



MICROSOFT WINDOWS 11 22H2

封装教程：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 Apps：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？

摘要

- 章节 1 封装
- 章节 2 报告
- 章节 3 常见问题
- 章节 4 已知问题

章节 1	封装	第 6 页
A.	先决条件	第 6 页
II	正在运行的操作系统	第 6 页
III	ISO 工具	第 6 页
IV	要求	第 6 页
1.	系统安装包	第 6 页
2.	语言包	第 8 页
2.1.	学习	第 8 页
2.2.	语言包：下载	第 10 页
2.3.	语言包：修复	第 10 页
3.	InBox Apps	第 11 页
V	Windows 安全中心	第 11 页
VI	命令行	第 12 页
B.	语言包：提取	第 12 页
II	语言包：准备	第 12 页
III	语言包：提取方案	第 12 页
IV	执行提取命令	第 13 页
C.	自定义封装	第 19 页
II	自定义封装：Install.wim	第 19 页
1.	查看 Install.wim 详细信息	第 19 页
2.	指定挂载 Install 路径	第 19 页
3.	开始挂载 Install.wim	第 19 页
3.1.	自定义封装：WinRE.wim	第 19 页
3.1.1.	查看 WinRE.wim 详细信息	第 19 页

3.1.2.	指定挂载 WinRE.wim 路径	第 19 页
3.1.3.	开始挂载 WinRE.wim	第 20 页
3.1.4.	语言包	第 20 页
3.1.4.1.	语言包：添加	第 20 页
3.1.4.2.	脱机映像语言：更改	第 21 页
3.1.4.2.1.	更改默认语言、区域设置和其他国际设置	第 22 页
3.1.4.2.2.	查看可用的语言设置	第 22 页
3.1.4.3.	语言包：删除	第 22 页
3.1.4.4.	组件：映像中已安装的所有包	第 24 页
3.1.5.	累积更新	第 24 页
3.1.5.1.	添加	第 24 页
3.1.5.2.	删除	第 25 页
3.1.5.3.	固化更新	第 25 页
3.1.5.3.1.	固化更新后清理组件	第 25 页
3.1.6.	驱动	第 25 页
3.1.7.	保存映像：WinRE.wim	第 25 页
3.1.8.	卸载映像：WinRE.wim	第 25 页
3.1.9.	重建 WinRE.wim 后，可缩小文件大小	第 25 页
3.1.10.	备份 WinRE.wim	第 26 页
3.1.11.	替换 Install.wim 映像内的 WinRE.wim	第 26 页
4.	语言包	第 27 页
4.1.	语言包：添加	第 27 页
4.2.	脱机映像语言：更改	第 29 页
4.2.1.	更改默认语言、区域设置和其他国际设置	第 29 页
4.2.2.	查看可用的语言设置	第 29 页

4.3.	语言包：删除	第 29 页
4.4.	组件：映像中已安装的所有包	第 32 页
5.	InBox Apps	第 32 页
5.1.	学习	第 32 页
5.2.	InBox Apps：预安装应用程序	第 32 页
5.3.	InBox Apps：已安装	第 34 页
5.4.	删除已安装的所有预应用程序	第 34 页
5.5.	区域标记：了解	第 35 页
5.6.	区域标记：添加	第 35 页
5.7.	InBox Apps：安装	第 38 页
5.8.	区域标记：删除	第 47 页
5.9.	InBox Apps：优化	第 48 页
6.	累积更新	第 48 页
6.1.	初始版本	第 48 页
6.2.	其它版本	第 49 页
6.3.	固化更新	第 50 页
6.3.1.	固化更新后清理组件	第 50 页
7.	驱动	第 50 页
8.	部署引擎：添加	第 50 页
9.	健康	第 50 页
10.	替换 WinRE.wim	第 51 页
11.	保存映像：Install.wim	第 51 页
12.	卸载映像：Install.wim	第 51 页
13.	如何批量替换 Install.wim 里的所有索引号里的 WinRE.wim	第 51 页
13.1.	获取 WimLib	第 51 页
13.2.	如何在 Install.wim 里提取和更新 WinRE.wim	第 51 页

14.	重建 Install.wim 后可缩小文件大小	第 52 页
III	自定义封装: boot.wim	第 53 页
1.	查看 Boot.wim 文件信息	第 53 页
2.	指定挂载 Boot.wim 路径	第 53 页
3.	开始挂载 Boot.wim	第 53 页
4.	语言包	第 53 页
4.1.	语言包: 添加	第 54 页
4.2.	脱机映像语言: 更改	第 55 页
4.2.1.	更改默认语言、区域设置和其他国际设置	第 55 页
4.2.2.	查看可用的语言设置	第 55 页
4.3.	语言包: 删除	第 55 页
4.4.	组件: 映像中已安装的所有包	第 58 页
4.5.	语言包: 修复	第 58 页
4.6.	语言包: 同步到 ISO 安装程序	第 58 页
4.7.	重新生成 Lang.ini	第 58 页
4.7.1.	重新生成已挂载目录 lang.ini	第 58 页
4.7.2.	重新生成 lang.ini 后, 同步到安装程序	第 59 页
5.	累积更新	第 59 页
5.1.	添加	第 59 页
5.2.	删除	第 59 页
5.3.	固化更新	第 59 页
5.3.1.	固化更新后清理组件	第 59 页
6.	驱动	第 59 页
7.	其它	第 59 页
7.1.	绕过 TPM 安装时检查	第 59 页

8.	保存映像：Boot.wim	第 60 页
9.	卸载映像：Boot.wim	第 60 页
IV	部署引擎	第 60 页
1.	添加方式	第 61 页
2.	部署引擎：进阶	第 64 页
D.	ISO	第 66 页
II	生成 ISO	第 66 页
III	绕过 TPM 安装检查	第 67 页
章节 2	报告	第 67 页
II	不同版本	第 67 页
III	语言安装包适用于：Install.wim	第 70 页
IV	语言安装包适用于：WinRE.wim	第 71 页
V	语言安装包适用于：Boot.wim	第 72 页
章节 3	常见问题	第 73 页
II	清理所有挂载到	第 73 页
III	修复挂载出现异常的问题	第 74 页
IV	清理目录	第 74 页
章节 4	已知问题	第 74 页

章节 1 封装

A. 先决条件

II 正在运行的操作系统

正在运行的操作系统是 Windows 10 或者低于 Windows 11 24H2 时，在某些情况下使用 DISM 命令在制作高版本镜像时，会引发一些未知的问题，例如：在 Windows 10 操作系统里运行 DISM 命令来处理 Windows Server 2025 脱机映像时，在封装过程中也许会收到报错信息：“此应用无法在你的电脑运行”，解决方法：

- 1. 升级正在运行的操作系统或重新安装到更高的版本（建议）；
- 2. 升级或安装新版 ADK 或 PowerShell 版本（不建议）
 - 2.1. 可尝试升级到最新 PowerShell 7 以上的版本；
 - 2.2. 安装最新版 ADK 后并替换 DISM 命令后能解决 DISM 版本低的问题，但是：封装脚本主要使用的命令行是 PowerShell 命令行，所以不建议你使用以上方法，最佳的方法是：升级正在运行的操作系统或重新安装到更高的版本。

III ISO 工具

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

IV 要求

1. 系统安装包

关键词：[迭代](#)、[跨版本](#)、[大版本](#)、[累积更新](#)、[初始版本](#)

1.1. 说明

- 1.1.1. 每版本更新时请重新制作镜像，例如从 21H1 跨越到 22H2 时，应避免出现其它兼容性问题请勿在旧镜像基础上制作；
- 1.1.2. 该条例已经在某些 OEM 厂商，通过各种形式向封装师明确传达了该法令，不允许直接从迭代版本里直接升级；
- 1.1.3. 制作中请使用“初始版本”、“开发者版”制作。微软官方文档里曾短暂的出现过在制作中必须使用初始版本，后来这句在官方文档里却被删除了；
- 1.1.4. 安装语言包后，必须重新添加累积更新（可同一版本号），不添加累积更新会出现“乱码”、“界面闪退”等问题。
- 1.1.5. 进化过程：Windows 11 22H2, Build 22261.382 + KB5027303 = OS Build 22621.1928

1.2. 准备下载初始版本或开发者版本

- 1.2.1. x64
 - 1.2.1.1. [en-us_windows_11_business_editions_version_22h2_x64_dvd_17a08ce3.iso](#)

- 1.2.1.2.

en-us_windows_11_consumer_editions_version_22h2_x64_dvd_e630fafd.iso
- 1.2.1.3.

en-us_windows_11_iot_enterprise_version_22h2_x64_dvd_eb39bf70.iso
- 1.2.2.

arm64

1.2.2.1.

en-us_windows_11_iot_enterprise_version_22h2_arm64_dvd_f09f5d29.iso
- 1.3.

