Yi' Solutions

Microsoft Windows 10 22H2

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

在这背后藏着很多的隐藏故事，想解开这些，你准备好开始尝试封装了吗?

摘要

目录

章节 1　　封装

A.　　先决条件

Ⅱ　　ISO 工具

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

Ⅲ　　要求

1.　　系统安装包

关键词：迭代、跨版本、大版本、累积更新、初始版本

1.1.　　说明

1.1.1.　　每版本更新时请重新制作镜像，例如从 21H1 跨越到 22H2 时，应避免出现其它兼容性问题请勿在旧镜像基础上制作；

1.1.2.　　该条例已经在某些 OEM 厂商，通过各种形式向封装师明确传达了该法令，不允许直接从迭代版本里直接升级；

1.1.3.　　制作中请使用 "初始版本"、"开发者版" 制作。微软官方文档里曾短暂的出现过在制作中必须使用初始版本，后来这句在官方文档里却被删除了；

1.1.4.　　安装语言包后，必须重新添加累积更新（可同一版本号），不添加累积更新会出现"乱码"、"界面闪退"等问题。

1.1.5.　　进化过程：Windows 10 22H2, Build 19045.2006 + KB5027215 = OS Build 19045.3086

1.2.　　准备下载初始版本或开发者版本

1.2.1.　　x64

1.2.1.1.　　en-us_windows_10_business_editions_version_22h2_x64_dvd_8cf17b79.iso

1.2.1.2.　　en-us_windows_10_consumer_editions_version_22h2_x64_dvd_8da72ab3.iso

1.2.1.3.　　en-us_windows_10_iot_enterprise_version_22h2_x64_dvd_51cc370f.iso

1.2.2.　　x86

1.2.2.1.　　en-us_windows_10_business_editions_version_22h2_x86_dvd_186a68c3.iso

1.2.2.2.　　en-us_windows_10_consumer_editions_version_22h2_x86_dvd_90883feb.iso

1.2.3.　　arm64

1.2.3.1.　　en-us_windows_10_iot_enterprise_version_22h2_arm64_dvd_39566b6b.iso

1.3.　　示例下载 en-us_windows_10_business_editions_version_22h2_x64_dvd_8cf17b79.iso 后，解压到：D:\en-us_windows_10_business_editions_version_22h2_x64_dvd_8cf17b79

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

1.4.　解压完成后，将目录 en-us_windows_10_business_editions_version_22h2_x64_dvd_8cf17b79 更改为 D:\OS_10

1.5.　所有脚本、所有路径，已默认设置为 D:\OS_10 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_10\Sources 里，创建 ei.cfg：

@"

[Channel]

volume

[VL]

1

"@ | Out-File -FilePath "D:\OS_10\sources\EI.CFG" -Encoding Ascii

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 10 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。

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

2.2. 语言包：下载

https://software-download.microsoft.com/download/pr/19041.1.191206-1406.vb_release_CLIENTLANGPACKDVD_OEM_MULTI.iso

2.3. 功能包：下载

ARM64: https://software-static.download.prss.microsoft.com/pr/download/19041.1.191206-1406.vb_release_arm64fre_FOD-PACKAGES_OEM_PT1_arm64fre_MULTI.iso

X64: https://software-download.microsoft.com/download/pr/19041.1.191206-1406.vb_release_amd64fre_FOD-PACKAGES_OEM_PT1_amd64fre_MULTI.iso

X86: https://software-static.download.prss.microsoft.com/pr/download/19041.1.191206-1406.vb_release_x86fre_FOD-PACKAGES_OEM_PT1_x86fre_MULTI.iso

3. InBox Apps

https://software-static.download.prss.microsoft.com/dbazure/888969d5-f34g-4e03-ac9d-1f9786c66749/19041.3031.230508-1728.vb_release_svc_prod3_amd64fre_InboxApps.iso

