累积更新：KB5030219，前往下载页面：<https://www.catalog.update.microsoft.com/Search.aspx?q=KB5030219> 或 "[直连下载](https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/b82b3f6d-17e2-46d5-998a-893deb622e0a/public/windows11.0-kb5030219-x64_659d407821da654d564a3108bf493594bd143ab1.msu)"，保存到：

D:\OS\_11\_Custom\Install\Install\Update\windoOS\_11.0-kb5030219-x64\_659d407821da654d564a3108bf493594bd143ab1.msu

* + 1. 添加

$KBPath = "D:\OS\_11\_Custom\Install\Install\Update\windoOS\_11.0-kb5030219-x64\_659d407821da654d564a3108bf493594bd143ab1.msu"

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

###### 固化更新，可选项

固化后不可卸载，固化时将清理恢复映像并重置任何被取代的组件的基础。

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

* + 1. 固化更新后清理组件，可选项

$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

}

}

##### 驱动，可选

##### 部署引擎：添加，可选

* 了解“[部署引擎](#_部署引擎)”，如果添加到 ISO 安装介质，可跳过添加到已挂载；
* 如果添加部署引擎到已挂载里，请继续在当前位置执行下一步。

##### 健康

保存前应检查是否损坏，健康状态异常时，中止保存

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

##### 替换 WinRE.wim

已批量替换 Install.wim 里的所有索引号里的 WinRE.wim 请跳过该步骤。

$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

##### 保存映像

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

##### 卸载映像

关闭所有可能正在访问映像中文件的应用程序，包括文件资源管理器。

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

循环操作区域，结束。

##### 重建 Install.wim 后可缩小文件大小

* Install.Rebuild.wim.ps1
  + [\Expand\Install\Install.Rebuild.wim.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Install\Install.Rebuild.wim.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Install/Install.Rebuild.wim.ps1>
* 复制代码

$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

}

##### [如何批量替换 Install.wim 里的所有索引号里的 WinRE.wim](#_3.1__)

###### 获取 WimLib

前往 <https://wimlib.net> 官方网站后，选择不同的版本：[arm64](https://wimlib.net/downloads/wimlib-1.14.3-windows-aarch64-bin.zip), [x64](https://wimlib.net/downloads/wimlib-1.14.3-windows-aarch64-bin.zip), [x86](https://wimlib.net/downloads/wimlib-1.14.3-windows-i686-bin.zip)，下载完成后解压到：D:\Wimlib

###### 如何在 Install.wim 里提取和更新 WinRE.wim

* + 1. 从 Install.wim 里提取 WinRE.wim 文件Install.wim
* Install.WinRE.Replace.wim.ps1
  + [\Expand\Install\Install.WinRE.Replace.wim.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Install\Install.WinRE.Replace.wim.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Install/Install.WinRE.Replace.wim.ps1>
* 复制代码

$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. 获取 Install.wim 所有索引号后并替换旧的 WinRE.wim
* Install.WinRE.Replace.wim.ps1
  + [\Expand\Install\Install.WinRE.Replace.wim.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Install\Install.WinRE.Replace.wim.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Install/Install.WinRE.Replace.wim.ps1>
* 复制代码

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

}

#### 自定义部署映像：boot.wim

##### 查看 Boot.wim 文件信息

映像名称、映像描述、映像大小、架构、版本、索引号等；

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

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

##### 指定挂载 Boot.wim 路径

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

##### 开始挂载 Boot.wim

默认索引号：2

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

##### 语言包

自动安装语言包：获取“组件：映像中已安装的所有包”后进行匹配，匹配到对应的名称后，再安装本地对应的语言包文件。

###### 语言包：添加

* Boot.Instl.lang.ps1
  + [\Expand\Boot\Boot.Instl.lang.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Boot\Boot.Instl.lang.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Boot/Boot.Instl.lang.ps1>
* 复制代码

$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

}

}

}

###### 脱机映像语言：更改

* + 1. 更改默认语言、区域设置和其他国际设置

区域标记：zh-CN

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

* + 1. 查看可用的语言设置

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

###### 语言包：删除，可选

* 添加语言后，如果要部署到非英语区域，可通过删除英语语言组件来节省空间。 删除语言时，请按与添加语言组件相反的顺序卸载语言组件。
* 添加中文后，反向删除“英语 - 美国”，区域：en-US，需提前提取语言包
* Boot.Del.Specified.lang.Tag.ps1
  + [\Expand\Boot\Boot.Del.Specified.lang.Tag.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Boot\Boot.Del.Specified.lang.Tag.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Boot/Boot/Boot.Del.Specified.lang.Tag.ps1>
* 复制代码