示例下载 en-us_windows_11_business_editions_version_22h2_x64_dvd_17a08ce3.iso 后，解压到：D:\en-us_windows_11_business_editions_version_22h2_x64_dvd_17a08c

注意：解压到 D 盘前，你应该检查是否 ReFS 分区格式，如果是 ReFS 分区格式时：执行部分命令将出现异常。解决方法：请使用 NTFS 格式的磁盘分区。
- 1.4.

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
- 1.5.

所有脚本、所有路径，已默认设置为 D:\OS_11 为映像来源。
- 1.6.

安装配置

1.6.1.

学习：[Windows 安装版配置和产品 ID 文件（EI.cfg 和 PID.txt）](#)

1.6.2.

已知问题

1.6.2.1.

没有 Ei.cfg 时，ISO 引导安装，选择某些版本时会报错，提示：Windows 找不到 Microsoft 软件许可条款。
请确保安装源有效。然后重新启动安装。

1.6.2.2.

如何解决，添加 ei.cfg 到 D:\OS_11\Sources 里，创建 ei.cfg：

@"
[Channel]
volume

[VL]
1
"@ | Out-File -FilePath "D:\OS_11\sources\EI.CFG" -Encoding Ascii

1.7.

报告

查看映像包里已安装：语言包、已关联的语言包组件、预安装 InBox Apps 应用程序等详细信息，下载报告后，使用 Excel 软件打开 .xlsx。

1.7.1.

报告：总览

包含：ISO 系统安装包、映像版本、区域、区域关联、已关联的语言包组件等。

[Report.Overview.xlsx](#)

1.7.2. 报告：完整

包容：ISO 系统安装包、映像版本、区域、区域关联、已关联的语言包组件、已安装 InBox Apps 预安装应用程序等。

en-us、ar-sa、bg-bg、cs-cz、da-dk、de-de、el-gr、en-gb、es-es、es-mx、et-ee、fi-fi、fr-ca、fr-fr、he-il、hr-hr、hu-hu、it-it、ja-jp、ko-kr、lt-lt、lv-lv、nb-no、nl-nl、pl-pl、pt-br、pt-pt、ro-ro、ru-ru、sk-sk、sl-si、sr-latn-rs、sv-se、th-th、tr-tr、uk-ua、zh-cn、zh-tw

2. 语言包

2.1. 学习

阅读时，请了解“蓝色”重要突出部分。

2.1.1. 语言概述

2.1.2. 将语言添加到 Windows 11 映像

2.1.3. 语言和区域按需功能 (FOD)

2.1.3.1. 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”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

2.1.3.2. 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。

已知区域关联：

2.1.3.2.1. 区域：ar-sa，可选关联区域：en-US，关联安装包：

- | | |
|--------------|---|
| 2.1.3.2.1.1. | Microsoft-Windows-LanguageFeatures-Basic-es-us-Package~31bf3856ad364e35~amd64~~.cab |
| 2.1.3.2.1.2. | Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~.cab |

2.1.3.2.2. 区域：bg-bg，可选关联区域：en-US，关联安装包：

- | | |
|--------------|---|
| 2.1.3.2.2.1. | Microsoft-Windows-LanguageFeatures-Basic-es-us-Package~31bf3856ad364e35~amd64~~.cab |
| 2.1.3.2.2.2. | Microsoft-Windows-LanguageFeatures-OCR-en-us- |

Package~31bf3856ad364e35~amd64~~.cab

2.1.3.2.3. 区域: [da-dk](#), 可选关联区域: [en-US](#), 关联安装包:

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

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

2.1.3.2.4. 区域: [el-gr](#), 可选关联区域: [en-US](#), 关联安装包:

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

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

2.1.3.2.5. 区域: [fr-ca](#), 可选关联区域: [en-US](#)、[fr-fr](#), 关联安装包:

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

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

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

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

2.1.3.2.6. 区域: [he-il](#), 可选关联区域: [en-US](#), 关联安装包:

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

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

2.1.3.2.7. 区域: [ru-ru](#), 可选关联区域: [en-US](#), 关联安装包:

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

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

2.1.3.2.8. 区域: [th-th](#), 可选关联区域: [en-US](#), 关联安装包:

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

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

2.1.3.2.9. 区域：uk-ua，可选关联区域：en-US，关联安装包：

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

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

2.1.3.2.10. 区域：zh-TW，可选关联区域：zh-HK，关联安装包：

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

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

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

2.1.3.3. 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”。

2.1.3.3.1. 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：

- 查看“报告：Windows 11 23H2”脱机映像语言列表，并未发现有此程序包；
- 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。

2.2. 语言包：下载

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

2.3. 语言包：修复

2.3.1. 任选一个网站并打开：

- 2.3.1.1. <https://uupdump.net>
- 2.3.1.2. <https://uup.ee>
- 2.3.1.3. <https://osdump.com>

2.3.2. 打开后，搜索关键词：22621.382，在搜索结果选择：Windows 11, version 22H2 (22621.382) amd64

2.3.3. 打开后，选择“全部文件”；

2.3.4. 在“全部文件”页面里依次搜索绿色部分并下载：

2.3.4.1. 适用于：Install.wim（1项）

2.3.4.1.1. MediaPlayer

2.3.4.2. 适用于：WinRE.wim，暂无

2.3.4.3. 适用于：Boot.wim，暂无

2.3.5. 下载所有文件后，拉到到页面下方，下载并运行“生成重命名脚本（Windows）”；

2.3.6. 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；

3. InBox Apps

3.1. 下载：https://software-static.download.prss.microsoft.com/dbazure/888969d5-f34g-4e03-ac9d-1f9786c66749/22621.1778.230511-2102.ni_release_svc_prod3_amd64fre_InboxApps.iso

3.2. 下载：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：

3.2.1. Microsoft.HEVCVideoExtension

- 3.2.1.1. Microsoft.HEVCVideoExtension_8wekyb3d8bbwe.arm64.appx
- 3.2.1.2. Microsoft.HEVCVideoExtension_8wekyb3d8bbwe.arm64.xml
- 3.2.1.3. Microsoft.HEVCVideoExtension_8wekyb3d8bbwe.x64.appx
- 3.2.1.4. Microsoft.HEVCVideoExtension_8wekyb3d8bbwe.x64.xml

3.3. 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；

V Windows 安全中心

- 在处理封装任务时，将产生大量的临时文件，安装 InBox Apps 里的应用时会释放大量的安装文件；

- 开启 Windows 安全中心会扫描文件、会占用大量的 CPU。
- 测试中：未关闭前耗时 1 小时 22 分，关闭后耗时 20 分钟。

如何关闭：

绿色为命令行，按住 Windows 键并按 R 键启动“运行”。

1. 打开“Windows 安全中心”或运行： `windowsdefender:`
2. 选择“病毒和威胁防护”或运行： `windowsdefender://threat`
3. 找到“病毒和威胁防护设置”，点击“管理设置”或运行： `windowsdefender://threatsettings`，建议您关闭部分功能：
 - 3.1. 实时保护
 - 3.2. 云提供的保护
 - 3.3. 自动提交样本
 - 3.4. 篡改防护
4. 未处于封装时，建议您开启 Windows 安全中心。

