



## MICROSOFT WINDOWS 11 24H2 AND 25H2

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，你准备好开始这场封装之旅了吗？

### 摘要

章节 1    封装

章节 2    常见问题

章节 3    已知问题

章节 1	封装	第 6 页
A.	先决条件	第 6 页
I.	正在运行的系统	第 6 页
1.	使用 DISM 命令制作高版本镜像时	第 6 页
2.	磁盘分区	第 6 页
3.	加速 Windows 操作系统的 N 种方法	第 6 页
3.1.	关闭 Windows 安全中心	第 6 页
3.2.	关闭基于虚拟化的安全性	第 7 页
4.	ISO 工具	第 7 页
5.	压缩软件	第 7 页
6.	命令行	第 8 页
II.	要求	第 8 页
1.	系统安装包	第 8 页
2.	语言包	第 11 页
2.1.	学习	第 11 页
2.2.	语言包：下载	第 13 页
3.	InBox Apps	第 13 页
B.	语言包：提取	第 15 页
I.	语言包：准备	第 15 页
II.	语言包：提取方案	第 15 页
III.	执行提取命令	第 15 页
C.	自定义封装	第 22 页
I.	自定义封装：Install.wim	第 22 页
1.	查看 Install.wim 详细信息	第 22 页

2.	指定挂载 Install 路径 .....	第 22 页
3.	开始挂载 Install.wim .....	第 22 页
3.1.	自定义封装：WinRE.wim .....	第 22 页
3.1.1.	查看 WinRE.wim 详细信息 .....	第 22 页
3.1.2.	指定挂载 WinRE.wim 路径 .....	第 23 页
3.1.3.	开始挂载 WinRE.wim .....	第 23 页
3.1.4.	语言包 .....	第 23 页
3.1.4.1.	语言包：添加 .....	第 23 页
3.1.4.2.	脱机映像语言：更改 .....	第 25 页
3.1.4.2.1.	更改默认语言、区域设置和其他国际设置 .....	第 25 页
3.1.4.2.2.	查看可用的语言设置 .....	第 25 页
3.1.4.3.	组件：映像中已安装的所有包 .....	第 25 页
3.1.5.	累积更新 .....	第 25 页
3.1.5.1.	添加 .....	第 25 页
3.1.5.2.	固化更新 .....	第 26 页
3.1.5.2.1.	固化更新后清理组件 .....	第 26 页
3.1.6.	驱动 .....	第 26 页
3.1.7.	保存映像：WinRE.wim .....	第 26 页
3.1.8.	卸载映像：WinRE.wim .....	第 26 页
3.1.9.	重建 WinRE.wim 后，可缩小文件大小 .....	第 27 页
3.1.10.	备份 WinRE.wim .....	第 27 页
3.1.11.	替换 Install.wim 映像内的 WinRE.wim .....	第 28 页
4.	语言包 .....	第 28 页
4.1.	语言包：添加 .....	第 28 页
4.2.	脱机映像语言：更改 .....	第 34 页
4.2.1.	更改默认语言、区域设置和其他国际设置 .....	第 34 页

4.2.2.	查看可用的语言设置	第 34 页
4.3.	组件：映像中已安装的所有包	第 34 页
5.	InBox Apps	第 35 页
5.1.	学习	第 35 页
5.2.	InBox Apps：预安装应用程序	第 35 页
5.3.	InBox Apps：已安装	第 37 页
5.4.	删除已安装的所有预应用程序	第 37 页
5.5.	已安装语言标记：了解	第 38 页
5.6.	脱机映像已安装语言标记：添加方式	第 38 页
5.7.	InBox Apps：安装	第 41 页
5.8.	脱机映像已安装语言标记：删除	第 51 页
5.9.	InBox Apps：优化	第 52 页
6.	累积更新	第 52 页
6.1.	功能启用包	第 52 页
6.2.	初始版本累积更新	第 53 页
6.3.	其它版本	第 54 页
6.4.	固化更新	第 55 页
6.4.1.	固化更新后清理组件	第 55 页
7.	驱动	第 55 页
8.	部署引擎：添加	第 55 页
9.	健康	第 55 页
10.	替换 WinRE.wim	第 55 页
11.	保存映像：Install.wim	第 55 页
12.	卸载映像：Install.wim	第 55 页
13.	如何批量替换 Install.wim 里的所有索引号里的 WinRE.wim	第 56 页
13.1.	获取 WimLib	第 56 页

13.2.	如何在 Install.wim 里提取和更新 WinRE.wim	第 56 页
14.	重建 Install.wim 后可缩小文件大小	第 57 页
15.	拆分、合并、压缩、互转	第 58 页
15.1.	拆分和合并	第 58 页
15.1.1.	拆分	第 58 页
15.1.2.	合并	第 58 页
15.2.	固实压缩 ESD 格式和互转 WIM 格式	第 59 页
15.2.1.	固实压缩	第 59 页
15.2.2.	压缩文件转换为 WIM 文件格式	第 60 页
II.	自定义封装: boot.wim	第 61 页
1.	查看 Boot.wim 文件信息	第 61 页
2.	指定挂载 Boot.wim 路径	第 61 页
3.	开始挂载 Boot.wim	第 61 页
4.	语言包	第 61 页
4.1.	语言包: 添加	第 61 页
4.2.	脱机映像语言: 更改	第 63 页
4.2.1.	更改默认语言、区域设置和其他国际设置	第 63 页
4.2.2.	查看可用的语言设置	第 63 页
4.3.	组件: 映像中已安装的所有包	第 63 页
4.4.	语言包: 同步到 ISO 安装程序	第 63 页
4.5.	重新生成 Lang.ini	第 63 页
4.5.1.	重新生成已挂载目录 lang.ini	第 64 页
4.5.2.	重新生成 lang.ini 后, 同步到安装程序	第 64 页
5.	累积更新	第 64 页
5.1.	添加	第 64 页
5.2.	删除	第 64 页

5.3.	固化更新	第 64 页
5.3.1.	固化更新后清理组件	第 64 页
6.	驱动	第 64 页
7.	其它	第 64 页
7.1.	绕过 TPM 安装时检查	第 65 页
8.	保存映像：Boot.wim	第 65 页
9.	卸载映像：Boot.wim	第 65 页
III.	部署引擎	第 65 页
1.	添加方式	第 66 页
2.	部署引擎：进阶	第 69 页
D.	ISO	第 71 页
I.	生成 ISO	第 71 页
II.	绕过 TPM 安装检查	第 72 页
章节 2	常见问题	第 72 页
I.	清理所有挂载到	第 72 页
II.	修复挂载出现异常的问题	第 72 页
III.	清理	第 73 页
章节 3	已知问题	第 74 页

## 章节 1 封装

### A. 先决条件

#### I. 正在运行的系统

##### 1. 使用 DISM 命令制作高版本镜像时

正在运行的操作系统是 Windows 10 或者低于 Windows 11 25H2 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：

1.1. 升级正在运行的操作系统或重新安装到更高的版本（建议）；

1.2. 升级或安装新版 ADK 或 PowerShell 版本（不建议）

1.2.1. 可尝试升级到最新 PowerShell 7 以上的版本；

1.2.2. 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。

##### 2. 磁盘分区

2.1. 挂载脱机映像到 REFS 磁盘分区后，执行部分 DISM 命令会出现异常，使用 NTFS 格式的磁盘分区不受此影响。

2.2. ISO 解压后，所在的位置不受 REFS 分区影响。

##### 3. 加速 Windows 操作系统的 N 种方法

在处理封装任务时，安装累积更新、安装驱动、安装 InBox Apps 里的应用时，会产生大量的临时文件，可用以下几种方法加速系统：

###### 3.1. 关闭 Windows 安全中心

- 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”。

3.1.1. 打开“Windows 安全中心”或运行：`windowsdefender:`

3.1.2. 选择“病毒和威胁防护”或运行：`windowsdefender://threat`

3.1.3. 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：

- 3.1.3.1. 实时保护
- 3.1.3.2. 云提供的保护
- 3.1.3.3. 自动提交样本
- 3.1.3.4. 篡改防护

3.1.4. 未处于封装时，建议您开启 Windows 安全中心。

3.2. 关闭基于虚拟化的安全性

即便是关闭 Windows 安全中心后，基于虚拟化的安全性处于运行中，系统运行速度也会大打折扣，关闭后提度很明显。

3.2.1. 运行后，重新启动计算机

```
dism /Online /Disable-Feature:microsoft-hyper-v-all /NoRestart

dism /Online /Disable-Feature:IsolatedUserMode /NoRestart

dism /Online /Disable-Feature:Microsoft-Hyper-V-Hypervisor /NoRestart

dism /Online /Disable-Feature:Microsoft-Hyper-V-Online /NoRestart

dism /Online /Disable-Feature:HypervisorPlatform /NoRestart

mountvol X: /s

cmd /c copy /y %WINDIR%\System32\SecConfig.efi X:\EFI\Microsoft\Boot\SecConfig.efi

bcdedit /create {0cb3b571-2f2e-4343-a879-d86a476d7215} /d "DebugTool" /application osloader

bcdedit /set {0cb3b571-2f2e-4343-a879-d86a476d7215} path "\EFI\Microsoft\Boot\SecConfig.efi"

bcdedit /set {bootmgr} bootsequence {0cb3b571-2f2e-4343-a879-d86a476d7215}

bcdedit /set {0cb3b571-2f2e-4343-a879-d86a476d7215} loadoptions DISABLE-LSA-ISO,DISABLE-VBS

bcdedit /set {0cb3b571-2f2e-4343-a879-d86a476d7215} device partition=X:

mountvol X: /d

bcdedit /set hypervisorlaunchtype off
```

3.2.2. 查看状态

运行 `Msinfo32`，在系统摘要查看 “基于虚拟化的安全” 状态

4. ISO 工具

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

5. 压缩软件



准备一款常用的压缩软件：7-Zip、WinRAR 或其它，推荐您下载 7-Zip 并安装，根据架构选择下载：

5.1. 7-Zip

x64: <https://www.7-zip.org/a/7z2501-x64.exe>

x86: <https://www.7-zip.org/a/7z2501.exe>

arm64: <https://www.7-zip.org/a/7z2501-arm64.exe>

6. 命令行

6.1. 可选“Terminal”或“PowerShell ISE”，未安装“Terminal”，请前往 <https://github.com/microsoft/terminal/releases> 后下载；