IV   Windows 安全中心

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

- 开启 Windows 安全中心会扫描文件、会占用大量的 CPU。

- 测试中：未关闭前耗时 1 小时 22 分，关闭后耗时 20 分钟。

如何关闭：

绿色为命令行，按住 Windows 键并按 R 键启动"运行"。

1. 打开"Windows 安全中心"或运行：windowsdefender:

2. 选择"病毒和威胁防护"或运行：windowsdefender://threat

3. 找到"病毒和威胁防护设置"，点击"管理设置"或运行：windowsdefender://threatsettings，建议您关闭部分功能：

    3.1.   实时保护

    3.2.   云提供的保护

    3.3.   自动提交样本

    3.4.   篡改防护

4.   未处于封装时，建议您开启 Windows 安全中心。

V   命令行

1.   可选"Terminal"或"PowerShell ISE"，未安装"Terminal"，请前往 https://github.com/microsoft/terminal/releases 后下载；

2.   以管理员身份打开"Terminal"或"PowerShell ISE"，设置 PowerShell 执行策略：绕过，PS 命令行：

Set-ExecutionPolicy -ExecutionPolicy Bypass -Force

3.   在本文中，绿色部分属于 PS 命令行，请复制后，粘贴到"Terminal"对话框，按回车键（Enter）后开始运行；

4.   有 .ps1 时，点击文件右键，选择以 PowerShell 运行，或复制路径，粘贴到"Terminal "或"PowerShell ISE"里运行，带冒号的路径，在命令行添加 & 字符，示例：& "D:\YiSolutions_Encapsulation_SIP.ps1"

B.   语言包：提取

II   语言包：准备

1.   语言包：挂载

挂载 19041.1.191206-1406.vb_release_CLIENTLANGPACKDVD_OEM_MULTI.iso 或解压到任意位置；

2.   功能包：挂载

请正确选择架构版本，提取错误的语言包，安装会报错误。

ARM64: 19041.1.191206-1406.vb_release_arm64fre_FOD-PACKAGES_OEM_PT1_arm64fre_MULTI.iso

X64: 19041.1.191206-1406.vb_release_amd64fre_FOD-PACKAGES_OEM_PT1_amd64fre_MULTI.iso

X86: 19041.1.191206-1406.vb_release_x86fre_FOD-PACKAGES_OEM_PT1_x86fre_MULTI.iso

III   语言包：提取方案

1.   添加

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

2.   删除

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

3.   架构：x64

IV   执行提取命令

- Auto = 自动搜索本地所有磁盘，默认；

- 自定义路径，例如指定为 E 盘：$ISO = "E:\"