$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

}

}

}

###### 组件：映像中已安装的所有包

* + 1. 查看

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

* + 1. 导出到 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

###### 语言包：修复

* + 1. 提取

打开：D:\OS\_11\_Custom\Install\Install\Language\Add\zh-CN\Microsoft-Windows-Client-Language-Pack\_x64\_zh-CN.cab，进入目录：Setup\sources\zh-cn\cli，复制以下文件到桌面：

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

将提取的文件复制到：D:\OS\_11\_Custom\Boot\Boot\Mount\sources\zh-CN

###### 语言包：同步到 ISO 安装程序

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

###### 重新生成 Lang.ini

重新生成后，可调整“安装界面”，选择“语言”时的顺序，打开 lang.ini，默认首选值 = 3，非默认值 = 2。

* + 1. 重新生成已挂载目录 lang.ini

重新生成的Lang.ini 文件位置：D:\en-us\_windows\_server\_2022\_x64\_dvd\_620d7eac\_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. 重新生成 lang.ini 后，同步到安装程序

重新生成的Lang.ini 文件位置：D:\en-us\_windows\_server\_2022\_x64\_dvd\_620d7eac\Sources\lang.ini

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

##### 累积更新，可选

准备可用的累积更新文件，请更改示例文件名：KB\_Boot.cab

###### 添加

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

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

###### 删除

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

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

###### 固化更新，可选项

固化后不可卸载，固化时将清理恢复映像并重置任何被取代的组件的基础。

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

* + 1. 固化更新后清理组件，可选项

$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

}

}

##### 驱动，可选

##### [其它](#_自定义部署映像：boot.wim_1)

###### 绕过 TPM 安装时检查

* Boot.Bypass.TPM.ps1
  + [\Expand\Boot\Boot.Bypass.TPM.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\Boot\Boot.Bypass.TPM.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/Boot/Boot.Bypass.TPM.ps1>
* 复制代码

$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

##### 保存映像

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

##### 卸载映像

关闭所有可能正在访问映像中文件的应用程序，包括文件资源管理器。

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

#### 部署引擎

* + - * 了解“自动添加 Windows 系统已安装的语言”，学习：<https://github.com/ilikeyi/Multilingual>，如何下载：
      * 进入网站后，点击 “代码”，“下载压缩包”，下载完成后得到 main.zip 压缩包文件。
      * 前往 <https://github.com/ilikeyi/Multilingual/releases> 下载页面，选择可用版本：1.1.0.4，选择下载源代码格式：zip，下载完成后得到 Multilingual-1.1.0.4.zip 压缩包文件；
      * 将已下载的 main.zip 或 Multilingual-1.1.0.4.zip，解压到：D:\Multilingual-1.1.0.4，重命名：D:\Multilingual
      * 学习“[无人值守 Windows 安装参考](https://learn.microsoft.com/en-us/windows-hardware/customize/desktop/unattend/)”，通过无人值守来干预安装过程。

##### 添加方式

###### 添加到 ISO 安装介质

* + 1. 无人值守
       1. 添加到：[ISO]:\Autounattend.xml

引导 ISO 安装时，Autounattend.xml 干预 WinPE 安装程序。

复制 D:\Multilingual\Unattend\Mul.Unattend.xml 到 D:\OS\_11\Autounattend.xml

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

* + - 1. 添加到：[ISO]:\Sources\Unattend.xml

挂载或解压 ISO 时，运行 [ISO]:\Setup.exe 安装程序后，[ISO]:\Sources\Unattend.xml 将干预安装过程。

复制 D:\Multilingual\Unattend\Mul.Unattend.xml 到 D:\OS\_11\Sources\Unattend.xml

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

* + - 1. 添加到：[ISO]:\sources\$OEM$\$$\Panther\unattend.xml

安装过程中复制到系统盘里，复制到：{系统盘}:\Windows\Panther\unattend.xml

* + - * 1. 创建 $OEM$ 路径

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

* + - * 1. 复制

复制 D:\Multilingual\Unattend\Mul.Unattend.xml 到 D:\OS\_11\Sources\$OEM$\Panther\Unattend.xml