6.2. 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 -Scope LocalMachine -Force
```

6.3. 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；

6.4. 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"

II. 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. 制作 Windows 11 24H2 和 25H2 时，请使用 Windows 11 24H2（26100.1742）初始版本；

1.1.2. 每版本更新时请重新制作镜像，例如从 24H2 跨越到 25H2 时，应避免出现其它兼容性问题请勿在旧镜像基础上制作；

1.1.3. 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；

1.1.4. 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；

1.1.5. 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。

1.1.6. 进化过程：Windows 11 24H2, Build 26100.1742 + （“KB5054156 检查点更新” + “KB5044284 每月累积更新”）= OS Build 26200.6725

1.2. 准备下载初始版本或开发者版本

1.2.1. x64

- 1.2.1.1. 文件名: [en-us\\_windows\\_11\\_consumer\\_editions\\_version\\_24h2\\_x64\\_dvd\\_1d5fcad3.iso](#)  
  
文件列表: <https://files.rg-adguard.net/file/1ebf9c88-803f-636e-ad8a-5b60966dcd64>
- 1.2.1.2. 文件名: [en-us\\_windows\\_11\\_business\\_editions\\_version\\_24h2\\_x64\\_dvd\\_59a1851e.iso](#)  
  
文件列表: <https://files.rg-adguard.net/file/4bfa831a-0073-3bb4-5dde-6c07df68d7e3>
- 1.2.1.3. 文件名: [en-us\\_windows\\_11\\_iot\\_enterprise\\_version\\_24h2\\_x64\\_dvd\\_3a99b72b.iso](#)  
  
文件列表: <https://files.rg-adguard.net/file/d8ee9445-b9eb-5f45-f75e-e92a057820bf>
- 1.2.1.4. 文件名: [en-us\\_windows\\_11\\_enterprise\\_ltsc\\_2024\\_x64\\_dvd\\_965cfb00.iso](#)  
  
文件列表: <https://files.rg-adguard.net/file/142ca376-487f-e858-a606-e120e70b9d02>
- 1.2.1.5. 文件名: [en-us\\_windows\\_11\\_iot\\_enterprise\\_ltsc\\_2024\\_x64\\_dvd\\_f6b14814.iso](#)
- 1.2.1.6. 文件列表: <https://files.rg-adguard.net/file/9160d5cf-f480-a1c9-a62c-b75ab0708d2c>
- 1.2.2. arm64
  - 1.2.2.1. 文件名: [en-us\\_windows\\_11\\_consumer\\_editions\\_version\\_24h2\\_arm64\\_dvd\\_4cc70bf6.iso](#)  
  
文件列表: <https://files.rg-adguard.net/file/9f751f60-0919-4a5a-8fb5-7bd6340a5df6>
  - 1.2.2.2. 文件名: [en-us\\_windows\\_11\\_business\\_editions\\_version\\_24h2\\_arm64\\_dvd\\_ad92e9d8.iso](#)  
  
文件列表: <https://files.rg-adguard.net/file/cfb08972-7f11-f3a5-ae7c-8f084dc1996e>
  - 1.2.2.3. 文件名: [en-us\\_windows\\_11\\_iot\\_enterprise\\_version\\_24h2\\_arm64\\_dvd\\_e9155a10.iso](#)  
  
文件列表: <https://files.rg-adguard.net/file/bacf17e5-a307-5a01-1a35-71268dc0c2e3>
  - 1.2.2.4. 文件名: [en-us\\_windows\\_11\\_iot\\_enterprise\\_ltsc\\_2024\\_arm64\\_dvd\\_ec517836.iso](#)  
  
文件列表: <https://files.rg-adguard.net/file/de2234af-1514-ff50-50af-5a3395549c42>
- 1.3. 示例下载 [en-us\\_windows\\_11\\_consumer\\_editions\\_version\\_24h2\\_x64\\_dvd\\_1d5fcad3.iso](#) 后, 解压到: [D:\en-us\\_windows\\_11\\_consumer\\_editions\\_version\\_24h2\\_x64\\_dvd\\_1d5fcad3](#)

**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 DISM 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 D:\en-us\_windows\_11\_consumer\_editions\_version\_24h2\_x64\_dvd\_1d5fcad3 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](#)  
  
○ [\Expand\Install\Report.Overview.xlsx](#)  
  
○ [https://raw.githubusercontent.com/ilikeyi/solutions/main/\\_Learn/Packaging/tutorial/OS.11/25H2/Expand/Install/Report.Overview.xlsx](https://raw.githubusercontent.com/ilikeyi/solutions/main/_Learn/Packaging/tutorial/OS.11/25H2/Expand/Install/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:

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

2.2. 语言包: 下载

2.2.1. 下载: [https://software-static.download.prss.microsoft.com/dbazure/888969d5-f34g-4e03-ac9d-1f9786c66749/26100.1.240331-1435.ge\\_release\\_amd64fre\\_CLIENT\\_LOF\\_PACKAGES\\_OEM.iso](https://software-static.download.prss.microsoft.com/dbazure/888969d5-f34g-4e03-ac9d-1f9786c66749/26100.1.240331-1435.ge_release_amd64fre_CLIENT_LOF_PACKAGES_OEM.iso)

文件列表: <https://files.rg-adguard.net/file/025cfc5d-f5fa-7d00-246e-76c04a40e210>

3. InBox Apps

3.1. 下载: [https://software-static.download.prss.microsoft.com/dbazure/888969d5-f34g-4e03-ac9d-1f9786c66749/26100.6584.250904-1728.ge\\_release\\_svc\\_prod1\\_amd64fre\\_InboxApps.iso](https://software-static.download.prss.microsoft.com/dbazure/888969d5-f34g-4e03-ac9d-1f9786c66749/26100.6584.250904-1728.ge_release_svc_prod1_amd64fre_InboxApps.iso)

文件列表: <https://files.rg-adguard.net/file/ada764e8-717a-d080-099b-d3aa44b84dab>

3.2. 修复缺少项

3.2.1. Microsoft.HEVCVideoExtension

26100.6584.250904-1728.ge\_release\_svc\_prod1\_amd64fre\_InboxApps.iso 里未包含

Microsoft.HEVCVideoExtension\_8wekyb3d8bbwe.appxbundle 安装包文件, 必须使用该版本: 2.4.13.0, 下载时请可通

过以下方式获取：

3.2.1.1. 无法获得文件时，请通过手动方式下载：

3.2.1.1.1. Microsoft.HEVCVideoExtension\_8wekyb3d8bbwe.appxbundle

3.2.1.1.1.1. 直连下载

- [https://github.com/ilikeyi/Solutions/raw/refs/heads/main/\\_Learn/Download.Package/Windows.11.25H2/Microsoft.HEVCVideoExtension\\_8wekyb3d8bbwe.appxbundle](https://github.com/ilikeyi/Solutions/raw/refs/heads/main/_Learn/Download.Package/Windows.11.25H2/Microsoft.HEVCVideoExtension_8wekyb3d8bbwe.appxbundle)

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

3.2.1.1.1.2.1. a). 选择 URL (link) ， b). 选择频道 (Retail) ， c). 输入关键词：

<https://www.microsoft.com/store/productId/9N4WGH0Z6VHQ>

3.2.1.1.1.2.2. 搜索结果：

[Microsoft.HEVCVideoExtension\\_2.4.13.0.\\*.appxbundle](#)

3.2.1.1.1.2.3. 下载后保存到“桌面”，重命名：

[Microsoft.HEVCVideoExtension\\_8wekyb3d8bbwe.appxbundle](#)

3.2.1.1.2. 许可证

3.2.1.1.2.1. 预保存的许可证

将{所在位

[置}:\YiSolutions\\\_Learn\Packaging.tutorial\OS.11\25H2\Expand\License\15a63](#)

[83d188a454ca98e15931ce48dc4\\_License1.xml](#) 复制到桌面，本地没有就在

线下载获取。

3.2.1.1.2.2. 通过在线下载连接，下载文件后，保存到桌面

[https://github.com/ilikeyi/Solutions/raw/refs/heads/main/\\_Learn/Download.Package/Windows.11.25H2/15a6383d188a454ca98e15931ce48dc4\\_License1.xml](https://github.com/ilikeyi/Solutions/raw/refs/heads/main/_Learn/Download.Package/Windows.11.25H2/15a6383d188a454ca98e15931ce48dc4_License1.xml)

3.2.2. 通过以上方法下载后，将获得以下 4 个文件：

3.2.2.1. [Microsoft.HEVCVideoExtension\\_8wekyb3d8bbwe.appxbundle](#)

3.2.2.2. [15a6383d188a454ca98e15931ce48dc4\\_License1.xml](#)

3.2.3. 使用 ISO 编辑工具，编辑 [26100.6584.250904-1728.ge\\_release\\_svc\\_prod1\\_amd64fre\\_InboxApps.iso](#)，将已准备好的文件

添加到 [ISO]:\packages 目录里;

B. 语言包：提取

I. 语言包：准备

挂载 26100.1.240331-1435.ge\_release\_amd64fre\_CLIENT\_LOF\_PACKAGES\_OEM.iso 或解压到任意位置;

II. 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

III. 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.11/25H2/Expand/Extract.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/Expand/Extract.ps1)

• 复制代码

```
Function Extract_Language

{

    param($Act, $NewLang, $Expand, $ISO, $SaveTo)

    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 = "Lao"; }

@{ Match = @("ml", "ml-IN"); Name = "Mlym"; }

@{ Match = @("or", "or-IN"); Name = "Orya"; }

@{ Match = @("si", "si-LK"); Name = "Sinh"; }

@{ Match = @("tr-tr", "arc-Syrc", "syr", "syr-SY", "syr-Syrc"); Name = "Syrc"; }

@{ Match = @("ta", "ta-IN", "ta-LK", "ta-MY", "ta-SG"); Name = "Taml"; }

@{ Match = @("te", "te-IN"); Name = "Telu"; }

@{ Match = @("th", "th-TH"); Name = "Thai"; }

)

ForEach ($item in $Fonts) {

    if (($item.Match) -Contains $Lang) {

        return $item.Name

    }

}

return "Not_matched"

}

Function Match_Other_Region_Specific_Requirements

{

    param($Lang)
```

```
$RegionSpecific = @(

    @{ Match = @"(zh-TW)"; Name = "Taiwan"; }

)

ForEach ($item in $RegionSpecific) {

    if (($item.Match) -Contains $Lang) {

        return $item.Name

    }

}

return "Skip_specific_packages"

}

Function Extract_Process

{

    param($Package, $Name, $NewSaveTo)

    $NewSaveTo = "$($SaveTo)\$($NewSaveTo)\Language\$($Act)\$($NewLang)"

    New-Item -Path $NewSaveTo -ItemType Directory -ErrorAction SilentlyContinue | Out-Null

    if ($ISO -eq "Auto") {

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

            ForEach ($item in $Package) {

                $TempFilePath = Join-Path -Path $_.Root -ChildPath $item -ErrorAction SilentlyContinue

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

                    Write-host "`n Find: " -NoNewLine; Write-host $TempFilePath -ForegroundColor Green

                    Write-host " Copy to: " -NoNewLine; Write-host $NewSaveTo

                    Copy-Item -Path $TempFilePath -Destination $NewSaveTo -Force

                }

            }

        }

    } else {

        ForEach ($item in $Package) {

            $TempFilePath = Join-Path -Path $ISO -ChildPath $item -ErrorAction SilentlyContinue

            Write-host "`n Find: " -NoNewline; Write-host $TempFilePath -ForegroundColor Green

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

                Write-host " Copy to: " -NoNewLine; Write-host $NewSaveTo
```

```

Copy-Item -Path $TempFilePath -Destination $NewSaveTo -Force

} else {

    Write-host "  Not found"

}

}

}

Write-host "`n  Verify the language pack file"

ForEach ($item in $Package) {

    $Path = "$($NewSaveTo)\${[IO.Path]::GetFileName($item)}"

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

        Write-host "  Discover: " -NoNewLine; Write-host $Path -ForegroundColor Green

    } else {

        Write-host "  Not found: " -NoNewLine; Write-host $Path -ForegroundColor Red

    }

}

}

$AdvLanguage = @(

    @{

        Path = "Install\Install"

        Rule = @(

            "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-Fonts-{DiyLang}-Package~31bf3856ad364e35~AMD64~~.cab"

            "LanguagesAndOptionalFeatures\Microsoft-Windows-Client-Language-Pack_x64_{Lang}.cab"

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

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

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

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

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

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

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

            "LanguagesAndOptionalFeatures\Microsoft-Windows-Notepad-FoD-Package~31bf3856ad364e35~wow64~{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~31bf3856ad364e35~AMD64~{Lang}~.cab"

```

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

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

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

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

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

"LanguagesAndOptionalFeatures\Microsoft-Windows-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\HyperV-OptionalFeature-VirtualMachinePlatform-Client-Disabled-FOD-
Package~31bf3856ad364e35~AMD64~{Lang}~.cab"

"LanguagesAndOptionalFeatures\Microsoft-Windows-DirectoryServices-ADAM-Client-FOD-
Package~31bf3856ad364e35~AMD64~{Lang}~.cab"

"LanguagesAndOptionalFeatures\Microsoft-Windows-DirectoryServices-ADAM-Client-FOD-
Package~31bf3856ad364e35~wow64~{Lang}~.cab"

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

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

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

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

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

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

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

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

"LanguagesAndOptionalFeatures\Microsoft-Windows-TerminalServices-AppServer-Client-FOD-
Package~31bf3856ad364e35~AMD64~{Lang}~.cab"

"LanguagesAndOptionalFeatures\Microsoft-Windows-TerminalServices-AppServer-Client-FOD-
Package~31bf3856ad364e35~wow64~{Lang}~.cab"

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

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

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

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

"LanguagesAndOptionalFeatures\Microsoft-Windows-WinOcr-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_OCs\{Lang}\lp.cab"

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

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

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

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

        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-connectivity_{Lang}.cab"

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

    )

}

@{

    Path = "Boot\Boot"

    Rule = @(

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

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

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

```

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

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

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

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

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

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

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

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

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

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

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

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

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

)

}

)

$NewFonts = Match_Required_Fonts -Lang $NewLang

$SpecificPackage = Match_Other_Region_Specific_Requirements -Lang $NewLang

Foreach ($item in $Expand){

    $Language = @()

    Foreach ($itemList in $AdvLanguage){

        if ($itemList.Path -eq $item){

            Foreach ($PrintLang in $itemList.Rule){

                $Language += "$($PrintLang)".Replace("{Lang}", $NewLang).Replace("{DiyLang}", $NewFonts).Replace("{Specific}", $SpecificPackage)

            }

            Extract_Process -NewSaveTo $itemList.Path -Package $Language -Name $item

        }

    }

}

$Extract_language_Pack = @(

    @{ Tag = "zh-CN"; Act = "Add"; Scope = @( "Install\Install"; "Install\WinRE"; "Boot\Boot" ) }

    @{ Tag = "en-US"; Act = "Del"; Scope = @( "Install\Install"; "Install\WinRE"; "Boot\Boot" ) }

)

```

```
ForEach ($item in $Extract_language_Pack){  
  
    Extract_Language -Act $item.Act -NewLang $item.Tag -Expand $item.Scope -ISO "Auto" -SaveTo "D:\OS_11_Custom"  
  