VI 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`
3. 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；
4. 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，带冒号的路径，在命令行添加 & 字符，示例： `& "D:\YiSolutions\Encapsulation\SIP.ps1"`

B. 语言包：提取

II 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；

III 语言包：提取方案

1. 添加

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

2. 删除

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

IV 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
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging/tutorial/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 = "Syr"; }

@{ 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_OC\WinPE-FontSupport-{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\lp.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-securestartup_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-atbroker_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-audiocore_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-audiodrivers_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-enhancedstorage_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-narrator_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-scripting_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-speech-tts_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-srh_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-srt_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-wds-tools_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-wmi_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-appxdeployment_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-appxpackaging_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-storagewmi_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-wifi_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-windowsupdate_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-rejuv_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-opcservices_{Lang}.cab"

        "Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-hta_{Lang}.cab"

    )

}

@{

    Path = "Boot\Boot"

    Rule = @{
```

```

"Windows Preinstallation Environment\x64\WinPE_OC\WinPE-FontSupport-{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\lp.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\WinPE-Setup_{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\WINPE-SETUP-CLIENT_{Lang}.CAB"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-securestartup_{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-atbroker_{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-audiocore_{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-audiodrivers_{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-enhancedstorage_{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-narrator_{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-scripting_{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-speech-tts_{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-srh_{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-srt_{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{Lang}\winpe-wds-tools_{Lang}.cab"

"Windows Preinstallation Environment\x64\WinPE_OC\{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 }

```

C. 自定义封装

II 自定义封装：Install.wim

1. 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，

2. 指定挂载 Install 路径

```
New-Item -Path "D:\OS_11_Custom\Install\Install\Mount" -ItemType directory -ea SilentlyContinue
```

3. 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，

3.1. 自定义封装：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”。

3.1.1. 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 }
```

3.1.2. 指定挂载 WinRE.wim 路径

```
New-Item -Path "D:\OS_11_Custom\Install\WinRE\Mount" -ItemType directory -ea SilentlyContinue
```

3.1.3. 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"
```

3.1.4. 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，查看报告“[语言安装包适用于 WinRE.wim](#)”。
- 添加语言时，须对应不同的架构版本，未对应时，添加过程中报错等提示。

3.1.4.1. 语言包：添加

- WinRE.Instl.lang.ps1
 - [\Expand\Install\WinRE\WinRE.Instl.lang.ps1](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Instal/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

        }

    }

}

}
```

3.1.4.2. 脱机映像语言：更改

3.1.4.2.1. 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
```

3.1.4.2.2. 查看可用的语言设置

```
Dism /Image:"D:\OS_11_Custom\Install\WinRE\Mount" /Get-Intl
```

3.1.4.3. 语言包: 删除

- 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
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/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

        }

    }

}

}
```

3.1.4.4. 组件：映像中已安装的所有包

3.1.4.4.1. 查看

```
Get-WindowsPackage -Path "D:\OS_11_Custom\Install\WinRE\Mount" | Out-GridView
```

3.1.4.4.2. 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
```

3.1.5. 累积更新

准备可用的累积更新文件，请更改示例文件名： KB_WinRE.cab

3.1.5.1. 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
```

3.1.5.2. 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
```

3.1.5.3. 固化更新

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
```

3.1.5.3.1. 固化更新后清理组件

```
$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
```

```
    }
```

```
}
```

3.1.6. 驱动

3.1.7. 保存映像：WinRE.wim

```
Save-WindowsImage -Path "D:\OS_11_Custom\Install\WinRE\Mount"
```

3.1.8. 卸载映像：WinRE.wim

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
```

3.1.9. 重建 WinRE.wim 后，可缩小文件大小

- [WinRE.Rebuild.ps1](#)
 - [\Expand\Install\WinRE\WinRE.Rebuild.ps1](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/WinRE/WinRE.Rebuild.ps1

- 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

}

}
```

3.1.10. 备份 WinRE.wim

- WinRE.Backup.ps1
 - [\Expand\Install\WinRE\WinRE.Backup.ps1](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/WinRE/WinRE.Backup.ps1

- 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
```

3.1.11. 替换 Install.wim 映像内的 WinRE.wim

- 每次挂载 Install.wim 后“[替换 WinRE.wim](#)”;

- 学习“[获取 Install.wim 所有索引号后并替换旧的 WinRE.wim](#)”。

处理 Install.wim 映像内的文件，结束。

4. 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，查看报告“[语言安装包适用于 Install.wim](#)”。
- 添加语言时，须对应不同的架构版本，未对应时，添加过程中报错等提示。

4.1. 语言包：添加

- Install.Instl.lang.ps1
 - [\Expand\Install\Install.Instl.lang.ps1](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging/tutorial/OS.11/22H2/Expand/Install/Install.Instl.lang.ps1

- 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

}

}

}
```

4.2. 脱机映像语言：更改

- 从 Windows 11 开始，DISM 设置的默认系统 UI 语言在所有版本中保持不变（家庭版除外）。对于所有商业版，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，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。

4.2.1. 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"
```

4.2.2. 查看可用的语言设置

```
Dism /Image:"D:\OS_11_Custom\Install\Install\Mount" /Get-Intl
```

4.3. 语言包：删除

- 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
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging/tutorial/OS.11/22H2/Expand/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

        }

    }

}

```


}

4.4. 组件：映像中已安装的所有包

4.4.1. 查看

```
Get-WindowsPackage -Path "D:\OS_11_Custom\Install\Install\Mount" | Out-GridView
```