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

* + 1. 部署引擎：添加

添加“自动添加 Windows 系统已安装的语言”到 D:\OS\_11\sources\$OEM$\$1\Yi\Engine 目录里。

* + - 1. 部署引擎：复制

复制 D:\Multilingual\Engine 到 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. 部署引擎：自定义部署标记

$Flag = @(

"Is\_Mark\_Sync" # 允许全盘搜索并同步部署标记

# 先决部署

# "Auto\_Update" # 允许自动更新

# "Use\_UTF8" # Beta 版：使用 Unicode UTF-8 提供全球语言支持

"Disable\_Network\_Location\_Wizard" # 网络位置向导

"Disable\_Cleanup\_Appx\_Tasks" # Appx 清理维护任务

"Disable\_Cleanup\_On\_Demand\_Language" # 阻止清理未使用的按需功能语言包

"Disable\_Cleanup\_Unsed\_Language" # 阻止清理未使用的语言包

"Prerequisites\_Reboot" # 重新启动计算机

# 完成首次部署

# "Popup\_Engine" # 允许首次弹出部署引擎主界面

# "Allow\_First\_Pre\_Experience" # 允许首次预体验，按计划

"Reset\_Execution\_Policy" # 恢复 PowerShell 执行策略：受限

"Clear\_Solutions" # 删除整个解决方案

"Clear\_Engine" # 删除部署引擎，保留其它

# "First\_Experience\_Reboot" # 重新启动计算机

)

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

}

###### 添加到已挂载

通过“自定义部署映像：Install.wim”，执行“开始挂载 Install.wim”，挂载到：D:\OS\_11\_Custom\Install\Install\Mount

* + 1. 无人值守

复制 D:\Multilingual\Unattend\Mul.Unattend.xml 到 D:\OS\_11\_Custom\Install\Install\Mount\Panther\Unattend.xml

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

* + 1. 部署引擎

添加“自动添加 Windows 系统已安装的语言”到 D:\OS\_11\_Custom\Install\Install\Mount\Yi\Engine 目录里。

* + - 1. 部署引擎：复制

复制 D:\Multilingual\Engine 到 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. 部署引擎：自定义部署标记

$Flag = @(

"Is\_Mark\_Sync" # 允许全盘搜索并同步部署标记

# 先决部署

# "Auto\_Update" # 允许自动更新

# "Use\_UTF8" # Beta 版：使用 Unicode UTF-8 提供全球语言支持

"Disable\_Network\_Location\_Wizard" # 网络位置向导

"Disable\_Cleanup\_Appx\_Tasks" # Appx 清理维护任务

"Disable\_Cleanup\_On\_Demand\_Language" # 阻止清理未使用的按需功能语言包

"Disable\_Cleanup\_Unsed\_Language" # 阻止清理未使用的语言包

"Prerequisites\_Reboot" # 重新启动计算机

# 完成首次部署

# "Popup\_Engine" # 允许首次弹出部署引擎主界面

# "Allow\_First\_Pre\_Experience" # 允许首次预体验，按计划

"Reset\_Execution\_Policy" # 恢复 PowerShell 执行策略：受限

"Clear\_Solutions" # 删除整个解决方案

"Clear\_Engine" # 删除部署引擎，保留其它

# "First\_Experience\_Reboot" # 重新启动计算机

)

ForEach ($item in $Flag) {

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

New-Item -Path "D:\OS\_11\_Custom\Install\Install\Mount\Yi\Engine\Deploy\Allow" -ItemType Directory -ErrorAction SilentlyContinue | Out-Null

Out-File -FilePath "D:\OS\_11\_Custom\Install\Install\Mount\Yi\Engine\Deploy\Allow\$($item)" -Encoding utf8 -ErrorAction SilentlyContinue