- Extract.ps1

  o \Expand\Extract.ps1

  o https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/Expand/Extract.ps1

- 复制代码

```
$ISO = "Auto"

$SaveTo = "D:\OS_10_Custom"

$Extract_language_Pack = @(

  @{ Tag = "zh-CN"; Arch = "AMD64"; Act = "Add"; Scope = @("Install\Install"; "Install\WinRE"; "Boot\Boot") }

  @{ Tag = "en-US"; Arch = "AMD64"; Act = "Del"; Scope = @( "Install\Install"; "Install\WinRE"; "Boot\Boot" ) }

)

Function Extract_Language

{

  param($NewArch, $Act, $NewLang, $Expand)

  Function Match_Required_Fonts

  {

    param($Lang)

    $Fonts = @(

      @{ Match = @("as", "ar-SA", "ar", "ar-AE", "ar-BH", "ar-DJ", "ar-DZ", "ar-EG", "ar-ER", "ar-IL", "ar-IQ", "ar-JO", "ar-KM", "ar-KW", "ar-LB", "ar-LY", "ar-MA", "ar-MR", "ar-OM", "ar-PS", "ar-QA", "ar-SD", "ar-SO", "ar-SS", "ar-SY", "ar-TD", "ar-TN", "ar-YE", "arz-Arab", "ckb-Arab", "fa", "fa-AF", "fa-IR", "glk-Arab", "ha-Arab", "ks-Arab", "ks-Arab-IN", "ku-Arab", "ku-Arab-IQ", "mzn-Arab", "pa-Arab", "pa-Arab-PK", "pnb-Arab", "prs", "prs-AF", "prs-Arab", "ps", "ps-AF", "sd-Arab", "sd-Arab-PK", "tk-Arab", "ug", "ug-Arab", "ug-CN", "ur", "ur-IN", "ur-PK", "uz-Arab", "uz-Arab-AF"); Name = "Arab"; }

      @{ Match = @("bn-IN", "as-IN", "bn", "bn-BD", "bpy-Beng"); Name = "Beng"; }

      @{ Match = @("da-dk", "iu-Cans", "iu-Cans-CA"); Name = "Cans"; }

      @{ Match = @("chr-Cher-US", "chr-Cher"); Name = "Cher"; }

      @{ Match = @("hi-IN", "bh-Deva", "brx", "brx-Deva", "brx-IN", "hi", "ks-Deva", "mai", "mr", "mr-IN", "ne", "ne-IN", "ne-NP", "new-Deva", "pi-Deva", "sa", "sa-Deva", "sa-IN"); Name = "Deva"; }

      @{ Match = @("am", "am-ET", "byn", "byn-ER", "byn-Ethi", "ti", "ti-ER", "ti-ET", "tig", "tig-ER", "tig-Ethi", "ve-Ethi", "wal", "wal-ET", "wal-Ethi"); Name = "Ethi"; }

      @{ Match = @("gu", "gu-IN"); Name = "Gujr"; }

      @{ Match = @("pa", "pa-IN", "pa-Guru"); Name = "Guru"; }

      @{ Match = @("zh-CN", "cmn-Hans", "gan-Hans", "hak-Hans", "wuu-Hans", "yue-Hans", "zh-gan-Hans", "zh-hak-Hans", "zh-Hans", "zh-SG", "zh-wuu-Hans", "zh-yue-Hans"); Name = "Hans"; }

      @{ Match = @("zh-TW", "cmn-Hant", "hak-Hant", "lzh-Hant", "zh-hak-Hant", "zh-Hant", "zh-HK", "zh-lzh-Hant", "zh-MO", "zh-yue-Hant"); Name = "Hant"; }
```

```
    @{ Match = @("he", "he-IL", "yi"); Name = "Hebr"; }

    @{ Match = @("ja", "ja-JP"); Name = "Jpan"; }

    @{ Match = @("km", "km-KH"); Name = "Khmr"; }

    @{ Match = @("kn", "kn-IN"); Name = "Knda"; }

    @{ Match = @("ko", "ko-KR"); Name = "Kore"; }

    @{ Match = @("de-de", "lo", "lo-LA"); Name = "Laoo"; }

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

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

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

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

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

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

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

  )

  ForEach ($item in $Fonts) {

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

      return $item.Name

    }

  }

  return "Not_matched"

}

Function Match_Other_Region_Specific_Requirements

{

  param($Lang)

  $RegionSpecific = @(

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

  )

  ForEach ($item in $RegionSpecific) {

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

      return $item.Name

    }

  }

  return "Skip_specific_packages"

}
```

```powershell
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 {
```

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

            }

        }

    }

    $AdvLanguage = @(

        @{

            Path = "Install\Install"

            Rule = @(

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

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

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

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

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

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

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

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

                "Microsoft-Windows-MSPaint-FoD-Package~31bf3856ad364e35~wow64~{Lang}~.cab"

                "Microsoft-Windows-MSPaint-FoD-Package~31bf3856ad364e35~{ARCH}~{Lang}~.cab"

                "Microsoft-Windows-Notepad-FoD-Package~31bf3856ad364e35~{ARCH}~{Lang}~.cab"

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

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

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

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

                "Microsoft-Windows-Printing-WFS-FoD-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"

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

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

                "Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"

                "Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~{Lang}~.cab"

                "Microsoft-Windows-InternationalFeatures-{Specific}-Package~31bf3856ad364e35~amd64~~.cab"

            )

        }

        @{

            Path = "Install\WinRE"