4.4.2. 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
```

5. InBox Apps

5.1. 学习

5.1.1. DISM 应用包（.appx 或 .appxbundle）服务命令行选项

5.2. InBox Apps：预安装应用程序

- 编组缩写
 - 群组 A: 教育版, 专业版, 专业教育版, 专业版工作站, 企业版, IoT 企业版, 适用于虚拟桌面的 Windows 10 企业版
 - 群组 B: 企业版 N, 企业版 GN, 企业版 SN, 专业版 N, 教育版 N, 专业版工作站 N, 专业教育版 N
 - 群组 C: 家庭版, 家庭版 N, 家庭单语版

	名称	产品 ID	编组
1.	Microsoft.HEIFImageExtension	9PMMSR1CGPWG	A, C
2.	Microsoft.HEVCVideoExtension	9NMZLZ57R3T7	A, C
3.	Microsoft.VP9VideoExtensions	9N4D0MSMP0PT	A, C
4.	Microsoft.WebpImageExtension	9PG2DK419DRG	A, C
5.	Microsoft.GamingApp	9MV0B5HZVK9Z	A, C
6.	Microsoft.ZuneMusic	9WZDNCRFJ3PT	A, C
7.	MicrosoftCorporationII.MicrosoftFamily	9PDJDS743XF	C
8.	Microsoft.WebMediaExtensions	9N5TDP8VCMHS	A, C
9.	Microsoft.WindowsSoundRecorder	9WZDNCRFHWKN	A, C
10.	Microsoft.Xbox.TCUI	9NKNC0LD5NN6	A, C
11.	Microsoft.ZuneVideo	9WZDNCRFJ3P2	A, C

12.	Microsoft.XboxGamingOverlay	9NZKPSTSNW4P	A, C
13.	Microsoft.RawImageExtension	9NCTDW2W1BH8	A, C
14.	Microsoft.UI.Xaml.2.3		A, B, C
15.	Microsoft.UI.Xaml.2.4		A, B, C
16.	Microsoft.UI.Xaml.2.7		A, B, C
17.	Microsoft.NET.Native.Framework.2.2		A, B, C
18.	Microsoft.NET.Native.Runtime.2.2		A, B, C
19.	Microsoft.VCLibs.140.00		A, B, C
20.	Microsoft.VCLibs.140.00.UWPDesktop		A, B, C
21.	Microsoft.SecHealthUI	9NK0JLZT7JD0	A, B, C
22.	Microsoft.WindowsStore	9WZDNCRFJBMP	A, B, C
23.	Microsoft.Sticky.Notes	9NBLGGH4QGHW	A, B, C
24.	Microsoft.Paint	9PCFS5B6T72H	A, B, C
25.	Microsoft.PowerAutomateDesktop	9NFTCH6J7FHV	A, B, C
26.	Microsoft.ScreenSketch	9MZ95KL8MR0L	A, B, C
27.	Microsoft.WindowsNotepad	9MSMLRH6LZF3	A, B, C
28.	Microsoft.WindowsTerminal	9N0DX20HK701	A, B, C
29.	Clipchamp.Clipchamp	9P1J8S7CCWWT	A, B, C
30.	Microsoft.Solitaire.Collection	9WZDNCRFHWD2	A, B, C
31.	Microsoft.WindowsAlarms	9WZDNCRFJ3PR	A, B, C
32.	Microsoft.WindowsFeedbackHub	9NBLGGH4R32N	A, B, C
33.	Microsoft.WindowsMaps	9WZDNCRDTBVB	A, B, C
34.	Microsoft.BingNews	9WZDNCRFHVFW	A, B, C
35.	Microsoft.DesktopAppInstaller	9NBLGGH4NNS1	A, B, C
36.	Microsoft.WindowsCamera	9WZDNCRFJBGG	A, B, C
37.	Microsoft.Getstarted	9WZDNCRDTBJJ	A, B, C
38.	Microsoft.Cortana	9NFFX4SZZ23L	A, B, C
39.	Microsoft.BingWeather	9WZDNCRFJ3Q2	A, B, C
40.	Microsoft.GetHelp	9PKDZBMV1H3T	A, B, C
41.	Microsoft.MicrosoftOfficeHub	9WZDNCRD29V9	A, B, C
42.	Microsoft.People	9NBLGGH10PG8	A, B, C
43.	Microsoft.StorePurchaseApp	9NBLGGH4LS1F	A, B, C
44.	Microsoft.Todos	9NBLGGH5R558	A, B, C
45.	Microsoft.Windows.Photos	9WZDNCRFJBH4	A, B, C

46.	Microsoft.WindowsCalculator	9WZDNCRFHVN5	A, B, C
47.	Microsoft.Windows.CommunicationsApps	9WZDNCRFHVQM	A, B, C
48.	Microsoft.XboxGameOverlay	9NBLGGH537C2	A, B, C
49.	Microsoft.XboxIdentityProvider	9WZDNCRD1HKW	A, B, C
50.	Microsoft.XboxSpeechToTextOverlay	9P086NHDNB9W	A, B, C
51.	Microsoft.YourPhone	9NMPJ99VJBWV	A, B, C
52.	MicrosoftCorporationII.QuickAssist	9P7BP5VNWKX5	A, B, C
53.	MicrosoftWindows.Client.WebExperience	9MSSGKG348SP	A, B, C

5.3. Inbox Apps：已安装

5.3.1. 查看

```
Get-AppXProvisionedPackage -Path "D:\OS_11_Custom\Install\Install\Mount" | Out-GridView
```

5.3.2. 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
```

5.4. 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](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/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

    }
}
```

}

5.5. 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。

5.6. 区域标记：添加

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，想获取：

5.6.1. 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](#)

5.6.2. 手动下载

5.6.2.1. 区域

区域	产品 ID	区域	产品 ID
en-US	C9PDSCC711RVF	af-za	9PDW16B5HMXR
am-et	9NGL4R61W3PL	ar-sa	9N4S78P86PKX
as-in	9NTJLXMX35J	az-latn-az	9P5TFKZH5K8
be-by	9MXPBGNNNDW3L	bg-bg	9MX54588434F
bn-bd	9PH7TKVXGGM8	bn-in	9P1M44L7W84T
bs-latn-ba	9MVFKLJ10MFL	ca-es	9P6JMKJQZ9S7
ca-es-valencia	9P9K3WMFSW90	chr-cher-us	9MX15485N3RK
cs-cz	9P3WXZ1KTM7C	cy-gb	9NKJ9TBML4HB
da-dk	9NDMT2VKSNL1	de-de	9P6CT0SLW589

el-gr	9N586B13PBLD	en-gb	9NT52VQ39BVN
es-es	9NWVGWLHPB1Z	es-mx	9N8MCM1X3928
et-ee	9NFBHFMCR30L	eu-es	9NMCHQH37HZ
fa-ir	9NGS7DD4QS21	fi-fi	9MW3PQ7SD3QK
fil-ph	9NWM2KGTDSSS	fr-ca	9MTP2VP0VL92
fr-fr	9NHMG4BJKMDG	ga-ie	9P0L5Q848KXT
gd-gb	9P1DBPF36BF3	gl-es	9NXRNBARNJ9B
gu-in	9P2HMSWDJDQ1	ha-latn-ng	9N1L95DBGRG3
he-il	9NB6ZFND5HCQ	hi-in	9NZC3GRX8LD3
hr-hr	9NW01VND4LTW	hu-hu	9MWN3C58HL87
hy-am	9NKM28TM6P67	id-id	9P4X3N4SDK8P
ig-ng	9PG4ZFJ48JSX	is-is	9NTHJR7TQXX1
it-it	9P8PQWNS6VJX	ja-jp	9N1W692FV4S1
ka-ge	9P60JZL05WGH	kk-kz	9PHV179R97LV
km-kh	9PGKTS4JS531	kn-in	9NC6DB7N95F9
kok-in	9MV3P55CMZ6P	ko-kr	9N4TXPCVRNGF
ku-arab-iq	9P1C18QL3D7H	ky-kg	9P7D3JJGZM48
lb-lu	9N0ST1WBZ9D9	lo-la	9N8X352G5NZV
lt-lt	9NWWD891H6HN	lv-lv	9N5CQDPH6SQT
mi-nz	9P2GDFOB3JPSX	mk-mk	9P1X6XB1K3RN
ml-in	9NWDTV8FFV7L	mn-mn	9PG1DHC4VTZW
mr-in	9MWXCKHJVR1J	ms-my	9NPXL8ZSDDQ7
mt-mt	9PDG96SQ6BN8	nb-no	9N6J0M5DHCK0
ne-np	9P7CHPLWDQVN	nl-nl	9PF1C9NB5PRV
nn-no	9PK7KM3Z06KH	nso-za	9NS49QLX5CDV
or-in	9NTHCXCSJDH	pa-arab-pk	9NJRL03WH6FM
pa-in	9NSNC0ZJX69B	pl-pl	9NC5HW94R0LD
prs-af	9P3NGC6X5ZQC	pt-br	9P8LBDM4FW35
pt-pt	9P7X8QJ7FL0X	quc-latn-gt	9P2V6MNNQZ0B
quz-pe	9NHTX8NVQ04K	ro-ro	9MWXGPJ5PJ3H
ru-ru	9NMJCX77QKPX	rw-rw	9NFW0M20H9WG
sd-arab-pk	9NB9JSCXW9X5	si-lk	9NVF9QSLGTL0
sk-sk	9N7LSNN099WB	sl-si	9NV27L34J4ST
sq-al	9MWLRGNMDGK7	sr-cyrl-ba	9MXGN7V65C7B

sr-cyrl-rs	9PPD6CCK9K5H	sr-latn-rs	9NBZ0SJDPPT
sv-se	9P0HSNX08177	sw-ke	9NFF2M19DQ55
ta-in	9PDZB1WT1B34	te-in	9PMQJJGF63FW
tg-cyrl-tj	9MZHLBPPT2HC	th-th	9MSTWFRL0LR4
ti-et	9NC8C9RDNK2S	tk-tm	9NKHQ4GL6VLT
tn-za	9NFSXM123DHT	tr-tr	9NL1D3T5HG9R
tt-ru	9NV90Q1X1ZR2	ug-cn	9P52C5D7VL5S
uk-ua	9PPPMZRSGHR8	ur-pk	9NDWFTFW12BQ
uz-latn-uz	9P5P2T5P5L9S	vi-vn	9P0W68X0XZPT
wo-sn	9NH3SW1CR90F	xh-za	9NW3QWSLQD17
yo-ng	9NGM3VPPZS5V	zh-cn	9NRMNT6GMZ7
zh-tw	9PCJ4DHCQ1JQ	zu-za	9NNRM7KT5NB0

下载区域: zh-CN, 应用程序 ID: 9NRMNT6GMZ70, 商店连接:
<https://www.microsoft.com/store/productId/9NRMNT6GMZ70>

5.6.2.2. 打开网站: <https://store.rg-adguard.net>

- 5.6.2.2.1. 搜索关键词:

<https://www.microsoft.com/store/productId/9NRMNT6GMZ70>
- 5.6.2.2.2. 网页内搜索 22621 内容, 搜索结果: [Microsoft.LanguageExperiencePackzh-CN_22621.*.neutral__8wekyb3d8bbwe.appx](#)
- 5.6.2.2.3. 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](#)

5.6.3. 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
```