}
```

C. 自定义封装

I. 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
```

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
```

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。
- 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.11/25H2/Expand/Install/WinRE/WinRE.Instl.lang.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/Expand/Install/WinRE/WinRE.Instl.lang.ps1)

- 复制代码

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

$Sources = "D:\OS_11_Custom\Install\WinRE\Language\Add\zh-CN"

$Initl_install_Language_Component = @()

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

    $Initl_install_Language_Component += $_.PackageName

}

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

$Language = @(

    @{ Match = "**WinPE-LanguagePack-Package*"; File = "lp.cab"; }

    @{ Match = "**WinPE-appxdeployment*"; File = "winpe-appxdeployment_zh-CN.cab"; }

    @{ Match = "**WinPE-AppxPackaging*"; File = "winpe-appxpackaging_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 = "*connectivity*"; File = "winpe-connectivity_zh-CN.cab"; }

@{ Match = "*EnhancedStorage*"; File = "winpe-enhancedstorage_zh-CN.cab"; }

@{ Match = "*hta*"; File = "winpe-hta_zh-CN.cab"; }

@{ Match = "*Narrator*"; File = "winpe-narrator_zh-CN.cab"; }

@{ Match = "*opcservices*"; File = "winpe-opcservices_zh-CN.cab"; }

@{ Match = "*rejuv*"; File = "winpe-rejuv_zh-CN.cab"; }

@{ Match = "*scripting*"; File = "winpe-scripting_zh-CN.cab"; }

@{ Match = "*SecureStartup*"; File = "winpe-securestartup_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 = "*StorageWMI*"; File = "winpe-storagewmi_zh-CN.cab"; }

@{ Match = "*wds-tools*"; File = "winpe-wds-tools_zh-CN.cab"; }

@{ Match = "*WiFi*"; File = "winpe-wifi_zh-CN.cab"; }

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

@{ Match = "*WinPE-WMI-Package*"; File = "winpe-wmi_zh-CN.cab"; }

)

ForEach ($Rule in $Language) {

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

    ForEach ($Component in $Initl_install_Language_Component) {

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

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

            Write-host "  Language 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

                Write-host "  $($_) " -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. 组件：映像中已安装的所有包

3.1.4.3.1. 查看

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

3.1.4.3.2. 导出到 Csv

```
$SaveTo = "D:\OS_11_Custom\Install\WinRE\Report.Components.${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

3.1.5.1. 添加

前往下载页面：<https://www.catalog.update.microsoft.com/Search.aspx?q=KB5070186>，下载后保

存到：D:\OS\_11\_Custom\Install\WinRE\Update，或通过直连下载，根据架构选择下载：

3.1.5.1.1. x64，默认

- 直连下载

[https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/4bbd25e0-ac80-41bc-bd38-1bdba0bb63d6/public/windows11.0-kb5070186-x64\\_71c4a1f083355193e464c0bd3e538dd34753f8dd.cab](https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/4bbd25e0-ac80-41bc-bd38-1bdba0bb63d6/public/windows11.0-kb5070186-x64_71c4a1f083355193e464c0bd3e538dd34753f8dd.cab)

- 安装

		<pre>Add-WindowsPackage -Path "D:\OS_11_Custom\Install\WinRE\Mount" - PackagePath "D:\OS_11_Custom\Install\WinRE\Update\windows11.0- kb5070186-x64_71c4a1f083355193e464c0bd3e538dd34753f8dd.cab"</pre>
3.1.5.1.2.	arm64	<div><div><ul style="list-style-type: none"><li>直连下载</li></ul></div><div><a href="https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/842afa37-e29b-435c-9c22-36f32e81e5c3/public/windows11.0-kb5070186-arm64_04864c55245d82fec095d6677bc26b4ed70cb1bf.cab">https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/842afa37-e29b-435c-9c22-36f32e81e5c3/public/windows11.0-kb5070186-arm64_04864c55245d82fec095d6677bc26b4ed70cb1bf.cab</a></div><div><ul style="list-style-type: none"><li>安装</li></ul></div><div><pre>Add-WindowsPackage -Path "D:\OS_11_Custom\Install\WinRE\Mount" -PackagePath "D:\OS_11_Custom\Install\WinRE\Update\windows11.0-kb5070186- arm64_04864c55245d82fec095d6677bc26b4ed70cb1bf.cab"</pre></div></div>
3.1.5.2.	固化更新	<p>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。</p> <pre>Dism /image:"D:\OS_11_Custom\Install\WinRE\Mount" /cleanup-image /StartComponentCleanup /ResetBase</pre>
3.1.5.2.1.	固化更新后清理组件	<pre>\$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      }  }</pre>
3.1.6.	驱动	
3.1.7.	保存映像：WinRE.wim	<pre>Save-WindowsImage -Path "D:\OS_11_Custom\Install\WinRE\Mount"</pre>
3.1.8.	卸载映像：WinRE.wim	<p>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。</p> <pre>Dismount-WindowsImage -Path "D:\OS_11_Custom\Install\WinRE\Mount" -Discard</pre>

### 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.11/25H2/Expand/Install/WinRE/WinRE.Rebuild.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/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  Under reconstruction ".PadRight(28) -NoNewline

    try {

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

        Write-Host "Finish" -ForegroundColor Green

    } catch {

        Write-Host $_ -ForegroundColor Yellow

        Write-host $Failed -ForegroundColor Red

    }

}

Write-Host "`n  Rename: " -NoNewline -ForegroundColor Yellow

if (Test-Path "$($FileName).New" -PathType Leaf) {

    Remove-Item -Path $Filename

    Move-Item -Path "$($FileName).New" -Destination $Filename

    Write-Host "Finish" -ForegroundColor Green

} else {

    Write-host "Failed" -ForegroundColor Red

    Write-host "  $_" -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.11/25H2/Expand/Install/WinRE/WinRE.Backup.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/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

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。
- 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.11/25H2/Expand/Install/Install.Instl.lang.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging/tutorial/OS.11/25H2/Expand/Install/Install.Instl.lang.ps1)

- 复制代码

```
Function Language_Install

{

    param($Mount, $Sources, $Lang)

    $Initl_install_Language_Component = @()

    if (Test-Path $Mount -PathType Container) {

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

    } else {

        Write-Host "Not mounted: $($Mount)"

    }

    return
```

```

}

$Script:Init_Folder_All_File = @()

if (Test-Path "$($Sources)\($Lang)" -PathType Container) {

    Get-ChildItem -Path $Sources -Recurse -Include "*.cab" -ErrorAction SilentlyContinue | ForEach-Object
{ $Script:Init_Folder_All_File += $_.FullName }

    Write-host "`n  Available language pack installation files"

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

        ForEach ($item in $Script:Init_Folder_All_File) { Write-host "  $($item)" }

    } else {

        Write-host "There are no language pack files locally"

        return

    }

} else {

    Write-Host "Path does not exist: $($Sources)\($Lang)"

    return

}

$Script:Init_Folder_All_File_Match_Done = @()

$Script:Init_Folder_All_File_Exclude = @()

$Script:Search_File_Order = @(

    @{

        Name = "Fonts"

        Description = "Fonts"

        Rule = @( "*Fonts*"; )

    }

    @{

        Name = "Basic"

        Description = "Basic"

        Rule = @( "*LanguageFeatures-Basic*"; "*Client*Language*Pack*"; )

    }

    @{

        Name = "OCR"

        Description = "Optical character recognition"

        Rule = @( "*LanguageFeatures-OCR*"; )

    }

    @{

```

```

        Name = "Handwriting"

        Description = "Handwriting recognition"

        Rule = @( "LanguageFeatures-Handwriting"; )

    }

    @{

        Name = "TextToSpeech"

        Description = "Text-to-speech"

        Rule = @( "LanguageFeatures-TextToSpeech"; )

    }

    @{

        Name = "Speech"

        Description = "Speech recognition"

        Rule = @( "LanguageFeatures-Speech"; )

    }

    @{

        Name = "RegionSpecific"

        Description = "Other region-specific requirements"

        Rule = @( "InternationalFeatures"; )

    }

    @{

        Name = "Retail"

        Description = "Retail demo experience"

        Rule = @( "RetailDemo"; )

    }

    @{

        Name = "Features_On_Demand"

        Description = "Features on demand"

        Rule = @(

            "InternetExplorer*~x86~*"; "InternetExplorer*~x64~*"; "InternetExplorer*~amd64~*"; "InternetExplorer*~wow64~*";
            "InternetExplorer*~arm64.x86~*"; "InternetExplorer*~arm64~*";

            "MSPaint*~x86~*"; "MSPaint*~x64~*"; "MSPaint*~amd64~*"; "MSPaint*~wow64~*"; "MSPaint*~arm64.x86~*";
            "MSPaint*~arm64~*";

            "Notepad-FoD-Package*~x86~*"; "Notepad-FoD-Package*~x64~*"; "Notepad-FoD-Package*~amd64~*"; "Notepad-
FoD-Package*~wow64~*"; "Notepad-FoD-Package*~arm64.x86~*"; "Notepad-FoD-Package*~arm64~*";

            "Notepad-System-FoD-Package*~x86~*"; "Notepad-System-FoD-Package*~x64~*"; "Notepad-System-FoD-
Package*~amd64~*"; "Notepad-System-FoD-Package*~wow64~*"; "Notepad-System-FoD-Package*~arm64.x86~*";

```

"\*Notepad-System-FoD-Package\*~arm64~\*";

"\*MediaPlayer\*~x86~\*"; "\*MediaPlayer\*~x64~\*"; "\*MediaPlayer\*~amd64~\*"; "\*MediaPlayer\*~wow64~\*";

"\*MediaPlayer\*~arm64.x86~\*"; "\*MediaPlayer\*~arm64~\*";

"\*PowerShell\*ISE\*~x86~\*"; "\*PowerShell\*ISE\*~x64~\*"; "\*PowerShell\*ISE\*~amd64~\*"; "\*PowerShell\*ISE\*~wow64~\*";

"\*PowerShell\*ISE\*~arm64.x86~\*"; "\*PowerShell\*ISE\*~arm64~\*";

"\*StepsRecorder\*~x86~\*"; "\*StepsRecorder\*~x64~\*"; "\*StepsRecorder\*~amd64~\*"; "\*StepsRecorder\*~wow64~\*";

"\*StepsRecorder\*~arm64.x86~\*"; "\*StepsRecorder\*~arm64~\*";

"\*SnippingTool\*~x86~\*"; "\*SnippingTool\*~x64~\*"; "\*SnippingTool\*~amd64~\*"; "\*SnippingTool\*~wow64~\*";

"\*SnippingTool\*~arm64.x86~\*"; "\*SnippingTool\*~arm64~\*";

"\*WMIC\*~x86~\*"; "\*WMIC\*~x64~\*"; "\*WMIC\*~amd64~\*"; "\*WMIC\*~wow64~\*"; "\*WMIC\*~arm64.x86~\*";

"\*WMIC\*~arm64~\*";

"\*WordPad\*~x86~\*"; "\*WordPad\*~x64~\*"; "\*WordPad\*~amd64~\*"; "\*WordPad\*~wow64~\*"; "\*WordPad\*~arm64.x86~\*";

"\*WordPad\*~arm64~\*";

"\*Printing-WFS\*~x86~\*"; "\*Printing-WFS\*~x64~\*"; "\*Printing-WFS\*~amd64~\*"; "\*Printing-WFS\*~wow64~\*"; "\*Printing-WFS\*~arm64.x86~\*"; "\*Printing-WFS\*~arm64~\*";

"\*Printing-PMCPPC\*~x86~\*"; "\*Printing-PMCPPC\*~x64~\*"; "\*Printing-PMCPPC\*~amd64~\*"; "\*Printing-PMCPPC\*~wow64~\*"; "\*Printing-PMCPPC\*~arm64.x86~\*"; "\*Printing-PMCPPC\*~arm64~\*";

"\*Telnet-Client\*~x86~\*"; "\*Telnet-Client\*~x64~\*"; "\*Telnet-Client\*~amd64~\*"; "\*Telnet-Client\*~wow64~\*"; "\*Telnet-Client\*~arm64.x86~\*"; "\*Telnet-Client\*~arm64~\*";

"\*TFTP-Client\*~x86~\*"; "\*TFTP-Client\*~x64~\*"; "\*TFTP-Client\*~amd64~\*"; "\*TFTP-Client\*~wow64~\*"; "\*TFTP-Client\*~arm64.x86~\*"; "\*TFTP-Client\*~arm64~\*";

"\*VBSCRIPT\*~x86~\*"; "\*VBSCRIPT\*~x64~\*"; "\*VBSCRIPT\*~amd64~\*"; "\*VBSCRIPT\*~wow64~\*";

"\*VBSCRIPT\*~arm64.x86~\*"; "\*VBSCRIPT\*~arm64~\*";

"\*WinOcr-FOD-Package\*~x86~\*"; "\*WinOcr-FOD-Package\*~x64~\*"; "\*WinOcr-FOD-Package\*~amd64~\*"; "\*WinOcr-FOD-Package\*~wow64~\*"; "\*WinOcr-FOD-Package\*~arm64.x86~\*"; "\*WinOcr-FOD-Package\*~arm64~\*";

"\*ProjFS-OptionalFeature-FOD-Package\*~x86~\*"; "\*ProjFS-OptionalFeature-FOD-Package\*~x64~\*"; "\*ProjFS-OptionalFeature-FOD-Package\*~amd64~\*"; "\*ProjFS-OptionalFeature-FOD-Package\*~wow64~\*"; "\*ProjFS-OptionalFeature-FOD-Package\*~arm64.x86~\*"; "\*ProjFS-OptionalFeature-FOD-Package\*~arm64~\*";

"\*ServerCoreFonts-NonCritical-Fonts-BitmapFonts-FOD-Package\*~x86~\*"; "\*ServerCoreFonts-NonCritical-Fonts-BitmapFonts-FOD-Package\*~x64~\*"; "\*ServerCoreFonts-NonCritical-Fonts-BitmapFonts-FOD-Package\*~amd64~\*";

"\*ServerCoreFonts-NonCritical-Fonts-BitmapFonts-FOD-Package\*~wow64~\*"; "\*ServerCoreFonts-NonCritical-Fonts-BitmapFonts-FOD-Package\*~arm64.x86~\*"; "\*ServerCoreFonts-NonCritical-Fonts-BitmapFonts-FOD-Package\*~arm64~\*";