            Rule = @(
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

        )

    }

    @{

        Path = "Boot\Boot"

        Rule = @(

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

        )

    }

)

$NewFonts = Match_Required_Fonts -Lang $NewLang

$SpecificPackage = Match_Other_Region_Specific_Requirements -Lang $NewLang

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

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).Replace("{ARCH}", $NewArch).Replace("{ARCHC}", $NewArchC)

            }

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

        }

    }

}

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

C.   自定义封装

II   自定义封装：Install.wim

1.   查看 Install.wim 详细信息

      映像名称、映像描述、映像大小、架构、版本、索引号等；

```
      $ViewFile = "D:\OS_10\Sources\Install.wim"

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

2. 指定挂载 Install 路径

New-Item -Path "D:\OS_10_Custom\Install\Install\Mount" -ItemType directory -ea SilentlyContinue

3. 开始挂载 Install.wim

默认索引号：1

Mount-WindowsImage -ImagePath "D:\OS_10\sources\install.wim" -Index "1" -Path "D:\OS_10_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_10_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_10_Custom\Install\WinRE\Mount" -ItemType directory -ea SilentlyContinue

3.1.3. 开始挂载 WinRE.wim

默认索引号：1

Mount-WindowsImage -ImagePath "D:\OS_10_Custom\Install\Install\Mount\Windows\System32\Recovery\WinRE.wim" -Index "1" -Path "D:\OS_10_Custom\Install\WinRE\Mount"

3.1.4. 语言包

- 自动安装语言包：获取"组件：映像中已安装的所有包"后进行匹配，匹配到对应的名称后，再安装本地对应的语言包文件，查看报告"语言安装包适用于 WinRE.wim"。

- 添加语言时，须对应不同的架构版本，未对应时，添加过程中报错等提示。

3.1.4.1. 语言包：添加

- WinRE.Instl.lang.ps1

  - \Expand\Install\WinRE\WinRE.Instl.lang.ps1

  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/Expand/Install/WinRE/WinRE.Instl.lang.ps1

- 复制代码

```
$Mount = "D:\OS_10_Custom\Install\WinRE\Mount"

$Sources = "D:\OS_10_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 = "*SecureStartup*"; File = "winpe-securestartup_zh-CN.cab"; }

  @{ Match = "*ATBroker*"; File = "winpe-atbroker_zh-CN.cab"; }

  @{ Match = "*AudioCore*"; File = "winpe-audiocore_zh-CN.cab"; }

  @{ Match = "*AudioDrivers*"; File = "winpe-audiodrivers_zh-CN.cab"; }

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

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

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

  @{ Match = "*Speech-TTS*"; File = "winpe-speech-tts_zh-CN.cab"; }

  @{ Match = "*srh*"; File = "winpe-srh_zh-CN.cab"; }

  @{ Match = "*srt*"; File = "winpe-srt_zh-CN.cab"; }

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

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

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

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

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

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

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

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

    )

    ForEach ($Rule in $Language) {

      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_10_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_10_Custom\Install\WinRE\Mount" /Get-Intl

3.1.4.3. 语言包：删除

- 添加语言后，如果要部署到非英语区域，可通过删除英语语言组件来节省空间。 删除语言时，
  请按与添加语言组件相反的顺序卸载语言组件。

- 添加中文后，反向删除"英语 - 美国"，区域：en-US，需提前提取语言包

- WinRE.Del.Specified.lang.Tag.ps1

  - \Expand\Install\WinRE\WinRE.Del.Specified.lang.Tag.ps1

  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/Expand/Install/WinRE/WinRE.Del.Specified.lang.Tag.ps1