5.6.4. 区域标记: 新增变化

脱机映像已安装的语言将成为默认, 例如已安装语言: en-US、zh-CN, 那么重新安装或更新 InBox Apps 应用程序时, 将参考已安装语言来进行匹配, 自动添加对应的语言包, 可通过以下方式查看变化。

5.6.4.1. InBox Apps：已安装的应用程序包

5.6.4.1.1. 查看

```
Get-AppXProvisionedPackage -Path "D:\OS_11_Custom\Install\Install\Mount" | Out-GridView
```

5.6.4.1.2. 导出出 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
```

5.6.4.2. 查看可用的语言设置

```
Dism /Image:"D:\OS_11_Custom\Install\Install\Mount" /Get-Intl
```

5.7. InBox Apps：安装

5.7.1. 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；

5.7.2. 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"](#)
- 架构：[x64](#)
- [Install.Inst.InBox.Appx.ps1](#)
 - [\Expand\Install\Install.Inst.InBox.Appx.ps1](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/Install.Inst.InBox.Appx.ps1
- 复制代码

```
$ISO = "Auto"

$Mount = "D:\OS_11_Custom\Install\Install\Mount"

$Arch = "x64"

try{

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

```

$Current_Edition_Version = (Get-WindowsEdition -Path $Mount).Edition

Write-Host $Current_Edition_Version -ForegroundColor Green

} catch {

Write-Host "Error" -ForegroundColor Red

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

return

}

$Pre_Config_Rules = @{

Edition = @(

@{

Name = @( "CloudEdition"; )

Apps = @(

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

)

}

@{

Name = @( "CloudEditionN"; )

Apps = @(

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

)

}

@{

```



```

        Name = @(
            "Core"; "CoreN"; "CoreSingleLanguage";

        )

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

        )

    }

    @{
        Name = @(
            "Education"; "Professional"; "ProfessionalEducation"; "ProfessionalWorkstation"; "Enterprise"; "IoTEnterprise";
            "ServerRdsh";

        )

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

        )

    }

    @{

```

```

        Name = @(

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

        )

        Apps = @(

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

        )

    }

)

Rule = @(

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

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

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

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

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

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

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

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

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

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

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

```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

@{ Name="Microsoft.DesktopAppInstaller"; Match="DesktopAppInstaller";License="DesktopAppInstaller";
```

Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name="Microsoft.WindowsCamera"; Match="WindowsCamera";License="WindowsCamera";

Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.Getstarted"; Match="Getstarted";License="Getstarted";

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

@{ Name="Microsoft.Cortana"; Match="Cortana";License="Cortana";

Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00","Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name="Microsoft.GetHelp"; Match="GetHelp";License="GetHelp";

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

@{ Name="Microsoft.MicrosoftOfficeHub"; Match="MicrosoftOfficeHub";License="MicrosoftOfficeHub";

Dependencies=@("Microsoft.VCLibs.140.00","Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name="Microsoft.People"; Match="People";License="People";

Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.StorePurchaseApp"; Match="StorePurchaseApp";License="StorePurchaseApp";

Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.Todos"; Match="Todos";License="Todos";

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

@{ Name="Microsoft.WebMediaExtensions"; Match="WebMediaExtensions";License="WebMediaExtensions";

Dependencies=@("Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.Windows.Photos"; Match="Windows.Photos";License="Windows.Photos";

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

@{ Name="Microsoft.WindowsCalculator"; Match="WindowsCalculator";License="WindowsCalculator";

Dependencies=@("Microsoft.UI.Xaml.2.4","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.windowscommunicationsapps";

Match="WindowsCommunicationsApps";License="WindowsCommunicationsApps";

Dependencies=@("Microsoft.UI.Xaml.2.7","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.WindowsSoundRecorder"; Match="WindowsSoundRecorder";License="WindowsSoundRecorder";

Dependencies=@("Microsoft.UI.Xaml.2.3","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.Xbox.TCUI"; Match="Xbox.TCUI";License="Xbox.TCUI";

Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.XboxGameOverlay"; Match="XboxGameOverlay";License="XboxGameOverlay";

Dependencies=@("Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.XboxGamingOverlay"; Match="XboxGamingOverlay";License="XboxGamingOverlay";

Dependencies=@("Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.XboxIdentityProvider"; Match="XboxIdentityProvider";License="XboxIdentityProvider";

Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00"); }

@{ Name="Microsoft.XboxSpeechToTextOverlay";

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

@{ Name="Microsoft.YourPhone"; Match="YourPhone";License="YourPhone";

```
Dependencies=@("Microsoft.NET.Native.Framework.2.2","Microsoft.NET.Native.Runtime.2.2","Microsoft.VCLibs.140.00","Microsoft.VCLibs.140.00.UWPDesktop"); }

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

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

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

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

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

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

)

}

$Allow_Install_App = @()

ForEach ($item in $Pre_Config_Rules.Edition) {

    if ($item.Name -contains $Current_Edition_Version) {

        Write-host "`n  Match to: "-NoNewline; Write-host $Current_Edition_Version -ForegroundColor Green

        $Allow_Install_App = $item.Apps

        break

    }

}

Write-host "`n  The app to install ( $($Allow_Install_App.Count) item )" -ForegroundColor Yellow

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

ForEach ($item in $Allow_Install_App) {

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

}

Function Match_InBox_Apps_Install_Pack

{

    param ( $NewPath )

    $NewArch = $Arch

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

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

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

    if ($Pre_Config_Rules.Rule.Count -gt 0) {
```

```

ForEach ($itemInBoxApps in $Pre_Config_Rules.Rule){

    $InstallPacker = ""

    $InstallPackerCert = ""

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

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

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

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

            $InstallPacker = $_.FullName

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

                $InstallPackerCert = $_.FullName

            }

            $Script:InBoxAppx += @{

                Name      = $itemInBoxApps.Name;

                Depend    = $itemInBoxApps.Dependencies;

                Search    = $SearchNewStructure;

                InstallPacker = $InstallPacker;

                Certificate = $InstallPackerCert

                CertificateRule = $SearchNewLicense

            }

            return

        }

    }

}

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

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

$Script:InBoxAppx = @()

if ($ISO -eq "Auto") {

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

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

```

```

        Match_InBox_Apps_Install_Pack -NewPath $AppPath

    }

} else {

    Match_InBox_Apps_Install_Pack -NewPath $ISO

}

Write-host "  Search Complete" -ForegroundColor Green

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

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

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

    Write-host "  Match successful" -ForegroundColor Green

} else {

    Write-host "  Failed match" -ForegroundColor Red

    return

}

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

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

ForEach ($Rule in $Script:InBoxAppx) {

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

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

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

    Write-host ""

}

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

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

ForEach ($Rule in $Script:InBoxAppx) {

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

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

    if($Allow_Install_App -contains $Rule.Name) {

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

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

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

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

                Write-host "  License: " -NoNewline

                Write-host $Rule.Certificate -ForegroundColor Yellow

            }
        }
    }
}

```

```

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

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

        try{

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

            Write-Host "Done`n" -ForegroundColor Green

        } catch {

            Write-Host "Failed" -ForegroundColor Red

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

        }

    } else {

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

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

        try{

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

            Write-Host "Done`n" -ForegroundColor Green

        } catch {

            Write-Host "Failed" -ForegroundColor Red

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

        }

    }

} else {

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

}

} else {

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

}

}

```

5.8. 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](#)

- https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging/tutorial/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

        }

    }

}
```