"\*ServerCoreFonts-NonCritical-Fonts-Support-FOD-Package\*~x86~\*"; "\*ServerCoreFonts-NonCritical-Fonts-Support-FOD-Package\*~x64~\*"; "\*ServerCoreFonts-NonCritical-Fonts-Support-FOD-Package\*~amd64~\*"; "\*ServerCoreFonts-NonCritical-Fonts-Support-FOD-Package\*~wow64~\*"; "\*ServerCoreFonts-NonCritical-Fonts-Support-FOD-Package\*~arm64.x86~\*";

"\*ServerCoreFonts-NonCritical-Fonts-Support-FOD-Package\*~arm64~\*";

"\*ServerCoreFonts-NonCritical-Fonts-TrueType-FOD-Package\*~x86~\*"; "\*ServerCoreFonts-NonCritical-Fonts-TrueType-FOD-Package\*~x64~\*"; "\*ServerCoreFonts-NonCritical-Fonts-TrueType-FOD-Package\*~amd64~\*"; "\*ServerCoreFonts-NonCritical-Fonts-TrueType-FOD-Package\*~wow64~\*"; "\*ServerCoreFonts-NonCritical-Fonts-TrueType-FOD-Package\*~arm64.x86~\*"; "\*ServerCoreFonts-NonCritical-Fonts-TrueType-FOD-Package\*~arm64~\*";

"\*SimpleTCP-FOD-Package\*~x86~\*"; "\*SimpleTCP-FOD-Package\*~x64~\*"; "\*SimpleTCP-FOD-Package\*~amd64~\*";

"\*SimpleTCP-FOD-Package\*~wow64~\*"; "\*SimpleTCP-FOD-Package\*~arm64.x86~\*"; "\*SimpleTCP-FOD-Package\*~arm64~\*";

"\*VirtualMachinePlatform-Client-Disabled-FOD-Package\*~x86~\*"; "\*VirtualMachinePlatform-Client-Disabled-FOD-Package\*~x64~\*"; "\*VirtualMachinePlatform-Client-Disabled-FOD-Package\*~amd64~\*"; "\*VirtualMachinePlatform-Client-Disabled-FOD-Package\*~wow64~\*"; "\*VirtualMachinePlatform-Client-Disabled-FOD-Package\*~arm64.x86~\*";



```
"*VirtualMachinePlatform-Client-Disabled-FOD-Package*~arm64~*";

    "*DirectoryServices-ADAM-Client-FOD-Package*~x86~*"; "*DirectoryServices-ADAM-Client-FOD-Package*~x64~*";
"*DirectoryServices-ADAM-Client-FOD-Package*~amd64~*"; "*DirectoryServices-ADAM-Client-FOD-Package*~wow64~*";
"*DirectoryServices-ADAM-Client-FOD-Package*~arm64.x86~*"; "*DirectoryServices-ADAM-Client-FOD-Package*~arm64~*";

    "*EnterpriseClientSync-Host-FOD-Package*~x86~*"; "*EnterpriseClientSync-Host-FOD-Package*~x64~*";
"*EnterpriseClientSync-Host-FOD-Package*~amd64~*"; "*EnterpriseClientSync-Host-FOD-Package*~wow64~*";
"*EnterpriseClientSync-Host-FOD-Package*~arm64.x86~*"; "*EnterpriseClientSync-Host-FOD-Package*~arm64~*";

    "*SenseClient-FoD-Package*~x86~*"; "*SenseClient-FoD-Package*~x64~*"; "*SenseClient-FoD-Package*~amd64~*";
"*SenseClient-FoD-Package*~wow64~*"; "*SenseClient-FoD-Package*~arm64.x86~*"; "*SenseClient-FoD-Package*~arm64~*";

    "*SmbDirect-FOD-Package*~x86~*"; "*SmbDirect-FOD-Package*~x64~*"; "*SmbDirect-FOD-Package*~amd64~*";
"*SmbDirect-FOD-Package*~wow64~*"; "*SmbDirect-FOD-Package*~arm64.x86~*"; "*SmbDirect-FOD-Package*~arm64~*";

    "*TerminalServices-AppServer-Client-FOD-Package*~x86~*"; "*TerminalServices-AppServer-Client-FOD-
Package*~x64~*"; "*TerminalServices-AppServer-Client-FOD-Package*~amd64~*"; "*TerminalServices-AppServer-Client-FOD-
Package*~wow64~*"; "*TerminalServices-AppServer-Client-FOD-Package*~arm64.x86~*"; "*TerminalServices-AppServer-Client-
FOD-Package*~arm64~*";

)

}

)

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

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

ForEach ($Wildcard in $Script:Init_Folder_All_File) {

    ForEach ($item in $Script:Search_File_Order) {

        ForEach ($NewRule in $item.Rule) {

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

                Write-host "`n  Fuzzy matching: " -NoNewline; Write-host $NewRule -ForegroundColor Green

                Write-host "  File name: " -NoNewline; Write-host $Wildcard -ForegroundColor Green

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

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

                $new = $OSDefaultUser + $TempSave

                if ($item.name -Contains $ExcludeNameItem) {

                    Write-host "  Do not match, install directly" -ForegroundColor Yellow

                    New-Variable -Name "Init_File_Type_$(item.Name)" -Value $new -Force

                    $Script:Init_Folder_All_File_Match_Done += $Wildcard

                } else {

                    ForEach ($Component in $Initl_install_Language_Component) {

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

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

                            New-Variable -Name "Init_File_Type_$(item.Name)" -Value $new -Force
```

```
$Script:Init_Folder_All_File_Match_Done += $WildCard

        break
    }
}

}

}

}

}

}

}

}

Write-host "`n Grouping is complete, pending installation" -ForegroundColor Yellow


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


ForEach ($WildCard in $Script:Search_File_Order) {

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

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

    if ($OSDefaultUser.Count -gt 0) {

        ForEach ($item in $OSDefaultUser) { Write-host " $($item.FileName)" -ForegroundColor Green }

    } else { Write-host " Not available" -ForegroundColor Red }

}

Write-host "`n Not matched, no longer installed" -ForegroundColor Yellow


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


ForEach ($item in $Script:Init_Folder_All_File) {

    if ($Script:Init_Folder_All_File_Match_Done -notcontains $item) {

        $Script:Init_Folder_All_File_Exclude += $item

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

    }

}

Write-host "`n Install" -ForegroundColor Yellow


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


ForEach ($WildCard in $Script:Search_File_Order) {

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

    Write-host "`n $($WildCard.Description) ( $($OSDefaultUser.Count) item )"; Write-host " $('-' * 80)"

    if ($OSDefaultUser.Count -gt 0) {

        ForEach ($item in $OSDefaultUser) {

            Write-host " File name: " -NoNewline; Write-host $item.FileName -ForegroundColor Green
```

```
Write-Host " Installing ".PadRight(22) -NoNewline

if (Test-Path $item.FileName -PathType Leaf) {

    try {

        Add-WindowsPackage -Path $Mount -PackagePath $item.FileName | Out-Null

        Write-host "Finish`n" -ForegroundColor Green

    } catch {

        Write-host "Failed" -ForegroundColor Red

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

    }

} else { Write-host "Does not exist`n" }

}

} else { Write-host " Not available`n" -ForegroundColor Red }

}

}

Language_Install -Mount "D:\OS_11_Custom\Install\Install\Mount" -Sources "D:\OS_11_Custom\Install\Install\Language\Add" -
Lang "zh-CN"
```

#### 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. 组件：映像中已安装的所有包

##### 4.3.1. 查看

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

4.3.2. 导出到 Csv

```
$SaveTo = "D:\OS_11_Custom\Install\Install\Report.Components.${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

重要:

DISM /add-provisionedappxpackage 从 Windows 10 版本 1803 开始，使用该选项时，可以预安装应用而无需将其固定到“开始”菜单 /region。预安装应用时，您可以选择将应用排除在 LayoutModification.xml 之外，这样应用将成功安装，而不会显示为“开始”菜单磁贴。如果未指定区域列表，则仅当将应用固定到“开始”布局时才会预配应用。

再次警告您，使用命令行添加必须添加参数: /Region

5.2. InBox Apps：预安装应用程序

- 编组缩写
  - 群组 A: Core, CoreSingleLanguage
  - 群组 B: CoreN
  - 群组 C: Education, Professional, ProfessionalEducation, ProfessionalWorkstation, Enterprise, IoTEnterprise, IoTEnterpriseK, ServerRdsh, ServerStandard, ServerDatacenter
  - 群组 D: EducationN, ProfessionalN, ProfessionalEducationN, ProfessionalWorkstationN, EnterpriseN, EnterpriseGN, EnterpriseSN, CloudN, CloudEN, CloudEditionLN, StarterN

	名称	产品 ID	编组
1.	Microsoft.AV1VideoExtension	9MVZQVXJBQ9V	A, C
2.	Microsoft.AVCEncoderVideoExtension	9PB0TRCNRHFX	A, C
3.	Microsoft.GamingApp	9MV0B5HZVK9Z	A, C
4.	Microsoft.HEIFImageExtension	9PMMSR1CGPWG	A, C
5.	Microsoft.HEVCVideoExtension	9NMZLZ57R3T7	A, C
6.	Microsoft.RawImageExtension	9NCTDW2W1BH8	A, C
7.	Microsoft.VP9VideoExtensions	9N4D0MSMP0PT	A, C
8.	Microsoft.WebMediaExtensions	9N5TDP8VCMHS	A, C

9.	Microsoft.WebpImageExtension	9PG2DK419DRG	A, C
10.	Microsoft.MPEG2VideoExtension	9N95Q1ZZPMH4	A, C
11.	Microsoft.Windows.DevHome	9N8MHTPHNGVV	A, C
12.	Microsoft.WindowsSoundRecorder	9WZDNCRFHWKN	A, C
13.	Microsoft.Xbox.TCUI	9NKNC0LD5NN6	A, C
14.	Microsoft.XboxGamingOverlay	9NZKPSTSNW4P	A, C
15.	Microsoft.ZuneMusic	9WZDNCRFJ3PT	A, C
16.	Microsoft.CrossDevice	9NTXGKQ8P7N0	A, C, D
17.	MSTeams	XP8BT8DW290MPQ	A, C, D
18.	MicrosoftCorporationII.MicrosoftFamily	9PDJDS743XF	A, B
19.	Microsoft.UI.Xaml.2.8		A, B, C, D
20.	Microsoft.WindowsAppRuntime.1.5		A, B, C, D
21.	Microsoft.WindowsAppRuntime.1.6		A, B, C, D
22.	Microsoft.WindowsAppRuntime.1.7		A, B, C, D
23.	Microsoft.NET.Native.Framework.2.2		A, B, C, D
24.	Microsoft.NET.Native.Runtime.2.2		A, B, C, D
25.	Microsoft.VCLibs.140.00		A, B, C, D
26.	Microsoft.VCLibs.140.00.UWPDesktop		A, B, C, D
27.	Microsoft.Services.Store.Engagement		A, B, C, D
28.	Microsoft.DesktopAppInstaller	9NBLGGH4NNS1	A, B, C, D
29.	Microsoft.WindowsStore	9WZDNCRFJBMP	A, B, C, D
30.	MicrosoftWindows.Client.WebExperience	9MSSGKG348SP	A, B, C, D
31.	Clipchamp.Clipchamp	9P1J8S7CCWWT	A, B, C, D
32.	ApplicationCompatibilityEnhancements	9PCSD6N03BKV	A, B, C, D
33.	Microsoft.BingNews	9WZDNCRFHVFW	A, B, C, D
34.	Microsoft.BingSearch	9NZBF4GT040C	A, B, C, D
35.	Microsoft.BingWeather	9WZDNCRFJ3Q2	A, B, C, D
36.	Microsoft.Copilot	9NHT9RB2F4HD	A, B, C, D
37.	Microsoft.GetHelp	9PKDZBMV1H3T	A, B, C, D
38.	Microsoft.MicrosoftOfficeHub	9WZDNCRD29V9	A, B, C, D
39.	Microsoft.MicrosoftSolitaireCollection	9WZDNCRFHWD2	A, B, C, D
40.	Microsoft.StickyNotes	9NBLGGH4QGHW	A, B, C, D
41.	Microsoft.OutlookForWindows	9NRX63209R7B	A, B, C, D
42.	Microsoft.Paint	9PCFS5B6T72H	A, B, C, D

43.	Microsoft.PowerAutomateDesktop	9NFTCH6J7FHV	A, B, C, D
44.	Microsoft.ScreenSketch	9MZ95KL8MR0L	A, B, C, D
45.	Microsoft.SecHealthUI	9NK0JLZT7JD0	A, B, C, D
46.	Microsoft.StorePurchaseApp	9NBLGGH4LS1F	A, B, C, D
47.	Microsoft.Todos	9NBLGGH5R558	A, B, C, D
48.	Microsoft.Windows.Photos	9WZDNCRFJBH4	A, B, C, D
49.	Microsoft.WindowsAlarms	9WZDNCRFJ3PR	A, B, C, D
50.	Microsoft.WindowsCalculator	9WZDNCRFHVN5	A, B, C, D
51.	Microsoft.WindowsCamera	9WZDNCRFJBBG	A, B, C, D
52.	Microsoft.WindowsFeedbackHub	9NBLGGH4R32N	A, B, C, D
53.	Microsoft.WindowsNotepad	9MSMLRH6LZF3	A, B, C, D
54.	Microsoft.WindowsTerminal	9N0DX20HK701	A, B, C, D
55.	Microsoft.XboxIdentityProvider	9WZDNCRD1HKW	A, B, C, D
56.	Microsoft.XboxSpeechToTextOverlay	9P086NHDNB9W	A, B, C, D
57.	Microsoft.YourPhone	9NMPJ99VJBWV	A, B, C, D
58.	MicrosoftCorporationII.QuickAssist	9P7BP5VNWKX5	A, B, C, D

5.3. Inbox Apps：已安装