- 复制代码

```
$Lang = "en-US"

$Mount = "D:\OS_10_Custom\Install\WinRE\Mount"

$Sources = "D:\OS_10_Custom\Install\WinRE\Language\Del\en-US"

$Initl_install_Language_Component = @()

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

    $Initl_install_Language_Component += $_.PackageName

}

$Language = @(

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

    @{ Match = "*ATBroker*"; File = "winpe-atbroker_$($Lang).cab"; }
```

```powershell
      @{ Match = "*SecureStartup*"; File = "winpe-securestartup_$($Lang).cab"; }

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

)

ForEach ($Rule in $Language) {

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

    ForEach ($Component in $Initl_install_Language_Component) {

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

            Write-host "   Component name: " -NoNewline

            Write-host $Component -ForegroundColor Green

            Write-host "   Language pack file: " -NoNewline

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

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

            try {

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

                Write-host "Finish" -ForegroundColor Green

            } catch {

                Write-host "Failed" -ForegroundColor Red

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

            }

            break

        }

    }

}

$InitlClearLanguagePackage = @()

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

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

        $InitlClearLanguagePackage += $_.PackageName

    }

}

if ($InitlClearLanguagePackage.count -gt 0) {

    ForEach ($item in $InitlClearLanguagePackage) {

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

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

        try {
```

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

        Write-host "Finish" -ForegroundColor Green

    } catch {

        Write-host "Failed" -ForegroundColor Red

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

    }

  }

}
```

3.1.4.4. 组件：映像中已安装的所有包

3.1.4.4.1. 查看

```
Get-WindowsPackage -Path "D:\OS_10_Custom\Install\WinRE\Mount" | Out-GridView
```

3.1.4.4.2. 导出到 Csv

```
$SaveTo = "D:\OS_10_Custom\Install\WinRE\Report.$(Get-Date -Format
"yyyyMMddHHmmss").csv"

Get-WindowsPackage -Path "D:\OS_10_Custom\Install\WinRE\Mount" | Export-CSV -NoType
-Path $SaveTo

Write-host $SaveTo -ForegroundColor Green
```

3.1.5. 累积更新

准备可用的累积更新文件，请更改示例文件名：KB_WinRE.cab

3.1.5.1. 添加

```
$KBPath = "D:\OS_10_Custom\Install\WinRE\Update\KB_WinRE.cab"

Add-WindowsPackage -Path "D:\OS_10_Custom\Install\WinRE\Mount" -PackagePath $KBPath
```

3.1.5.2. 删除

```
$KBPath = "D:\OS_10_Custom\Install\WinRE\Update\KB_WinRE.cab"

Remove-WindowsPackage -Path "D:\OS_10_Custom\Install\WinRE\Mount" -PackagePath $KBPath
```

3.1.5.3. 固化更新

固化后不可卸载，固化时将清理恢复映像并重置任何被取代的组件的基础。

```
Dism /image:"D:\OS_10_Custom\Install\WinRE\Mount" /cleanup-image /StartComponentCleanup
```

```
/ResetBase
```

3.1.5.3.1. 固化更新后清理组件

```powershell
$Mount = "D:\OS_10_Custom\Install\WinRE\Mount"

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

    if ($_.PackageState -eq "Superseded") {

        Write-Host "  $($_.PackageName)" -ForegroundColor Green

        Remove-WindowsPackage -Path $Mount -PackageName $_.PackageName | Out-Null

    }

}
```

3.1.6. 驱动

3.1.7. 保存映像：WinRE.wim

```powershell
Save-WindowsImage -Path "D:\OS_10_Custom\Install\WinRE\Mount"
```

3.1.8. 卸载映像：WinRE.wim

关闭所有可能正在访问映像中文件的应用程序，包括文件资源管理器。

```powershell
Dismount-WindowsImage -Path "D:\OS_10_Custom\Install\WinRE\Mount" -Discard
```

3.1.9. 重建 WinRE.wim 后，可缩小文件大小

- WinRE.Rebuild.ps1
  - o \Expand\Install\WinRE\WinRE.Rebuild.ps1
  - o https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/Expand/Install/WinRE/WinRE.Rebuild.ps1

- 复制代码

```powershell
$FileName = "D:\OS_10_Custom\Install\Install\Mount\Windows\System32\Recovery\WinRE.wim"