5.9. 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
```

6. 累积更新

- 不同版本、旧版本升级到最新版时，需优先添加“功能启用包”后才能添加最新版累积更新；
- 添加语言包后，可安装与初始版本相同的累积更新，以解决安装后未刷新“组件：映像中已安装的所有包”状态的已知问题；
- 为保持最新，推荐您下载最新版。

6.1. 初始版本

6.1.1. 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。

6.1.1.1. 任选一个网站并打开：

6.1.1.1.1. <https://uupdump.net>

6.1.1.1.2. <https://uup.ee>

6.1.1.1.3. <https://osdump.com>

6.1.1.2. 打开后，搜索关键词：22621.382，在搜索结果选择：Windows 11, version 22H2 (22621.382) amd64 或 Windows 11, version 22H2 (22621.382) arm64

6.1.1.3. 打开后，选择“全部文件”；

6.1.1.4. 在“全部文件”页面里搜索：KB5016632，下载：

- [Windows11.0-KB5016632-x64.cab](#)
- [Windows11.0-KB5016632-x64.psf](#)

6.1.1.5. 累积更新：合并

6.1.1.5.1. 前往 <https://github.com/abbodi1406/WHd/tree/master/scripts> 后下载 PSFX_Repack_6.zip

6.1.1.5.2. 下载后解压到：D:\PSFX_Repack_6

6.1.1.5.3. 复制 Windows11.0-KB5016632-x64.cab、Windows11.0-KB5016632-x64.psf 到：
D:\PSFX_Repack_6

6.1.1.5.4. 运行：psfx2cab_GUI.cmd 后，将得到一个新的文件：

[Windows11.0-KB5016632-x64-full_psfx.cab](#)

6.1.1.5.5. 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](#)

6.1.2. 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
```

6.2. 其它版本

查阅“Windows 11 版本信息”，例如下载累积更新：KB5035853，版本号：22621.3296，前往下载页面：

<https://www.catalog.update.microsoft.com/Search.aspx?q=KB5035853>，下载后保存到：D:\OS_11_Custom\Install\Install\Update，
或通过直连下载，根据架构选择下载：

6.2.1. x64，默认

- 直连下载

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

- 添加

```
$KBPath = "D:\OS_11_Custom\Install\Install\Update\windows11.0-kb5035853-x64_8ca1a9a646dbe25c071a8057f249633a61929efa.msu"

Add-WindowsPackage -Path "D:\OS_11_Custom\Install\Install\Mount" -PackagePath $KBPath
```

6.2.2. Arm64

- 直连下载

```
https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/72ba502a-7982-471c-aad2-355efddbff61/public/windows11.0-kb5035853-arm64\_076c009bca80cfa85272bd50d65951c2f663fcc5.msu
```

- 添加

```
$KBPath = "D:\OS_11_Custom\Install\Install\Update\windows11.0-kb5035853-arm64_076c009bca80cfa85272bd50d65951c2f663fcc5.msu"

Add-WindowsPackage -Path "D:\OS_11_Custom\Install\Install\Mount" -PackagePath $KBPath
```

6.3. 固化更新

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
```

6.3.1. 固化更新后清理组件

```
$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

    }

}
```

7. 驱动

8. 部署引擎：添加

- 了解“[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。

9. 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
```

10. 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
```

11. 保存映像： Install.wim

```
Save-WindowsImage -Path "D:\OS_11_Custom\Install\Install\Mount"
```

12. 卸载映像： Install.wim

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。

13. 如何批量替换 Install.wim 里的所有索引号里的 WinRE.wim

13.1. 获取 WimLib

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

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

13.2.1. 从 Install.wim 里提取 WinRE.wim 文件 Install.wim

- Install.WinRE.Replace.wim.ps1
 - [\Expand\Install\Install.WinRE.Replace.wim.ps1](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/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
```

13.2.2. 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](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/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

}
```

14. 重建 Install.wim 后可缩小文件大小

- Install.Rebuild.wim.ps1
 - [\Expand\Install\Install.Rebuild.wim.ps1](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Install/Install.Rebuild.wim.ps1

- 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

}
}
```

III 自定义封装：boot.wim

1. 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 }
```

2. 指定挂载 Boot.wim 路径

```
New-Item -Path "D:\OS_11_Custom\Boot\Boot\Mount" -ItemType directory -ea SilentlyContinue
```

3. 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"
```

4. 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，查看报告“[语言安装包适用于 Boot.wim](#)”。
- 添加语言时，须对应不同的架构版本，未对应时，添加过程中报错等提示。

4.1. 语言包：添加

- Boot.Instl.lang.ps1
 - [\Expand\Boot\Boot.Instl.lang.ps1](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Boot/Boot.Instl.lang.ps1

- 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

}

}

}
```

4.2. 脱机映像语言：更改

4.2.1. 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
```

4.2.2. 查看可用的语言设置

```
Dism /Image:"D:\OS_11_Custom\Boot\Boot\Mount" /Get-Intl
```

4.3. 语言包：删除

- 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](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/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

}

}

}
```

4.4. 组件：映像中已安装的所有包

4.4.1. 查看

```
Get-WindowsPackage -Path "D:\OS_11_Custom\Boot\Boot\Mount" | Out-GridView
```

4.4.2. 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
```

4.5. 语言包：修复

4.5.1. 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：

- 4.5.1.1. [arunres.dll.mui](#)
- 4.5.1.2. [spwizres.dll.mui](#)
- 4.5.1.3. [w32uires.dll.mui](#)

4.5.2. 复制

将提取的文件复制到： [D:\OS_11_Custom\Boot\Boot\Mount\sources\zh-CN](#)

4.6. 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
```

4.7. 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。

4.7.1. 重新生成已挂载目录 lang.ini

重新生成的 Lang.ini 文件位置： [D:\OS_11_Custom\Boot\Boot\Mount\Sources\lang.ini](#)

```
Dism /image:"D:\OS_11_Custom\Boot\Boot\Mount" /gen-langini /distribution:"D:\OS_11_Custom\Boot\Boot\Mount"
```

4.7.2. 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:\OS_11\Sources\lang.ini](#)

```
Dism /image:"D:\OS_11_Custom\Boot\Boot\Mount" /gen-langini /distribution:"D:\OS_11"
```

5. 累积更新

准备可用的累积更新文件，请更改示例文件名：[KB_Boot.cab](#)

5.1. 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
```

5.2. 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
```

5.3. 固化更新

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
```

5.3.1. 固化更新后清理组件

```
$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

    }

}
```

6. 驱动

7. 其它

7.1. 绕过 TPM 安装时检查

- Boot.Bypass.TPM.ps1
 - [\Expand\Boot\Boot.Bypass.TPM.ps1](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/Boot/Boot.Bypass.TPM.ps1

- 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
```

8. 保存映像： Boot.wim

```
Save-WindowsImage -Path "D:\OS_11_Custom\Boot\Boot\Mount"
```

9. 卸载映像： Boot.wim

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
```

IV 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：1.1.1.1，选择下载源代码格式：zip，下载完成后得到 Multilingual-1.1.1.1.zip 压缩包文件；

- 将已下载的 main.zip 或 Multilingual-1.1.1.1.zip，解压到：D:\Multilingual-1.1.1.1，重命名：D:\Multilingual
- 学习“无人值守 Windows 安装参考”，通过无人值守来干预安装过程。

1. 添加方式

1.1. 添加到 ISO 安装介质

1.1.1. 无人值守

1.1.1.1. 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_Learn\Unattend\Mul.Unattend.xml 到 D:\OS_11\Autounattend.xml

```
Copy-Item "D:\Multilingual\_Learn\Unattend\Mul.Unattend.xml" -Destination "D:\OS_11\Autounattend.xml" -Force
```