}

##### 部署引擎：进阶

###### 部署引擎：添加过程中

在复制部署引擎后，可添加部署标记来干预安装过程。

###### 无人值守方案

自定义无人值守时，以下文件存在时请同步修改：

* 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. 多语言或单语

多语言时，单语时，可互相切换，替换时，请替换文件里所有相同的。

* + - 1. 多语言

<UILanguage>%OSDUILanguage%</UILanguage>

<InputLocale>%OSDInputLocale%</InputLocale>

<SystemLocale>%OSDSystemLocale%</SystemLocale>

<UILanguage>%OSDUILanguage%</UILanguage>

<UILanguageFallback>%OSDUILanguageFallback%</UILanguageFallback>

<UserLocale>%OSDUserLocale%</UserLocale>

* + - 1. 单语

单语需指定区域，例如指定区域：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. 用户方案

默认使用自建用户Administrator 并自动登录，可通过修改以下配置切换：自建、自定义用户。

* + - 1. 自建用户 Administrator

默认使用自建用户：Administrator 并自动登录，插入到 <OOBE> 和 </OOBE> 之间。

<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. 自定义用户

设置自定义用户后，安装系统完成后，在 OOBE 里，可选择本地、在线用户等设置。

* + - * 1. 删除

用户名：从开始处删除 <UserAccounts> 到 </UserAccounts>

自动登录：从开始处删除 <AutoLogon> 到 </AutoLogon>

* + - * 1. 替换

从开始处 <OOBE> 到 </OOBE>

<OOBE>

<ProtectYourPC>3</ProtectYourPC>

<HideEULAPage>true</HideEULAPage>

<HideWirelessSetupInOOBE>true</HideWirelessSetupInOOBE>

</OOBE>

### ISO

#### 生成 ISO

1. 下载 oscdimg 后，保存到：C:\Windows\System32 目录里，保存在其它路径，或请输入 OScdimg.exe 绝对位置；
2. 使用 oscdimg命令行生成一个 ISO 文件，保存到：D:\Win11.iso

* ISO.ps1
  + [\Expand\ISO.ps1](file:///D:\Yi.Solutions\_Documents\Attachment\OS.11\22H2\Expand\ISO.ps1)
  + <https://github.com/ilikeyi/solutions/blob/main/_Documents/Attachment/OS.11/22H2/Expand/ISO.ps1>
* 复制代码

$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.exe" -ArgumentList $Arguments -wait -nonewwindow

#### 绕过 TPM 安装检查，可选

1. 了解 <https://github.com/AveYo/MediaCreationTool.bat/tree/main/bypass11> 并下载：Quick\_11\_iso\_esd\_wim\_TPM\_toggle.bat
2. 将 D:\OS\_11.iso 文件托到 Quick\_11\_iso\_esd\_wim\_TPM\_toggle.bat 里，反序“添加”或“删除”TPM 安装时检查功能。