Get-WindowsImage -ImagePath $Filename -ErrorAction SilentlyContinue | ForEach-Object {

    Write-Host "  Image name: " -NoNewline

    Write-Host $_.ImageName -ForegroundColor Yellow

    Write-Host "  The index number: " -NoNewline

    Write-Host $_.ImageIndex -ForegroundColor Yellow

    Write-Host "`n  Rebuild".PadRight(28) -NoNewline

    Export-WindowsImage -SourceImagePath $Filename -SourceIndex $_.ImageIndex -DestinationImagePath
```

```
            "$($FileName).New" -CompressionType max

        Write-Host "Finish`n" -ForegroundColor Green

    }

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

        Remove-Item -Path $Filename

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

        Write-Host "Finish" -ForegroundColor Green

    } else {

        Write-host "Failed" -ForegroundColor Red

    }
```

3.1.10. 备份 WinRE.wim

- WinRE.Backup.ps1

  o \Expand\Install\WinRE\WinRE.Backup.ps1

  o https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/Expand/Install/WinRE/WinRE.Backup.ps1

- 复制代码

```
$WimLibPath = "D:\OS_10_Custom\Install\Install\Update\Winlib"

$FileName = "D:\OS_10_Custom\Install\Install\Mount\Windows\System32\Recovery\WinRE.wim"

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

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

3.1.11. 替换 Install.wim 映像内的 WinRE.wim

- 每次挂载 Install.wim 后"替换 WinRE.wim"；

- 学习"获取 Install.wim 所有索引号后并替换旧的 WinRE.wim"。

处理 Install.wim 映像内的文件，结束。

4. 语言包

- 自动安装语言包：获取"组件：映像中已安装的所有包"后进行匹配，匹配到对应的名称后，再安装本地对应的语言包文件，查看报告"语言安装包适用于 Install.wim"。

- 添加语言时，须对应不同的架构版本，未对应时，添加过程中报错等提示。

4.1. 语言包：添加

- Install.Instl.lang.ps1

  - \Expand\Install\Install.Instl.lang.ps1

  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/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 = @(

            @{ Match_Name = "*Fonts*"; IsMatch = "No"; Capability = ""; }

        )

    }

    @{

        Name = "Basic"

        Description = "Basic"

        Rule = @(

            @{ Match_Name = "*LanguageFeatures-Basic*"; IsMatch = "Yes"; Capability = "Language.Basic~~~$($Lang)~0.0.1.0"; }

            @{ Match_Name = "*Client*Language*Pack*"; IsMatch = "Yes"; Capability = "Language.Basic~~~$($Lang)~0.0.1.0"; }

        )

    }

    @{

        Name = "OCR"

        Description = "Optical character recognition"

        Rule = @(

            @{ Match_Name = "*LanguageFeatures-OCR*"; IsMatch = "Yes"; Capability = "Language.OCR~~~$($Lang)~0.0.1.0"; }

        )

    }

    @{

        Name = "Handwriting"

        Description = "Handwriting recognition"

        Rule = @(

            @{ Match_Name = "*LanguageFeatures-Handwriting*"; IsMatch = "Yes"; Capability =
"Language.Handwriting~~~$($Lang)~0.0.1.0"; }

        )

    }

    @{

        Name = "TextToSpeech"
```

```
        Description = "Text-to-speech"

      Rule = @(

        @{ Match_Name = "*LanguageFeatures-TextToSpeech*"; IsMatch = "Yes"; Capability =
"Language.TextToSpeech~~~$($Lang)~0.0.1.0"; }

        )

      }

    @{

      Name = "Speech"

      Description = "Speech recognition"

      Rule = @(

        @{ Match_Name = "*LanguageFeatures-Speech*"; IsMatch = "Yes"; Capability = "Language.Speech~~~$($Lang)~0.0.1.0"; }

        )

      }

    @{

      Name = "RegionSpecific"

      Description = "Other region-specific requirements"

      Rule = @(

        @{ Match_Name = "*InternationalFeatures*"; IsMatch = "No"; Capability = ""; }

        )