```
$custom_array = @(); Get-AppXProvisionedPackage -Path "D:\OS_11_Custom\Install\Install\Mount" | ForEach-Object { $custom_array +=  
[PSCustomObject]@{ DisplayName = $_.DisplayName; PackageName = $_.PackageName; Version = $_.Version; Architecture =  
$_.Architecture; ResourceId = $_.ResourceId; Regions = $_.Regions; } };
```

- 优先执行以上 PS 代码后，才能选择以下 “查看” 或 “导出到 CSV” 。
- 查看已安装 Inbox Apps，请了解区域（Region）绑定，未绑定区域在执行完 OOBE 后，将不会出现在应用程序里。

5.3.1. 查看

```
$custom_array | Out-GridView
```

5.3.2. 导出到 Csv

```
$SaveTo = "D:\OS_11_Custom\Install\Install\Report.Inbox.Apps.$(Get-Date -Format "yyyyMMddHHmmss").csv"  
  
$custom_array | 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.apps.Clear.all.ps1

- [\Expand\Install\Install.Inbox.apps.Clear.all.ps1](#)
- [https://github.com/ilikeyi/solutions/blob/main/\\_Learn/Packaging.tutorial/OS.11/25H2/Expand/Install/Install.InBox.Apps.Clear.all.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/Expand/Install/Install.InBox.Apps.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. 已安装语言标记：了解

了解已安装语言标记前，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，这就是语言标记。

## 5.6. 脱机映像已安装语言标记：添加方式

向脱机映像标记：已安装语言，有二种方法，第一种安装 [Microsoft-Windows-Client-Language-Pack\\_x64\\_zh-cn.cab](#)，第二种安装“本地语言体验包（LXPs）”，微软官方向 Windows 10 提供了本地语言体验包（LXPS）安装文件，Windows 11 不再提供，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.apps](#), 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	9MXPBGNNDW3L	bg-bg	9MX54588434F
bn-bd	9PH7TKVXGGM8	bn-in	9P1M44L7W84T
bs-latn-ba	9MVFKLJ10MFL	ca-es	9P6JMKIQZ9S7
ca-es-valencia	9P9K3WMFSW90	chr-cher-us	9MX15485N3RK
cs-cz	9P3WXZ1KTM7C	cy-gb	9NKJ9TBML4HB
da-dk	9NDMT2VKS1L	de-de	9P6CT0SLW589
el-gr	9N586B13PBLD	en-gb	9NT52VQ39BVN
es-es	9NWWGWLHPB1Z	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	9NXRNBRN9B
gu-in	9P2HMSWDJDQ1	ha-latn-ng	9N1L95DBG3
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	9NWW891H6HN	lv-lv	9N5CQDPH6SQT
mi-nz	9P2GDFB3JPSX	mk-mk	9P1X6XB1K3RN
ml-in	9NWDTV8FFV7L	mn-mn	9PG1DHC4VTZW
mr-in	9MWXCXKHJVR1J	ms-my	9NPXL8ZSDDQ7
mt-mt	9PDG96SQ6BN8	nb-no	9N6J0M5DHCK0
ne-np	9P7CHPLWDQVN	nl-nl	9PF1C9NB5PRV
nn-no	9PK7KM3Z06KH	nso-za	9NS49QLX5CDV
or-in	9NTHCXCXSJDH	pa-arab-pk	9NJRL03WH6FM
pa-in	9NSNC0ZJX69B	pl-pl	9NC5HW94R0LD
prs-af	9P3NGC6X5ZQC	pt-br	9P8LBDM4FW35
pt-pt	9P7X8QJ7FL0X	quc-latn-gt	9P2V6MNNQZ0B
quz-pe	9NHTX8NVQ04K	ro-ro	9MWXGPJ5PJ3H
ru-ru	9NMJCX77QKPX	rw-rw	9NFW0M20H9WG
sd-arab-pk	9NB9JSCXW9X5	si-lk	9NVF9QSLGTL0
sk-sk	9N7LSNN099WB	sl-si	9NV27L34J4ST
sq-al	9MWLRGNMDGK7	sr-cyrl-ba	9MXGN7V65C7B
sr-cyrl-rs	9PPD6CCK9K5H	sr-latn-rs	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. a). 选择 URL (link) , b). 选择频道 (RP) , c). 输入关键词:  
<https://www.microsoft.com/store/productId/9NRMNT6GMZ70>

5.6.2.2.2.	网页内搜索 26200 内容，搜索结果： <a href="#">Microsoft.LanguageExperiencePackzh-CN_26200.*.neutral_8wekyb3d8bbwe.appx</a>	
5.6.2.2.3.	下载后保存到 <a href="#">D:\OS_11_Custom\Install\Install\Inbox.apps</a> 目录里，重命名： <a href="#">LanguageExperiencePack.zh-cn.Neutral.Appx</a>	
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 <a href="#">LanguageExperiencePack.zh-cn.Neutral.appx</a> 后，执行安装命令：		
<pre>Add-AppxProvisionedPackage -Path "D:\OS_11_Custom\Install\Install\Mount" -PackagePath "D:\OS_11_Custom\Install\Install\Inbox.apps\LanguageExperiencePack.zh-cn.Neutral.appx" -SkipLicense</pre>		
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： <a href="#">en-US</a> 、 <a href="#">zh-CN</a> ，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	
<pre>\$custom_array = @(); Get-AppXProvisionedPackage -Path "D:\OS_11_Custom\Install\Install\Mount"   ForEach-Object { \$custom_array += [PSCustomObject]@{ DisplayName = \$_.DisplayName; PackageName = \$_.PackageName; Version = \$_.Version; Architecture = \$_.Architecture; ResourceId = \$_.ResourceId; Regions = \$_.Regions; } };</pre>		
○ 优先执行以上 PS 代码后，才能选择以下“ <a href="#">查看</a> ”或“ <a href="#">导出到 CSV</a> ”。		
5.6.4.1.1.	查看	
<pre>\$custom_array   Out-GridView</pre>		
5.6.4.1.2.	导出到 Csv	
<pre>\$SaveTo = "D:\OS_11_Custom\Install\Install\Report.InBox.Apps.\$(Get-Date -Format "yyyyMMddHHmmss").csv"  \$custom_array   Export-CSV -NoType -Path \$SaveTo  Write-host \$SaveTo -ForegroundColor Green</pre>		
5.6.4.2.	查看可用的语言设置	
<pre>Dism /Image:"D:\OS_11_Custom\Install\Install\Mount" /Get-Intl</pre>		
5.7.	InBox Apps：安装	

### 5.7.1. 挂载或解压 InBox Apps 安装文件

挂载 [26100.6584.250904-1728.ge\\_release\\_svc\\_prod1\\_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.apps.ps1](#)
  - [\Expand\Install\Install.Inst.Inbox.apps.ps1](#)
  - [https://github.com/ilikeyi/solutions/blob/main/\\_Learn/Packaging/tutorial/OS.11/25H2/Expand/Install/Install.Inst.Inbox.Apps.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging/tutorial/OS.11/25H2/Expand/Install/Install.Inst.Inbox.Apps.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 = @(
```

```
#region Group 1
```

```
@{
```

```
Name = @( "Core"; "CoreSingleLanguage"; )
```

```
Apps = @(
```

```
"Microsoft.UI.Xaml.2.8"; "Microsoft.WindowsAppRuntime.1.5"; "Microsoft.WindowsAppRuntime.1.6";
```

```
"Microsoft.WindowsAppRuntime.1.7"; "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";
```

```
        "Clipchamp.Clipchamp"; "Microsoft.ApplicationCompatibilityEnhancements"; "Microsoft.AV1VideoExtension";  
        "Microsoft.AVCEncoderVideoExtension"; "Microsoft.BingNews"; "Microsoft.BingSearch"; "Microsoft.BingWeather";  
        "Microsoft.Copilot"; "Microsoft.DesktopAppInstaller";
```

```
        "Microsoft.GamingApp"; "Microsoft.GetHelp"; "Microsoft.HEIFImageExtension";  
        "Microsoft.HEVCVideoExtension"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.MicrosoftSolitaireCollection";  
        "Microsoft.MicrosoftStickyNotes"; "Microsoft.MPEG2VideoExtension";
```

```
        "Microsoft.OutlookForWindows"; "Microsoft.Paint"; "Microsoft.PowerAutomateDesktop";  
        "Microsoft.RawImageExtension"; "Microsoft.ScreenSketch"; "Microsoft.SecHealthUI"; "Microsoft.StorePurchaseApp";  
        "Microsoft.Todos"; "Microsoft.VP9VideoExtensions"; "Microsoft.WebMediaExtensions";
```

```
        "Microsoft.WebplImageExtension"; "Microsoft.Windows.DevHome"; "Microsoft.Windows.Photos";  
        "Microsoft.WindowsAlarms"; "Microsoft.WindowsCalculator"; "Microsoft.WindowsCamera";  
        "Microsoft.WindowsFeedbackHub"; "Microsoft.WindowsNotepad"; "Microsoft.WindowsSoundRecorder";
```

```
        "Microsoft.WindowsStore"; "Microsoft.WindowsTerminal"; "Microsoft.Xbox.TCUI";  
        "Microsoft.XboxGamingOverlay"; "Microsoft.XboxIdentityProvider"; "Microsoft.XboxSpeechToTextOverlay";  
        "Microsoft.YourPhone"; "Microsoft.ZuneMusic"; "MicrosoftCorporationII.MicrosoftFamily";
```

```
        "MicrosoftCorporationII.QuickAssist"; "MicrosoftWindows.Client.WebExperience"; "Microsoft.CrossDevice";  
        "MSTeams";
```

```
    )
```

```
    }
```

```
    @{
```

```
        Name = @( "CoreN" )
```

```
        Apps = @(
```

```
            "Microsoft.UI.Xaml.2.8"; "Microsoft.WindowsAppRuntime.1.3"; "Microsoft.WindowsAppRuntime.1.4";  
            "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";
```

```
            "Clipchamp.Clipchamp"; "Microsoft.ApplicationCompatibilityEnhancements"; "Microsoft.BingNews";  
            "Microsoft.BingSearch"; "Microsoft.BingWeather"; "Microsoft.Copilot"; "Microsoft.DesktopAppInstaller";  
            "Microsoft.GetHelp"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.MicrosoftSolitaireCollection";
```

```
            "Microsoft.MicrosoftStickyNotes"; "Microsoft.OutlookForWindows"; "Microsoft.Paint";  
            "Microsoft.PowerAutomateDesktop"; "Microsoft.ScreenSketch"; "Microsoft.SecHealthUI";  
            "Microsoft.StorePurchaseApp"; "Microsoft.Todos"; "Microsoft.Windows.Photos"; "Microsoft.WindowsAlarms";
```

```
            "Microsoft.WindowsCalculator"; "Microsoft.WindowsCamera"; "Microsoft.WindowsFeedbackHub";  
            "Microsoft.WindowsNotepad"; "Microsoft.WindowsStore"; "Microsoft.WindowsTerminal";  
            "Microsoft.XboxIdentityProvider"; "Microsoft.XboxSpeechToTextOverlay"; "Microsoft.YourPhone";
```

```
            "MicrosoftCorporationII.MicrosoftFamily"; "MicrosoftCorporationII.QuickAssist";  
            "MicrosoftWindows.Client.WebExperience"; "Microsoft.CrossDevice"; "MSTeams";
```

```
        )
```

```
    }
```

```
    @{
```

```
        Name = @( "Education"; "Professional"; "ProfessionalEducation"; "ProfessionalWorkstation"; "Enterprise";  
        "IoTEnterprise"; "IoTEnterpriseK"; "ServerRdsh"; "ServerStandard"; "ServerDatacenter"; )
```

```
        Apps = @(
```

```
            "Microsoft.UI.Xaml.2.8"; "Microsoft.WindowsAppRuntime.1.3"; "Microsoft.WindowsAppRuntime.1.4";
```

```
"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";

"Clipchamp.Clipchamp"; "Microsoft.ApplicationCompatibilityEnhancements"; "Microsoft.AV1VideoExtension";
"Microsoft.AVCEncoderVideoExtension"; "Microsoft.BingNews"; "Microsoft.BingSearch"; "Microsoft.BingWeather";
"Microsoft.Copilot"; "Microsoft.DesktopAppInstaller"; "Microsoft.GamingApp";

"Microsoft.GetHelp"; "Microsoft.HEIFImageExtension"; "Microsoft.HEVCVideoExtension";
"Microsoft.MicrosoftOfficeHub"; "Microsoft.MicrosoftSolitaireCollection"; "Microsoft.MicrosoftStickyNotes";
"Microsoft.MPEG2VideoExtension"; "Microsoft.OutlookForWindows"; "Microsoft.Paint";