## 常见问题

### 清理所有挂载到

关闭所有可能正在访问映像中文件的应用程序，包括文件资源管理器。

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

### 修复挂载出现异常的问题

1. 查看已挂载

Get-WindowsImage -Mounted

1. 删除保存在注册表里的 DISM 挂载记录

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

1. 删除与已损坏的已装载映像关联的所有资源。

Clear-WindowsCorruptMountPoint

Dism /cleanup-wim

### 清理目录

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

## 已知问题

暂无

## 组件：映像中已安装的所有包，对比

安装新的语言包后变化，组件列表：

| 序号 | 类型 | 组件：原始 | 安装“简体中文 - 中国”语言包后变化 |
| --- | --- | --- | --- |
|  | OnDemandPack | Microsoft-OneCore-ApplicationModel-Sync-Desktop-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.317 | Microsoft-OneCore-ApplicationModel-Sync-Desktop-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.317 |
|  | OnDemandPack | Microsoft-OneCore-DirectX-Database-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-OneCore-DirectX-Database-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | LanguagePack | Microsoft-Windows-Client-LanguagePack-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.382 | Microsoft-Windows-Client-LanguagePack-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.382 |
|  | LanguagePack |  | Microsoft-Windows-Client-LanguagePack-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.382 |
|  | OnDemandPack | Microsoft-Windows-Ethernet-Client-Intel-E1i68x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Ethernet-Client-Intel-E1i68x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Ethernet-Client-Intel-E2f68-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Ethernet-Client-Intel-E2f68-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Ethernet-Client-Realtek-Rtcx21x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Ethernet-Client-Realtek-Rtcx21x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Ethernet-Client-Vmware-Vmxnet3-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Ethernet-Client-Vmware-Vmxnet3-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | FeaturePack | Microsoft-Windows-FodMetadata-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-FodMetadata-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | Foundation | Microsoft-Windows-Foundation-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Foundation-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Hello-Face-Package~31bf3856ad364e35~amd64~~10.0.22621.317 | Microsoft-Windows-Hello-Face-Package~31bf3856ad364e35~amd64~~10.0.22621.317 |
|  | OnDemandPack | Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~en-US~11.0.22621.1 | Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~en-US~11.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~zh-CN~11.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~~11.0.22621.1 | Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~~11.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Kernel-LA57-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.382 | Microsoft-Windows-Kernel-LA57-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.382 |
|  | OnDemandPack | Microsoft-Windows-LanguageFeatures-Basic-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-LanguageFeatures-Basic-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-LanguageFeatures-Basic-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-LanguageFeatures-Fonts-Hans-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-LanguageFeatures-Handwriting-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-LanguageFeatures-Handwriting-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-LanguageFeatures-Handwriting-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.317 |
|  | OnDemandPack | Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-LanguageFeatures-OCR-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-LanguageFeatures-Speech-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-LanguageFeatures-Speech-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-LanguageFeatures-Speech-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-LanguageFeatures-TextToSpeech-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-LanguageFeatures-TextToSpeech-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-LanguageFeatures-TextToSpeech-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.317 | Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.317 |
|  | OnDemandPack |  | Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.317 |
|  | OnDemandPack | Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~~10.0.22621.382 | Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~~10.0.22621.382 |
|  | OnDemandPack | Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 | Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~~10.0.22621.1 | Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 | Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 | Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1 | Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 | Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 | Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~~10.0.22621.1 | Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 | Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 | Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 | Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~~10.0.22621.1 | Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-TabletPCMath-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-TabletPCMath-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wallpaper-Content-Extended-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wallpaper-Content-Extended-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Broadcom-Bcmpciedhd63-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Broadcom-Bcmpciedhd63-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Broadcom-Bcmwl63a-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Broadcom-Bcmwl63a-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Broadcom-Bcmwl63al-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Broadcom-Bcmwl63al-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Intel-Netwbw02-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Intel-Netwbw02-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Intel-Netwew00-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Intel-Netwew00-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Intel-Netwew01-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Intel-Netwew01-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Intel-Netwlv64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Intel-Netwlv64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Intel-Netwns64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Intel-Netwns64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Intel-Netwsw00-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Intel-Netwsw00-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Intel-Netwtw02-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Intel-Netwtw02-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Intel-Netwtw04-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Intel-Netwtw04-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Intel-Netwtw06-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Intel-Netwtw06-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Intel-Netwtw08-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Intel-Netwtw08-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Intel-Netwtw10-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Intel-Netwtw10-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Marvel-Mrvlpcie8897-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Marvel-Mrvlpcie8897-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Qualcomm-Athw8x-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Qualcomm-Athw8x-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Qualcomm-Athwnx-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Qualcomm-Athwnx-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Qualcomm-Qcamain10x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Qualcomm-Qcamain10x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Ralink-Netr28x-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Ralink-Netr28x-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Realtek-Rtl8187se-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Realtek-Rtl8187se-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Realtek-Rtl8192se-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Realtek-Rtl8192se-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Realtek-Rtl819xp-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Realtek-Rtl819xp-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Realtek-Rtl85n64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Realtek-Rtl85n64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Realtek-Rtwlane-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Realtek-Rtwlane-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Realtek-Rtwlane01-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Realtek-Rtwlane01-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-Wifi-Client-Realtek-Rtwlane13-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | Microsoft-Windows-Wifi-Client-Realtek-Rtwlane13-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 | Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.382 | Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.382 |
|  | OnDemandPack | Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 | Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1 | Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 | Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.317 | Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.317 |
|  | OnDemandPack | Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 | Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1 |
|  | OnDemandPack |  | Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1 |
|  | OnDemandPack | Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1 | Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1 |
|  | OnDemandPack | OpenSSH-Client-Package~31bf3856ad364e35~amd64~~10.0.22621.1 | OpenSSH-Client-Package~31bf3856ad364e35~amd64~~10.0.22621.1 |
|  | Update | Package\_for\_DotNetRollup\_481~31bf3856ad364e35~amd64~~10.0.9065.6 | Package\_for\_DotNetRollup\_481~31bf3856ad364e35~amd64~~10.0.9065.6 |
|  | SecurityUpdate | Package\_for\_RollupFix~31bf3856ad364e35~amd64~~22621.382.1.13 | Package\_for\_RollupFix~31bf3856ad364e35~amd64~~22621.382.1.13 |
|  | SecurityUpdate | Package\_for\_ServicingStack\_378~31bf3856ad364e35~amd64~~22621.378.1.0 | Package\_for\_ServicingStack\_378~31bf3856ad364e35~amd64~~22621.378.1.0 |

|  |  |
| --- | --- |
| A logo of a cube  Description automatically generated | Yi  Yi’s Solutions |

© Yi. All rights reserved. Design: Yi