      }

    @{

      Name = "Retail"

      Description = "Retail demo experience"

      Rule = @(

        @{ Match_Name = "*RetailDemo*"; IsMatch = "Yes"; Capability = ""; }

        )

      }

    @{

      Name = "Features_On_Demand"

      Description = "Features on demand"

      Rule = @(

        @{ Match_Name = "*InternetExplorer*"; IsMatch = "Yes"; Capability = ""; }

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

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

        @{ Match_Name = "*Notepad*amd64*"; IsMatch = "Yes"; Capability = "Microsoft.Windows.Notepad~~~~0.0.1.0"; }
```

```
            @{ Match_Name = "*Notepad*wow64*"; IsMatch = "Yes"; Capability = "Microsoft.Windows.Notepad~~~~0.0.1.0"; }

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

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

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

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

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

            @{ Match_Name = "*Printing*WFS*amd64*"; IsMatch = "Yes"; Capability = "Print.Management.Console~~~~0.0.1.0"; }

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

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

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

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

            @{ Match_Name = "*WMIC*FoD*Package*amd64*"; IsMatch = "Yes"; Capability = "WMIC~~~~"; }

            @{ Match_Name = "*WMIC*FoD*Package*wow64*"; IsMatch = "Yes"; Capability = "WMIC~~~~"; }

        )

    }

)

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

    ForEach ($WildCard in $Script:Init_Folder_All_File) {

      ForEach ($item in $Script:Search_File_Order) {

        ForEach ($NewRule in $item.Rule) {

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

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

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

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

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

            $new = $OSDefaultUser + $TempSave

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

              ForEach ($Component in $Initl_install_Language_Component) {

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

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

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

                  $Script:Init_Folder_All_File_Match_Done += $WildCard

                  break
```

```powershell
                }

              }

            } else {

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

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

                $Script:Init_Folder_All_File_Match_Done += $WildCard

            }

          }

        }

      }

    }

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

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

    ForEach ($WildCard in $Script:Search_File_Order) {

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

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

      if ($OSDefaultUser.Count -gt 0) {

        ForEach ($item in $OSDefaultUser) {

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

        }

      } else {

        Write-host "  Not available" -ForegroundColor Red

      }

    }

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

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

    ForEach ($item in $Script:Init_Folder_All_File) {

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

        $Script:Init_Folder_All_File_Exclude += $item

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

      }

    }

    Write-host "`n  Install" -ForegroundColor Yellow

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

```powershell
    ForEach ($WildCard in $Script:Search_File_Order) {

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

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

        if ($OSDefaultUser.Count -gt 0) {

            ForEach ($item in $OSDefaultUser) {

                Write-host "  Language pack file: " -NoNewline; Write-host $item.FileName -ForegroundColor Green

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

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

                    try {

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

                        Write-host "Finish`n" -ForegroundColor Green

                    } catch {

                        Write-host "Failed" -ForegroundColor Red

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

                    }

                } else {

                    Write-host "Does not exist`n"

                }

            }

        } else {

            Write-host "  Not available`n" -ForegroundColor Red

        }

    }

}

Language_Install -Mount "D:\OS_10_Custom\Install\Install\Mount" -Sources "D:\OS_10_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_10_Custom\Install\Install\Mount" /Set-AllIntl:"zh-CN"

4.2.2.　查看可用的语言设置

Dism /Image:"D:\OS_10_Custom\Install\Install\Mount" /Get-Intl

4.3.　语言包：删除

- 添加语言后，如果要部署到非英语区域，可通过删除英语语言组件来节省空间。删除语言时，请按与添加语言组件相反的顺序卸载语言组件。

- 添加中文后，反向删除"英语 - 美国"，区域：en-US，需提前提取语言包

- Install.Del.Specified.lang.Tag.ps1

  - \Expand\Install\Install.Del.Specified.lang.Tag.ps1

  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/Expand/Install/Install.Del.Specified.lang.Tag.ps1

- 复制代码