"Microsoft.PowerAutomateDesktop"; "Microsoft.RawImageExtension"; "Microsoft.ScreenSketch";
"Microsoft.SecHealthUI"; "Microsoft.StorePurchaseApp"; "Microsoft.Todos"; "Microsoft.VP9VideoExtensions";
"Microsoft.WebMediaExtensions"; "Microsoft.WebpImageExtension";

"Microsoft.Windows.DevHome"; "Microsoft.Windows.Photos"; "Microsoft.WindowsAlarms";
"Microsoft.WindowsCalculator"; "Microsoft.WindowsCamera"; "Microsoft.WindowsFeedbackHub";
"Microsoft.WindowsNotepad"; "Microsoft.WindowsSoundRecorder"; "Microsoft.WindowsStore";

"Microsoft.WindowsTerminal"; "Microsoft.Xbox.TCUI"; "Microsoft.XboxGamingOverlay";
"Microsoft.XboxIdentityProvider"; "Microsoft.XboxSpeechToTextOverlay"; "Microsoft.YourPhone";
"Microsoft.ZuneMusic"; "MicrosoftCorporationII.QuickAssist"; "MicrosoftWindows.Client.WebExperience";

"Microsoft.CrossDevice"; "MSTeams";

)

}

@{

    Name = @( "EducationN"; "ProfessionalN"; "ProfessionalEducationN"; "ProfessionalWorkstationN";
"EnterpriseN"; "EnterpriseGN"; "EnterpriseSN"; "CloudN"; "CloudEN"; "CloudEditionLN"; "StarterN"; )

    Apps = @(

        "Microsoft.UI.Xaml.2.8"; "Microsoft.WindowsAppRuntime.1.3"; "Microsoft.WindowsAppRuntime.1.4";
"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";

"Clipchamp.Clipchamp"; "Microsoft.ApplicationCompatibilityEnhancements"; "Microsoft.BingNews";
"Microsoft.BingSearch"; "Microsoft.BingWeather"; "Microsoft.Copilot"; "Microsoft.DesktopAppInstaller";
"Microsoft.GetHelp"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.MicrosoftSolitaireCollection";

"Microsoft.MicrosoftStickyNotes"; "Microsoft.OutlookForWindows"; "Microsoft.Paint";
"Microsoft.PowerAutomateDesktop"; "Microsoft.ScreenSketch"; "Microsoft.SecHealthUI";
"Microsoft.StorePurchaseApp"; "Microsoft.Todos"; "Microsoft.Windows.Photos"; "Microsoft.WindowsAlarms";

"Microsoft.WindowsCalculator"; "Microsoft.WindowsCamera"; "Microsoft.WindowsFeedbackHub";
"Microsoft.WindowsNotepad"; "Microsoft.WindowsStore"; "Microsoft.WindowsTerminal";
"Microsoft.XboxIdentityProvider"; "Microsoft.XboxSpeechToTextOverlay"; "Microsoft.YourPhone";

"MicrosoftCorporationII.QuickAssist"; "MicrosoftWindows.Client.WebExperience"; "Microsoft.CrossDevice";
"MSTeams";

    )

}

@{

    Name = @( "EnterpriseS"; "EnterpriseSN"; "IoTEnterpriseS"; "IoTEnterpriseSK"; )

    Apps = @( "Microsoft.SecHealthUI"; )

}
```

```
}

)

Rule = @(

    @{ Name = "Microsoft.UI.Xaml.2.8"; Match = "UI.Xaml*{ARCHTag}*2.8"; License = "UI.Xaml*{ARCHTag}*2.8";
Region = "All"; Dependencies = @(); }

    @{ Name = "Microsoft.WindowsAppRuntime.1.5"; Match = "WindowsAppRuntime.{ARCHC}.1.5"; License =
"WindowsAppRuntime.{ARCHC}.1.5"; Region = "All"; Dependencies = @(); }

    @{ Name = "Microsoft.WindowsAppRuntime.1.6"; Match = "WindowsAppRuntime.{ARCHC}.1.6"; License =
"WindowsAppRuntime.{ARCHC}.1.6"; Region = "All"; Dependencies = @(); }

    @{ Name = "Microsoft.WindowsAppRuntime.1.7"; Match = "WindowsAppRuntime.{ARCHC}.1.7"; License =
"WindowsAppRuntime.{ARCHC}.1.7"; Region = "All"; Dependencies = @(); }

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

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

    @{ Name = "Microsoft.VCLibs.140.00"; Match = "Microsoft.VCLibs.{ARCHTag}.14.00.appx"; License =
"VCLibs.{ARCHTag}.14.00.appx"; Region = "All"; Dependencies = @(); }

    @{ Name = "Microsoft.VCLibs.140.00.UWPDesktop"; Match = "Microsoft.VCLibs.{ARCHTag}.14.00.UWPDesktop";
License = "Microsoft.VCLibs.{ARCHTag}.14.00.UWPDesktop"; Region = "All"; Dependencies = @(); }

    @{ Name = "Microsoft.Services.Store.Engagement"; Match = "Services.Store.Engagement*{ARCHC}"; License =
"Services.Store.Engagement*{ARCHC}"; Region = "All"; Dependencies = @(); }

    @{ Name = "Microsoft.DesktopAppInstaller"; Match = "DesktopAppInstaller"; License = "DesktopAppInstaller";
Region = "All"; Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.VCLibs.140.00.UWPDesktop"); }

    @{ Name = "Microsoft.WindowsStore"; Match = "WindowsStore"; License = "WindowsStore"; Region = "All";
Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.NET.Native.Framework.2.2",
"Microsoft.NET.Native.Runtime.2.2", "Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop"); }

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

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

    @{ Name = "Microsoft.ApplicationCompatibilityEnhancements"; Match = "ApplicationCompatibilityEnhancements";
License = "ApplicationCompatibilityEnhancements"; Region = "All"; Dependencies = @(); }

    @{ Name = "Microsoft.AV1VideoExtension"; Match = "AV1VideoExtension"; License = "AV1VideoExtension"; Region
= "All"; Dependencies = @(); }

    @{ Name = "Microsoft.AVCEncoderVideoExtension"; Match = "AVCEncoderVideoExtension"; License =
"AVCEncoderVideoExtension"; Region = "All"; Dependencies = @(); }

    @{ Name = "Microsoft.BingNews"; Match = "BingNews"; License = "BingNews"; Region = "All"; Dependencies =
@("Microsoft.UI.Xaml.2.8", "Microsoft.NET.Native.Framework.2.2", "Microsoft.NET.Native.Runtime.2.2",
"Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop"); }

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

    @{ Name = "Microsoft.BingWeather"; Match = "BingWeather"; License = "BingWeather"; Region = "All";
Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.NET.Native.Framework.2.2",
```

"Microsoft.NET.Native.Runtime.2.2", "Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name = "Microsoft.Copilot"; Match = "Copilot"; License = "Copilot"; Region = "AD;AE;AF;AG;AI;AL;AM;AO;AQ;AR;AS;AT;AU;AW;AX;AZ;BA;BB;BD;BE;BF;BG;BH;BI;BJ;BL;BM;BN;BO;BQ;BR;BS;BT;BV;BW;BZ;CA;CC;CD;CF;CG;CH;CI;CK;CL;CM;CO;CR;CV;CW;CX;CY;CZ;DE;DJ;DK;DM;DO;DZ;EC;EE;EG;ER;ES;ET;FI;FJ;FK;FM;FO;FR;GA;GB;GD;GE;GF;GG;GH;GI;GL;GM;GN;GP;GQ;GR;GS;GT;GU;GW;GY;HK;HM;HN;HR;HT;HU;ID;IE;IL;IM;IN;IO;IQ;IS;IT;JE;JM;JO;JP;KE;KG;KH;KI;KM;KN;KR;KW;KY;KZ;LA;LB;LC;LI;LK;LR;LS;LT;LU;LV;LY;MA;MC;MD;ME;MF;MG;MH;MK;ML;MM;MN;MO;MP;MQ;MR;MS;MT;MU;MV;MW;MX;MY;MZ;NA;NC;NE;NF;NG;NI;NL;NO;NP;NR;NU;NZ;OM;PA;PE;PF;PG;PH;PK;PL;PM;PN;PR;PS;PT;PW;PY;QA;RE;RO;RS;RW;SA;SB;SC;SD;SE;SG;SH;SI;SJ;SK;SL;SM;SN;SO;SR;SS;ST;SV;SX;SZ;TC;TD;TF;TG;TH;TJ;TK;TL;TM;TN;TO;TR;TT;TV;TW;TZ;UA;UG;UM;US;UY;UZ;VA;VC;VE;VG;VI;VN;VU;WF;WS;XK;YE;YT;ZA;ZM;ZW"; Dependencies = @("Microsoft.VCLibs.140.00"); }

@{ Name = "Microsoft.GamingApp"; Match = "GamingApp"; License = "GamingApp"; Region = "All"; Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop"); }

@{ Name = "Microsoft.GetHelp"; Match = "GetHelp"; License = "GetHelp"; Region = "All"; Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.NET.Native.Framework.2.2", "Microsoft.NET.Native.Runtime.2.2", "Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop"); }

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

@{ Name = "Microsoft.HEVCVideoExtension"; Match = "HEVCVideoExtension"; License = "15a6383d188a454ca98e15931ce48dc4\_License1"; Region = "All"; Dependencies = @("Microsoft.VCLibs.140.00"); }

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

@{ Name = "Microsoft.MicrosoftSolitaireCollection"; Match = "MicrosoftSolitaireCollection"; License = "MicrosoftSolitaireCollection"; Region = "All"; Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.NET.Native.Framework.2.2", "Microsoft.NET.Native.Runtime.2.2", "Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop"); }

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

@{ Name = "Microsoft.MPEG2VideoExtension"; Match = "MPEG2VideoExtension"; License = "MPEG2VideoExtension"; Region = "All"; Dependencies = @(); }

@{ Name = "Microsoft.OutlookForWindows"; Match = "Microsoft.OutlookForWindows\_8wekyb3d8bbwe"; License = "Microsoft.OutlookForWindows\_8wekyb3d8bbwe"; Region = "AD;AE;AF;AG;AI;AL;AM;AO;AQ;AR;AS;AT;AU;AW;AX;AZ;BA;BB;BD;BE;BF;BG;BH;BI;BJ;BL;BM;BN;BO;BQ;BR;BS;BT;BV;BW;BY;BZ;CA;CC;CD;CF;CG;CH;CI;CK;CL;CM;CO;CR;CU;CV;CW;CX;CY;CZ;DE;DJ;DK;DM;DO;DZ;EC;EE;EG;ER;ES;ET;FI;FJ;FK;FM;FO;FR;GA;GB;GD;GE;GF;GG;GH;GI;GL;GM;GN;GP;GQ;GR;GS;GT;GU;GW;GY;HK;HM;HN;HR;HT;HU;ID;IE;IL;IM;IN;IO;IQ;IR;IS;IT;JE;JM;JO;JP;KE;KG;KH;KI;KM;KN;KP;KR;KW;KY;KZ;LA;LB;LC;LI;LK;LR;LS;LT;LU;LV;LY;MA;MC;MD;ME;MF;MG;MH;MK;ML;MM;MN;MO;MP;MQ;MR;MS;MT;MU;MV;MW;MX;MY;MZ;NA;NC;NE;NF;NG;NI;NL;NO;NP;NR;NU;NZ;OM;PA;PE;PF;PG;PH;PK;PL;PM;PN;PR;PS;PT;PW;PY;QA;RE;RO;RS;RU;RW;SA;SB;SC;SD;SE;SG;SH;SI;SJ;SK;SL;SM;SN;SO;SR;SS;ST;SV;SX;SY;SZ;TC;TD;TF;TG;TH;TJ;TK;TL;TM;TN;TO;TR;TT;TV;TW;TZ;UA;UG;UM;US;UY;UZ;VA;VC;VE;VG;VI;VN;VU;WF;WS;XK;YE;YT;ZA;ZM;ZW"; Dependencies = @(); }

@{ Name = "Microsoft.Paint"; Match = "Paint"; License = "Paint"; Region = "All"; Dependencies = @("Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop", "Microsoft.UI.Xaml.2.8"); }

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

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

@{ Name = "Microsoft.ScreenSketch"; Match = "ScreenSketch"; License = "ScreenSketch"; Region = "All";



```
Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.WindowsAppRuntime.1.3", "Microsoft.VCLibs.140.00",
"Microsoft.VCLibs.140.00.UWPDesktop"); }

    @{ Name = "Microsoft.SecHealthUI"; Match = "SecHealthUI*{ARCHC}"; License = "SecHealthUI*{ARCHC}"; Region
= "All"; Dependencies = @(); }

    @{ Name = "Microsoft.StorePurchaseApp"; Match = "StorePurchaseApp"; License = "StorePurchaseApp"; Region =
"All"; Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.NET.Native.Framework.2.2",
"Microsoft.NET.Native.Runtime.2.2", "Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop"); }

    @{ Name = "Microsoft.Todos"; Match = "Todos"; License = "Todos"; Region = "All"; Dependencies =
@("Microsoft.UI.Xaml.2.8", "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"); }

    @{ Name = "Microsoft.VP9VideoExtensions"; Match = "VP9VideoExtensions"; License = "VP9VideoExtensions";
Region = "All"; Dependencies = @(); }