1.1.1.2. 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_Learn\Unattend\Mul.Unattend.xml 到 D:\OS_11\Sources\Unattend.xml

```
Copy-Item "D:\Multilingual\_Learn\Unattend\Mul.Unattend.xml" -Destination "D:\OS_11\Sources\Unattend.xml" -Force
```

1.1.1.3. 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

1.1.1.3.1. 创建 \$OEM\$ 路径

```
New-Item -Path "D:\OS_11\sources\`$OEM$\`$$\Panther" -ItemType Directory
```

1.1.1.3.2. 复制

复制 D:\Multilingual_Learn\Unattend\Mul.Unattend.xml 到

D:\OS_11\Sources\\$\OEM\$\\$\$\Panther\Unattend.xml

```
Copy-Item "D:\Multilingual\_Learn\Unattend\Mul.Unattend.xml" -Destination "D:\OS_11\sources\`$OEM$\`$$\Panther\Unattend.xml" -Force
```

1.1.2. 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。

1.1.2.1. 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
```

1.1.2.2. 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
```

```
}
```

1.2. 添加到已挂载

通过“自定义封装：Install.wim”，执行“开始挂载 Install.wim”，挂载到：D:\OS_11_Custom\Install\Install\Mount

1.2.1. 无人值守

复制 D:\Multilingual\Learn\Unattend\Mul.Unattend.xml 到 D:\OS_11_Custom\Install\Install\Mount\Panther\Unattend.xml

```
Copy-Item "D:\Multilingual\Learn\Unattend\Mul.Unattend.xml" -Destination "D:\OS_11_Custom\Install\Install\Mount\Panther" -Force
```

1.2.2. 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。

1.2.2.1. 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
```

1.2.2.2. 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

}
```

2. 部署引擎：进阶

2.1. 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。

2.2. 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](#)

2.2.1. 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。

2.2.1.1. 多语言

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

<InputLocale>%OSDInputLocale%</InputLocale>

<SystemLocale>%OSDSysLocale%</SystemLocale>

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

```
<UILanguageFallback>%OSDUILanguageFallback%</UILanguageFallback>

<UserLocale>%OSDUserLocale%</UserLocale>
```

2.2.1.2. 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>
```

2.2.2. 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。

2.2.2.1. 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>
```

2.2.2.2. 自定义用户

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

2.2.2.2.1. 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>`

2.2.2.2.2. 替换

从开始处 `<OOBE>` 到 `</OOBE>`

```
<OOBE>

<ProtectYourPC>3</ProtectYourPC>

<HideEULAPage>true</HideEULAPage>

<HideWirelessSetupInOOBE>true</HideWirelessSetupInOOBE>

</OOBE>
```

D. ISO

II 生成 ISO

1. 下载 OScdimg

根据架构选择 OScdimg 版本，下载后保存到：D:\，保存在其它路径请输入 OScdimg.exe 绝对路径；

1.1. x64

https://github.com/ilikeyi/solutions/raw/main/_Software/OScdimg/amd64/oscdimg.exe

1.2. x86

https://github.com/ilikeyi/solutions/raw/main/_Software/OScdimg/x86/oscdimg.exe

1.3. arm64

2. 使用 oscding 命令行生成一个 ISO 文件，保存到：D:\Win11.iso

- ISO.ps1
 - [\Expand\ISO.ps1](#)
 - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/22H2/Expand/ISO.ps1

• 复制代码

```
$Oscding = "D:\Oscding.exe"

$ISO = "D:\OS_11"

$Volume = "OS_11"

$SaveTo = "D:\OS_11.iso"

$Arguments = @("-m", "-o", "-u2", "-udfver102", "-l" "$($Volume)", "-bootdata:2#p0,e,b" "$($ISO)\boot\etfsboot.com" "#pEF,e,b" "$($ISO)\efi\microsoft\boot\efisys.bin", $ISO, $SaveTo)

Start-Process -FilePath $Oscding -ArgumentList $Arguments -wait -nonewwindow
```

III 绕过 TPM 安装检查

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。

章节 2 报告

- 编组缩写：
 - 群组 A：Windows 11 教育版、Windows 11 企业版、Windows 11 专业版、Windows 11 专业教育版、Windows 11 专业版工作站、Windows 11 IoT 企业版
 - 群组 B：Windows 11 教育版 N、Windows 11 企业版 N、Windows 11 专业版 N、Windows 11 专业教育版 N、Windows 11 专业版工作站 N
 - 群组 C：Windows 11 家庭版、Windows 11 家庭单语版
 - 群组 D：Windows 11 家庭版 N

II 不同版本

1. Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~~10.0.22621.382 A, C
- Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.317
2. Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~~10.0.22621.1 A, C
- Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1

3.	Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B
4.	Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1	
5.	Microsoft-OneCore-ApplicationModel-Sync-Desktop-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.317	A, B, C, D
6.	Microsoft-OneCore-DirectX-Database-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
7.	Microsoft-Windows-Client-LanguagePack-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.382	A, B, C, D
8.	Microsoft-Windows-Ethernet-Client-Intel-E1i68x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
9.	Microsoft-Windows-Ethernet-Client-Intel-E2f68-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
10.	Microsoft-Windows-Ethernet-Client-Realtek-Rtcx21x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
11.	Microsoft-Windows-Ethernet-Client-Vmware-Vmxnet3-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
12.	Microsoft-Windows-FodMetadata-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
13.	Microsoft-Windows-Foundation-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
14.	Microsoft-Windows-Hello-Face-Package~31bf3856ad364e35~amd64~~10.0.22621.317	A, B, C, D
15.	Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~~11.0.22621.1	A, B, C, D
	Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~en-US~11.0.22621.1	A, B, C, D
16.	Microsoft-Windows-Kernel-LA57-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.382	A, B, C, D
17.	Microsoft-Windows-LanguageFeatures-Basic-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
18.	Microsoft-Windows-LanguageFeatures-Handwriting-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
19.	Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
20.	Microsoft-Windows-LanguageFeatures-Speech-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
21.	Microsoft-Windows-LanguageFeatures-TextToSpeech-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
22.	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1	A, B, C, D
23.	Microsoft-Windows -Notepad-System-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1	A, B, C, D
	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1	A, B, C, D
24.	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1	A, B, C, D
25.	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~~10.0.22621.1	A, B, C, D
	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1	A, B, C, D
26.	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1	A, B, C, D
27.	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~~10.0.22621.1	A, B, C, D
	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1	A, B, C, D
28.	Microsoft-Windows-TabletPCMath-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
29.	Microsoft-Windows-Wallpaper-Content-Extended-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D

30.	Microsoft-Windows-Wifi-Client-Broadcom-Bcmpecidhd63-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
31.	Microsoft-Windows-Wifi-Client-Broadcom-Bcmwl63a-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
32.	Microsoft-Windows-Wifi-Client-Broadcom-Bcmwl63al-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
33.	Microsoft-Windows-Wifi-Client-Intel-Netwbw02-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
34.	Microsoft-Windows-Wifi-Client-Intel-Netwew00-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
35.	Microsoft-Windows-Wifi-Client-Intel-Netwew01-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
36.	Microsoft-Windows-Wifi-Client-Intel-Netwlv64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
37.	Microsoft-Windows-Wifi-Client-Intel-Netwns64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
38.	Microsoft-Windows-Wifi-Client-Intel-Netwsw00-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
39.	Microsoft-Windows-Wifi-Client-Intel-Netwtw02-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
40.	Microsoft-Windows-Wifi-Client-Intel-Netwtw04-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
41.	Microsoft-Windows-Wifi-Client-Intel-Netwtw06-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
42.	Microsoft-Windows-Wifi-Client-Intel-Netwtw08-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
43.	Microsoft-Windows-Wifi-Client-Intel-Netwtw10-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
44.	Microsoft-Windows-Wifi-Client-Marvel-Mrvlpcie8897-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
45.	Microsoft-Windows-Wifi-Client-Qualcomm-Athw8x-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
46.	Microsoft-Windows-Wifi-Client-Qualcomm-Athwnx-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
47.	Microsoft-Windows-Wifi-Client-Qualcomm-Qcamain10x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
48.	Microsoft-Windows-Wifi-Client-Ralink-Netr28x-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
49.	Microsoft-Windows-Wifi-Client-Realtek-Rtl8187se-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
50.	Microsoft-Windows-Wifi-Client-Realtek-Rtl8192se-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
51.	Microsoft-Windows-Wifi-Client-Realtek-Rtl819xp-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
52.	Microsoft-Windows-Wifi-Client-Realtek-Rtl85n64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
53.	Microsoft-Windows-Wifi-Client-Realtek-Rtlwlan-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
54.	Microsoft-Windows-Wifi-Client-Realtek-Rtlwlan01-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
55.	Microsoft-Windows-Wifi-Client-Realtek-Rtlwlan13-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
56.	Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.382	A, B, C, D
	Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621	A, B, C, D
57.	Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1	A, B, C, D
	Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1	A, B, C, D
58.	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.317	A, B, C, D
	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1	A, B, C, D
59.	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1	A, B, C, D
	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1	A, B, C, D

60.	OpenSSH-Client-Package~31bf3856ad364e35~amd64~~10.0.22621.1	A, B, C, D
61.	Package_for_DotNetRollup_481~31bf3856ad364e35~amd64~~10.0.9065.6	A, B, C, D
62.	Package_for_RollupFix~31bf3856ad364e35~amd64~~22621.382.1.13	A, B, C, D
63.	Package_for_ServicingStack_378~31bf3856ad364e35~amd64~~22621.378.1.0	A, B, C, D

III 语言安装包适用于：Install.wim

1.	Microsoft-Windows-MediaPlayer-Package-AMD64-zh-CN.cab	A, C
	Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1	
2.	Microsoft-Windows-MediaPlayer-Package-wow64-zh-CN.cab	A,C
	Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1	
3.	Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab	A, B
	Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1	
4.	Microsoft-Windows-LanguageFeatures-Fonts-Hans-Package~31bf3856ad364e35~AMD64~~.cab	A, B, C, D
	Microsoft-Windows-LanguageFeatures-Fonts-Hans-Package~31bf3856ad364e35~amd64~~10.0.22621.1	
5.	Microsoft-Windows-Client-Language-Pack_x64_zh-CN.cab	A, B, C, D
	Microsoft-Windows-Client-LanguagePack-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1	
6.	Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~AMD64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~zh-CN~11.0.22621.1	
7.	Microsoft-Windows-LanguageFeatures-Basic-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab	A, B, C, D
	Microsoft-Windows-LanguageFeatures-Basic-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1	
8.	Microsoft-Windows-LanguageFeatures-Handwriting-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab	A, B, C, D
	Microsoft-Windows-LanguageFeatures-Handwriting-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1	
9.	Microsoft-Windows-LanguageFeatures-OCR-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab	A, B, C, D
	Microsoft-Windows-LanguageFeatures-OCR-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1	
10.	Microsoft-Windows-LanguageFeatures-Speech-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab	A, B, C, D
	Microsoft-Windows-LanguageFeatures-Speech-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1	
11.	Microsoft-Windows-LanguageFeatures-TextToSpeech-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab	A, B, C, D
	Microsoft-Windows-LanguageFeatures-TextToSpeech-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1	
12.	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1	
13.	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1	
14.	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1	

15.

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

Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1
16.

Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~AMD64~zh-CN~.cab

A, B, C, D

Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
17.

Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~zh-CN~.cab

A, B, C, D

Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1
18.

Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab

A, B, C, D

Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
19.

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

A, B, C, D

Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1
20.

Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab

A, B, C, D

Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
21.

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

A, B, C, D

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

IV

语言安装包适用于：WinRE.wim

1.

WinPE-FontSupport-zh-CN.cab

Microsoft-Windows-WinPE-FontSupport-ZH-CN-Package~31bf3856ad364e35~amd64~~10.0.22621.1
2.

lp.cab

Microsoft-Windows-WinPE-LanguagePack-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
3.

winpe-appxdeployment_zh-CN.cab

WinPE-AppxDeployment-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
4.

winpe-appxpackaging_zh-CN.cab

WinPE-AppxPackaging-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
5.

winpe-atbroker_zh-CN.cab

WinPE-ATBroker-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
6.

winpe-audiocore_zh-CN.cab

WinPE-AudioCore-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
7.

winpe-audiodrivers_zh-CN.cab

WinPE-AudioDrivers-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
8.

winpe-enhancedstorage_zh-CN.cab

WinPE-EnhancedStorage-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
9.

winpe-hta_zh-CN.cab

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

10.

winpe-narrator_zh-CN.cab

WinPE-Narrator-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
11.

winpe-opcservices_zh-CN.cab

WinPE-OpcServices-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
12.

winpe-rejuv_zh-CN.cab

WinPE-Rejuv-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
13.

winpe-scripting_zh-CN.cab

WinPE-Scripting-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
14.

winpe-securestartup_zh-CN.cab

WinPE-SecureStartup-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
15.

winpe-speech-tts_zh-CN.cab

WinPE-Speech-TTS-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
16.

winpe-srh_zh-CN.cab

WinPE-SRH-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
17.

winpe-srt_zh-CN.cab

WinPE-SRT-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
18.

winpe-storagewmi_zh-CN.cab

WinPE-StorageWMI-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
19.

winpe-wds-tools_zh-CN.cab

WinPE-WDS-Tools-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
20.

winpe-wifi_zh-CN.cab

WinPE-WiFi-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
21.

winpe-windowsupdate_zh-CN.cab

WinPE-WindowsUpdate-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
22.

winpe-wmi_zh-CN.cab

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

V

语言安装包适用于：Boot.wim

1.

WinPE-FontSupport-zh-CN.cab

Microsoft-Windows-WinPE-FontSupport-ZH-CN-Package~31bf3856ad364e35~amd64~~10.0.22621.1
2.

lp.cab

Microsoft-Windows-WinPE-LanguagePack-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
3.

winpe-atbroker_zh-CN.cab

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

4.

winpe-audiocore_zh-CN.cab

WinPE-AudioCore-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
5.

winpe-audiodrivers_zh-CN.cab

WinPE-AudioDrivers-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
6.

winpe-enhancedstorage_zh-CN.cab

WinPE-EnhancedStorage-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
7.

winpe-narrator_zh-CN.cab

WinPE-Narrator-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
8.

winpe-scripting_zh-CN.cab

WinPE-Scripting-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
9.

winpe-securestartup_zh-CN.cab

WinPE-SecureStartup-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
10.

WINPE-SETUP-CLIENT_zh-CN.CAB

WinPE-Setup-Client-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
11.

WinPE-Setup_zh-CN.cab

WinPE-Setup-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
12.

winpe-speech-tts_zh-CN.cab

WinPE-Speech-TTS-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
13.

winpe-srh_zh-CN.cab

WinPE-SRH-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
14.

winpe-srt_zh-CN.cab

WinPE-SRT-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
15.

winpe-wds-tools_zh-CN.cab

WinPE-WDS-Tools-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
16.

winpe-wmi_zh-CN.cab

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

章节 3

常见问题

II

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

III 修复挂载出现异常的问题

1. 查看已挂载

```
Get-WindowsImage -Mounted
```

2. 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
```

3. 删除与已损坏的已装载映像关联的所有资源。

```
Clear-WindowsCorruptMountPoint
```

```
Dism /cleanup-wim
```

IV 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
```

章节 4 已知问题

暂无



此副本封装教程隶属于 Yi's Solutions 内容，学习更多：

- Yi 的官方网站 | <https://fengyi.tel/solutions>
- Github | <https://github.com/ilikeyi/solutions>

作者：Yi

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

文档版本：1.2

文档模型：完整版

更新日期：2024 - 8

建议或反馈： <https://github.com/ilikeyi/solutions/issues>