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

    @{ Name = "Microsoft.WebplImageExtension"; Match = "WebplImageExtension"; License = "WebplImageExtension";
Region = "All"; Dependencies = @(); }

    @{ Name = "Microsoft.Windows.DevHome"; Match = "DevHome"; License = "DevHome"; Region = "All";
Dependencies = @("Microsoft.WindowsAppRuntime.1.3", "Microsoft.VCLibs.140.00.UWPDesktop"); }

    @{ Name = "Microsoft.Windows.Photos"; Match = "Windows.Photos"; License = "Windows.Photos"; Region = "All";
Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop"); }

    @{ Name = "Microsoft.WindowsAlarms"; Match = "WindowsAlarms"; License = "WindowsAlarms"; Region = "All";
Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.NET.Native.Framework.2.2",
"Microsoft.NET.Native.Runtime.2.2", "Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop",
"Microsoft.WindowsAppRuntime.1.3"); }

    @{ Name = "Microsoft.WindowsCalculator"; Match = "WindowsCalculator"; License = "WindowsCalculator"; Region
= "All"; Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.NET.Native.Framework.2.2",
"Microsoft.NET.Native.Runtime.2.2", "Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop"); }

    @{ Name = "Microsoft.WindowsCamera"; Match = "WindowsCamera"; License = "WindowsCamera"; Region = "All";
Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.NET.Native.Framework.2.2",
"Microsoft.NET.Native.Runtime.2.2", "Microsoft.VCLibs.140.00"); }

    @{ Name = "Microsoft.WindowsFeedbackHub"; Match = "WindowsFeedbackHub"; License =
"WindowsFeedbackHub"; Region = "All"; Dependencies = @("Microsoft.UI.Xaml.2.8",
"Microsoft.NET.Native.Framework.2.2", "Microsoft.NET.Native.Runtime.2.2", "Microsoft.VCLibs.140.00",
"Microsoft.VCLibs.140.00.UWPDesktop"); }

    @{ Name = "Microsoft.WindowsNotepad"; Match = "WindowsNotepad"; License = "WindowsNotepad"; Region =
"All"; Dependencies = @("Microsoft.UI.Xaml.2.8", "Microsoft.VCLibs.140.00",
"Microsoft.VCLibs.140.00.UWPDesktop"); }

    @{ Name = "Microsoft.WindowsSoundRecorder"; Match = "WindowsSoundRecorder"; License =
"WindowsSoundRecorder"; Region = "All"; Dependencies = @("Microsoft.UI.Xaml.2.8",
"Microsoft.NET.Native.Framework.2.2", "Microsoft.NET.Native.Runtime.2.2", "Microsoft.VCLibs.140.00",
"Microsoft.VCLibs.140.00.UWPDesktop"); }

    @{ Name = "Microsoft.WindowsTerminal"; Match = "WindowsTerminal"; License = "WindowsTerminal"; Region =
"All"; Dependencies = @("Microsoft.UI.Xaml.2.8"); }

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



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

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

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

```
@{ Name = "Microsoft.YourPhone"; Match = "YourPhone"; License = "YourPhone"; Region = "All"; Dependencies =  
@("Microsoft.WindowsAppRuntime.1.4", "Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop"); }
```

```
@{ Name = "Microsoft.ZuneMusic"; Match = "ZuneMusic"; License = "ZuneMusic"; Region = "All"; Dependencies =  
@("Microsoft.UI.Xaml.2.8", "Microsoft.NET.Native.Framework.2.2", "Microsoft.NET.Native.Runtime.2.2",  
"Microsoft.VCLibs.140.00"); }
```

```
@{ Name = "MicrosoftCorporationII.QuickAssist"; Match = "QuickAssist"; License = "QuickAssist"; Region = "All";  
Dependencies = @("Microsoft.VCLibs.140.00.UWPDesktop"); }
```

```
@{ Name = "Microsoft.CrossDevice"; Match = "CrossDevice"; License = "CrossDevice"; Region = "All";  
Dependencies = @("Microsoft.VCLibs.140.00", "Microsoft.VCLibs.140.00.UWPDesktop"); }
```

```
@{ Name = "MSTeams"; Match = "MSTeams*{ARCHC}"; License = "MSTeams*{ARCHC}"; Region =  
"AD;AE;AF;AG;AI;AL;AM;AO;AQ;AR;AS;AT;AU;AW;AX;AZ;BA;BB;BD;BE;BF;BG;BH;BI;BJ;BL;BM;BN;BO;BQ;BR;BS;BT;BV;B  
W;BY;BZ;CA;CC;CD;CF;CG;CH;CI;CK;CL;CM;CO;CR;CU;CV;CW;CX;CY;CZ;DE;DJ;DK;DM;DO;DZ;EC;EE;EG;ER;ES;ET;FI  
;FJ;FK;FM;FO;FR;GA;GB;GD;GE;GF;GG;GH;GI;GL;GM;GN;GP;GQ;GR;GS;GT;GU;GW;GY;HK;HM;HN;HR;HT;HU;ID;IE;IL;I  
M;IN;IO;IQ;IR;IS;IT;JE;JM;JO;JP;KE;KG;KH;KI;KM;KN;KP;KR;KW;KY;KZ;LA;LB;LC;LI;LK;LR;LS;LT;LU;LV;LY;MA;MC;MD;ME;  
MF;MG;MH;MK;ML;MM;MN;MO;MP;MQ;MR;MS;MT;MU;MV;MW;MX;MY;MZ;NA;NC;NE;NF;NG;NI;NL;NO;NP;NR;NU;NZ;  
OM;PA;PE;PF;PG;PH;PK;PL;PM;PN;PR;PS;PT;PW;PY;QA;RE;RO;RS;RU;RW;SA;SB;SC;SD;SE;SG;SH;SI;SJ;SK;SL;SM;SN;S  
O;SR;SS;ST;SV;SX;SY;SZ;TC;TD;TF;TG;TH;TJ;TK;TL;TM;TN;TO;TR;TT;TV;TW;TZ;UA;UG;UM;US;UY;UZ;VA;VC;VE;VG;VI;VN;  
VU;WF;WS;XK;YE;YT;ZA;ZM;ZW"; Dependencies = @("Microsoft.WindowsAppRuntime.1.3", "Microsoft.VCLibs.140.00",  
"Microsoft.VCLibs.140.00.UWPDesktop"); }
```

```
@{ Name = "MicrosoftCorporationII.MicrosoftFamily"; Match = "MicrosoftFamily"; License = "MicrosoftFamily";  
Region = "All"; Dependencies = @("Microsoft.VCLibs.140.00.UWPDesktop"); }
```

```
)
```

```
}
```

```
$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", "*.msix" -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

                        Region    = $itemInBoxApps.Region

                        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: ".PadRight(22) -NoNewline; Write-host $Rule.Name -ForegroundColor Yellow

    Write-host " Region: ".PadRight(22) -NoNewline; Write-host $Rule.Region -ForegroundColor Yellow

    Write-host " Apps installer: ".PadRight(22) -NoNewline; Write-host $Rule.InstallPacker -ForegroundColor Yellow

    Write-host " License: ".PadRight(22) -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) {

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

    if ([string]::IsNullOrEmpty($Rule.Certificate)) {

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

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

        try {

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

            Write-Host "Done`n" -ForegroundColor Green

        } catch { Write-Host "Failed" -ForegroundColor Red; Write-Host "  $($_)`n" -ForegroundColor Red; }

    } else {

        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 -Regions $Rule.Region -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 -Regions
$Rule.Region -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. 脱机映像已安装语言标记：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.11/25H2/Expand/Install/Install.Clear.Flag.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/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。

```
Optimize-AppXProvisionedPackages -Path "D:\OS_11_Custom\Install\Install\Mount"
```

## 6. 累积更新

- 不同版本、旧版本升级到最新版时，需优先添加“检查点更新”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. 功能启用包

- 你添加最新累积更新前，需要优先添加功能启用包，否则无法添加最新的累积更新，前往下载页面：

<https://www.catalog.update.microsoft.com/Search.aspx?q=KB5054156>，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：

6.1.1. x64 eKB，默认

- 直连下载

[https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/67731ce0-1988-426a-afa3-044a032eb6c6/public/windows11.0-kb5054156-x64\\_9fd13360d2c5af23ec4f591b86f1c1db37aada37.msu](https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/67731ce0-1988-426a-afa3-044a032eb6c6/public/windows11.0-kb5054156-x64_9fd13360d2c5af23ec4f591b86f1c1db37aada37.msu)

- 添加

```
Add-WindowsPackage -Path "D:\OS_11_Custom\Install\Install\Mount" -PackagePath  
"D:\OS_11_Custom\Install\Install\Update\windows11.0-kb5054156-  
x64_9fd13360d2c5af23ec4f591b86f1c1db37aada37.msu"
```

6.1.2. Arm64 eKB

- 直连下载

[https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/2e42e2b9-993c-4cc5-a8c5-6e20fc64b17e/public/windows11.0-kb5054156-arm64\\_e57cf6837a8b9877786ec3e7a366503a6072f5e5.msu](https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/2e42e2b9-993c-4cc5-a8c5-6e20fc64b17e/public/windows11.0-kb5054156-arm64_e57cf6837a8b9877786ec3e7a366503a6072f5e5.msu)

- 添加

```
Add-WindowsPackage -Path "D:\OS_11_Custom\Install\Install\Mount" -PackagePath  
"D:\OS_11_Custom\Install\Install\Update\windows11.0-kb5054156-  
arm64_e57cf6837a8b9877786ec3e7a366503a6072f5e5.msu"
```

6.2. 初始版本累积更新

- 添加功能启用包后，再添加初始版本相同的累积更新，否则无法添加最新的累积更新，前往下载页面：

<https://www.catalog.update.microsoft.com/Search.aspx?q=KB5043080>，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/d8b7f92b-bd35-4b4c-96e5-46ce984b31e0/public/windows11.0-kb5043080-x64\\_953449672073f8fb99badb4cc6d5d7849b9c83e8.msu](https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/d8b7f92b-bd35-4b4c-96e5-46ce984b31e0/public/windows11.0-kb5043080-x64_953449672073f8fb99badb4cc6d5d7849b9c83e8.msu)

- 添加

```
Add-WindowsPackage -Path "D:\OS_11_Custom\Install\Install\Mount" -PackagePath  
"D:\OS_11_Custom\Install\Install\Update\windows11.0-kb5043080-  
x64_953449672073f8fb99badb4cc6d5d7849b9c83e8.msu"
```

6.2.2. Arm64

- 直连下载

[https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/f263aa64-f367-42f0-9cad-328f342b93f7/public/windows11.0-kb5043080-arm64\\_df540a05f9b118e339c5520f4090bb5d450f090b.msu](https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/f263aa64-f367-42f0-9cad-328f342b93f7/public/windows11.0-kb5043080-arm64_df540a05f9b118e339c5520f4090bb5d450f090b.msu)

- 添加

```
Add-WindowsPackage -Path "D:\OS_11_Custom\Install\Install\Mount" -PackagePath  
"D:\OS_11_Custom\Install\Install\Update\windows11.0-kb5043080-  
arm64_df540a05f9b118e339c5520f4090bb5d450f090b.msu"
```

6.3. 其它版本

- 顺序：添加功能启用包（KB5054156）、添加初始版本累积更新（KB5043080）后，再添加其它累积更新，否则无法安装。

- 查阅“Windows 11 版本信息”，例如下载累积更新：KB5065789，版本号：26200.6725，前往下载页面：

<https://www.catalog.update.microsoft.com/Search.aspx?q=KB5065789>，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：

6.3.1. x64，默认

- 直连下载

[https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/72f34af9-6ab3-4bb0-9dfe-103e56a305fb/public/windows11.0-kb5044284-x64\\_d7eb7ceaa4798b92b988fd7dcc7c6bb39476ccf3.msu](https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/72f34af9-6ab3-4bb0-9dfe-103e56a305fb/public/windows11.0-kb5044284-x64_d7eb7ceaa4798b92b988fd7dcc7c6bb39476ccf3.msu)

- 添加

```
Add-WindowsPackage -Path "D:\OS_11_Custom\Install\Install\Mount" -PackagePath  
"D:\OS_11_Custom\Install\Install\Update\windows11.0-kb5044284-  
x64_d7eb7ceaa4798b92b988fd7dcc7c6bb39476ccf3.msu"
```

6.3.2. Arm64

- 直连下载

[https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/02370eb1-b77a-4986-8fa7-5f834b10133f/public/windows11.0-kb5044284-arm64\\_c2971fdbcc0bc11bf914c23465e88e812614d8c4.msu](https://catalog.sf.dl.delivery.mp.microsoft.com/filestreamingservice/files/02370eb1-b77a-4986-8fa7-5f834b10133f/public/windows11.0-kb5044284-arm64_c2971fdbcc0bc11bf914c23465e88e812614d8c4.msu)

- 添加

```
Add-WindowsPackage -Path "D:\OS_11_Custom\Install\Install\Mount" -PackagePath  
"D:\OS_11_Custom\Install\Install\Update\windows11.0-kb5044284-  
arm64_c2971fdbcc0bc11bf914c23465e88e812614d8c4.msu"
```

#### 6.4. 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
```

##### 6.4.1. 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.11/25H2/Expand/Install/Install.WinRE.Replace.wim.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/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

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.11/25H2/Expand/Install/Install.WinRE.Replace.wim.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/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.11/25H2/Expand/Install/Install.Rebuild.wim.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/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

}
```

## 15. 拆分、合并、压缩、互转

固实压缩为 ESD 文件格式，如果文件超过 4GB 时：无法使用拆分、无法复制到 FAT32 格式的磁盘，这是缺点。

使用 FAT32 格式存放 Windows 安装引导是最佳的解决方案，如果遇到 Install.wim 文件超过 4GB 时无法复制到 FAT32 格式的磁盘时，这时你需要将 Install.wim 拆分，小于 4GB 文件大小后才能复制到 FAT32 格式的磁盘里。

学会如何拆分和合并，固实压缩和互转，尤为重要。

### 15.1. 拆分和合并

#### 15.1.1. 拆分

将 Install.wim 拆分为 4GB 文件大小后，获得新的文件名 Install.\*.swm 后，删除旧的 Install.wim。

- Install.Split.ps1
  - [\Expand\Install\Install.Split.ps1](#)
  - [https://github.com/ilikeyi/solutions/blob/main/\\_Learn/Packaging.tutorial/OS.11/25H2/Expand/Install/Install.Split.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/Expand/Install/Install.Split.ps1)

- 复制代码

```
Write-host "Split Install.wim into Install.*.swm";

Write-host "Splitting" -NoNewline;

Split-WindowsImage -ImagePath "D:\OS_11\sources\install.wim" -SplitImagePath "D:\OS_11\sources\install.swm" -
FileSize "4096" -CheckIntegrity -ErrorAction SilentlyContinue | Out-Null

Write-Host "Split Complete`n" -ForegroundColor Green

Write-host "`nVerify completion and delete old files"

if (Test-Path -Path "D:\OS_11\sources\install.swm" -PathType leaf) {

    Remove-Item -Path "D:\OS_11\sources\install.wim"

    Write-Host "Done" -ForegroundColor Green

} else {

    Write-Host "Failed" -ForegroundColor Red

}
```

#### 15.1.2. 合并

把所有 Install.\*.swm 合并为 Install.wim 后，删除旧的 Install.\*.swm。

- Install.Merging.ps1

- [\Expand\Install\Install.Merging.ps1](#)
- [https://github.com/ilikeyi/solutions/blob/main/\\_Learn/Packaging.tutorial/OS.11/25H2/Expand/Install/Install.Merging.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/Expand/Install/Install.Merging.ps1)

- 复制代码

```
Write-host "Merge all Install.*.swm files into Install.wim";

Get-WindowsImage -ImagePath "D:\OS_11\Sources\install.swm" -ErrorAction SilentlyContinue | ForEach-Object {

    Write-Host "Image Name: " -NoNewline; Write-Host $_.ImageName -ForegroundColor Yellow;

    Write-Host "Index Number: " -NoNewline; Write-Host $_.ImageIndex -ForegroundColor Yellow;

    Write-Host "Exporting".PadRight(28) -NoNewline

    dism /export-image /SourceImageFile:"D:\OS_11\Sources\install.swm" /swmfile:"D:\OS_11\sources\install*.swm"
    /SourceIndex:"$(($_.ImageIndex))" /DestinationImageFile:"D:\OS_11\Sources\install.wim" /Compress:"Max"
    /CheckIntegrity

    Write-Host "Export Complete`n" -ForegroundColor Green

}

Write-host "`nVerify completion and delete old files"

if (Test-Path -Path "D:\OS_11\Sources\install.wim" -PathType leaf) {

    Get-ChildItem -Path "D:\OS_11\sources" -Recurse -include "*.swm" | ForEach-Object {

        Write-Host "Delete: $(($_.Fullname))" -ForegroundColor Green

        Remove-Item -Path $_.Fullname

    }

    Write-Host "Done" -ForegroundColor Green

} else {

    Write-Host "Falied" -ForegroundColor Green

}
```

## 15.2. 固实压缩 ESD 格式和互转 WIM 格式

### 15.2.1. 固实压缩

固实压缩后可以编辑版本信息和应用文件等；不可以挂载映像等，获得新的文件 install.esd 后，删除旧的 Install.wim。

- Install.Compress.ps1
  - [\Expand\Install\Install.Compress.ps1](#)
  - [https://github.com/ilikeyi/solutions/blob/main/\\_Learn/Packaging.tutorial/OS.11/25H2/Expand/Install/Install.Compress.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/Expand/Install/Install.Compress.ps1)

- 复制代码

```

Write-host "Solid compressed Install.wim";

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 "Index Number: " -NoNewline; Write-Host $_.ImageIndex -ForegroundColor Yellow;

    Write-Host "Compressing".PadRight(28) -NoNewline

    dism /export-image /SourceImageFile:"D:\OS_11\Sources\install.wim" /SourceIndex:"$($_.ImageIndex)"
/DestinationImageFile:"D:\OS_11\Sources\install.esd" /Compress:recovery /CheckIntegrity

    Write-Host "Compression completed`n" -ForegroundColor Green

}

Write-host "`nVerify completion and delete old files"

if (Test-Path -Path "D:\OS_11\Sources\install.esd" -PathType leaf) {

    Remove-Item -Path "D:\OS_11\Sources\install.wim"

    Write-Host "Done" -ForegroundColor Green

} else {

    Write-Host "Falied" -ForegroundColor Green

}

```

#### 15.2.2. 压缩文件转换为 WIM 文件格式

- Install.Convert.ps1
  - [\Expand\Install\Install.Convert.ps1](#)
  - [https://github.com/ilikeyi/solutions/blob/main/\\_Learn/Packaging.tutorial/OS.11/25H2/Expand/Install/Install.Convert.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/Expand/Install/Install.Convert.ps1)
- 复制代码

```

Write-host "Convert ESD to WIM";

Get-WindowsImage -ImagePath "D:\OS_11\Sources\install.esd" -ErrorAction SilentlyContinue | ForEach-Object {

    Write-Host "Image Name: " -NoNewline; Write-Host $_.ImageName -ForegroundColor Yellow;

    Write-Host "Index Number: " -NoNewline; Write-Host $_.ImageIndex -ForegroundColor Yellow;

    Write-Host "Exporting".PadRight(28) -NoNewline

    try {

        Export-WindowsImage -SourceImagePath "D:\OS_11\Sources\install.esd" -SourceIndex $_.ImageIndex -
DestinationImagePath "D:\OS_11\Sources\install.wim" -CompressionType "Max" -CheckIntegrity -ErrorAction
SilentlyContinue | Out-Null

        Write-Host "Done`n" -ForegroundColor Green

    } catch {

        Write-Host $_ -ForegroundColor Yellow
    }
}

```

```
Write-host "Falied`n" -ForegroundColor Red

}

}

Write-host "`nVerify completion and delete old files"

if (Test-Path -Path "D:\OS_11\Sources\install.wim" -PathType leaf) {

    Remove-Item -Path "D:\OS_11\Sources\install.esd"

    Write-Host "Done" -ForegroundColor Green

} else {

    Write-Host "Falied" -ForegroundColor Green

}
```

## II. 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
```

### 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。
- 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.11/25H2/Expand/Boot/Boot.Instl.lang.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/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 = @(

    @{ Match = "*WinPE-LanguagePack-Package*"; File = "lp.cab"; }

    @{ Match = "*WinPE*Setup*Client*Package*"; File = "WINPE-SETUP-CLIENT_zh-CN.CAB"; }

    @{ Match = "*WinPE*Setup*Package*"; File = "WinPE-Setup_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 = "*SecureStartup*"; File = "winpe-securestartup_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 = "*WinPE-WMI-Package*"; File = "winpe-wmi_zh-CN.cab"; }

)

ForEach ($Rule in $Language) {

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

    ForEach ($Component in $InitL_install_Language_Component) {

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

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

            Write-host "  Language 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

            } catch {
```

```
Write-host "Finish" -ForegroundColor Green

} catch {

Write-host "Failed" -ForegroundColor Red

Write-host " $($_) " -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. 组件：映像中已安装的所有包

4.3.1. 查看

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

4.3.2. 导出到 Csv

```
$SaveTo = "D:\OS_11_Custom\Boot\Boot\Report.Components.$(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
```

4.4. 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
```

4.5. 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。



4.5.1. 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"
```

4.5.2. 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

```
Add-WindowsPackage -Path "D:\OS_11_Custom\Boot\Boot\Mount" -PackagePath "D:\OS_11_Custom\Boot\Boot\Update\KB_Boot.cab"
```

5.2. 删除

```
Remove-WindowsPackage -Path "D:\OS_11_Custom\Boot\Boot\Mount" -PackagePath  
"D:\OS_11_Custom\Boot\Boot\Update\KB_Boot.cab"
```

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.11/25H2/Expand/Boot/Boot.Bypass.TPM.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/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
```

## III. 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>%OSDSystemLocale%/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

I. 生成 ISO

1. 下载 OScdimg

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

1.1. x64

[https://github.com/ilikeyi/Solutions/raw/refs/heads/main/\\_Encapsulation/Modules/AIO/Oscding/amd64/oscdimg.exe](https://github.com/ilikeyi/Solutions/raw/refs/heads/main/_Encapsulation/Modules/AIO/Oscding/amd64/oscdimg.exe)

1.2. x86

[https://github.com/ilikeyi/Solutions/raw/refs/heads/main/\\_Encapsulation/Modules/AIO/Oscding/x86/oscdimg.exe](https://github.com/ilikeyi/Solutions/raw/refs/heads/main/_Encapsulation/Modules/AIO/Oscding/x86/oscdimg.exe)



1.3. arm64

[https://github.com/ilikeyi/Solutions/raw/refs/heads/main/\\_Encapsulation/Modules/AIO/Oscdimg/arm64/oscdimg.exe](https://github.com/ilikeyi/Solutions/raw/refs/heads/main/_Encapsulation/Modules/AIO/Oscdimg/arm64/oscdimg.exe)

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

- ISO.ps1
  - [\Expand\ISO.ps1](#)
  - [https://github.com/ilikeyi/solutions/blob/main/\\_Learn/Packaging.tutorial/OS.11/25H2/Expand/ISO.ps1](https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.11/25H2/Expand/ISO.ps1)

- 复制代码

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

$ISO = "D:\OS_11"

$Volume = "OS_11"

$SaveTo = "D:\OS_11.iso"

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

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

II. 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 常见问题

I. 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
```

II. 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
```

III. 清理

封装过程中会产生大量的临时文件，安装 InBox Apps 应用、安装累积更新、安装语言包时会临时释放安装文件，所以不定期清理过时的会长期占用大量的磁盘空间，建议您尝试以下方法来实现清理计划，以达到释放更多的空间：

1. 常见日志

1.1. 使用命令行清理

```
$TempPaths = @( $env:Temp; "$($env:SystemRoot)\Logs\DISM"; )

foreach ($TempPath in $TempPaths) {

    if (Test-Path -Path $TempPath) {

        write-host " $($TempPath)" -ForegroundColor Green

        Get-ChildItem -Path $TempPath -Recurse -Force | ForEach-Object {

            try{

                Remove-Item $_.FullName -Force -Recurse -ErrorAction SilentlyContinue | Out-Null

            } catch {

                write-host $_ -ForegroundColor Red

            }

        }

    }

}
```

1.2. 手动删除

1.2.1. DISM 日志

使用“磁盘清理”功能，无法清理 DISM 产生的日志，需手动删除，路径：[{系统盘}:\Windows\Logs\DISM](#)

1.2.2. 临时目录

使用“磁盘清理”功能，无法清理临时目录的文件，需要手动操作，运行：[%Temp%](#)可快速定位并打开临时目录，路径：

[{系统盘}:\Users\{用户名}\AppData\Local\Temp](#)

### 1.2.3. 清理“终端”运行的命令行记录

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

执行清理命令行记录后，需重新启动“终端”生效。

## 2. 磁盘清理

运行 `cleanmgr`，选择要清理的磁盘和类型。

## 章节 3 已知问题

### 1. 使用 Windows 11 25H2（26200.6584）初始版本添加“累积更新”后“固化更新”会导致“健康”问题

测试文件：[en-us\\_windows\\_11\\_business\\_editions\\_version\\_25h2\\_x64\\_dvd\\_41c521e7.iso](#)，在封装过程中添加累积更新（KB5070311“26200.7296”）后，检查组件健康是正常的，未固化更新前无任何问题。

使用固化更新命令“`Dism /Image:"D:\OS_11" /cleanup-image /StartComponentCleanup /ResetBase`”后，检查健康状态为“损坏”。已知解决方案：

- 使用 Windows 11 24H2（26100.1742）初始版本制作即可解决固化更新后的健康问题。

### 2. Windows 安全中心

在测试全新安装初始版本 Windows 11 25H2 26200.6584 中文版时，映像文件：[zh-cn\\_windows\\_11\\_consumer\\_editions\\_version\\_25h2\\_x64\\_dvd\\_0549fc93.iso](#)，安装系统完成后打开 Windows 安全中心是英文版，语言包未汉化。该问题从 Windows 11 24H2 的 5 月 6 月份的镜像就已经开始出现，脱机和在线安装“简体中文体验包”并不能解决问题。已知解决方案：

- 打开 Windows 更新，更新推送后即可解决。



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, ilikayi@outlook.com

文档版本：1.1

首次公开发行时间：10 / 2025

文档里包含的所有脚本，最后测试时间：10 / 2025

文档最近更新日期：11 / 2025

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