```
$Lang = "en-US"

$Mount = "D:\OS_10_Custom\Install\Install\Mount"

$Sources = "D:\OS_10_Custom\Install\Install\Language\Del\en-US"

$Initl_install_Language_Component = @()

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

    $Initl_install_Language_Component += $_.PackageName

}

$Language = @(

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

    @{ Match = "*Client-LanguagePack-Package*"; File = "Microsoft-Windows-Client-Language-Pack_x64_$($Lang).cab"; }

)

ForEach ($Rule in $Language) {

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

   ForEach ($Component in $Initl_install_Language_Component) {

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

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

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

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

       try {

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

         Write-host "Finish" -ForegroundColor Green

       } catch {

         Write-host "Failed" -ForegroundColor Red

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

       }

       break
```

```
          }

        }

      }

    $InitlClearLanguagePackage = @()

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

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

          $InitlClearLanguagePackage += $_.PackageName

       }

    }

    if ($InitlClearLanguagePackage.count -gt 0) {

      ForEach ($item in $InitlClearLanguagePackage) {

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

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

        try {

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

          Write-host "Finish" -ForegroundColor Green

        } catch {

          Write-host "Failed" -ForegroundColor Red

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

        }

      }

    }
```

4.4. 组件：映像中已安装的所有包

4.4.1. 查看

```
Get-WindowsPackage -Path "D:\OS_10_Custom\Install\Install\Mount" | Out-GridView
```

4.4.2. 导出到 Csv

```
$SaveTo = "D:\OS_10_Custom\Install\Install\Report.$(Get-Date -Format "yyyyMMddHHmmss").csv"

Get-WindowsPackage -Path "D:\OS_10_Custom\Install\Install\Mount" | Export-CSV -NoType -Path $SaveTo

Write-host $SaveTo -ForegroundColor Green
```

5. InBox Apps

5.1. 学习

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

5.2. InBox Apps：预安装应用程序

- 编组缩写

  - 群组 A: Windows 10 Education, Windows 10 Enterprise, Windows 10 Pro, Windows 10 Pro Education, Windows 10 Pro for Workstations, Windows 10 Home, Windows 10 Home Single Language, Windows 10 IoT Enterprise

  - 群组 B: Windows 10 Education N, Windows 10 Enterprise N, Windows 10 Pro N, Windows 10 Pro Education N, Windows 10 Pro N for Workstations, Windows 10 Home N

	名称	产品 ID	编组
1.	Microsoft.HEIFImageExtension	9PMMSR1CGPWG	A
2.	Microsoft.MixedReality.Portal	9NG1H8B3ZC7M	A
3.	Microsoft.SkypeApp	9WZDNCRFJ364	A
4.	Microsoft.VP9VideoExtensions	9N4D0MSMP0PT	A
5.	Microsoft.WebMediaExtensions	9N5TDP8VCMHS	A
6.	Microsoft.WebpImageExtension	9PG2DK419DRG	A
7.	Microsoft.WindowsSoundRecorder	9WZDNCRFHWKN	A
8.	Microsoft.Xbox.TCUI	9NKNC0LD5NN6	A
9.	Microsoft.XboxGamingOverlay	9NZKPSTSNW4P	A
10.	Microsoft.ZuneMusic	9WZDNCRFJ3PT	A
11.	Microsoft.ZuneVideo	9WZDNCRFJ3P2	A
12.	Microsoft.UI.Xaml.2.0		A, B
13.	Microsoft.UI.Xaml.2.1		A, B
14.	Microsoft.UI.Xaml.2.3		A, B
15.	Microsoft.Advertising.Xaml		A, B
16.	Microsoft.NET.Native.Framework.1.7		A, B
17.	Microsoft.NET.Native.Framework.2.2		A, B
18.	Microsoft.NET.Native.Runtime.1.7		A, B
19.	Microsoft.NET.Native.Runtime.2.2		A, B
20.	Microsoft.VCLibs.140.00		A, B
21.	Microsoft.VCLibs.140.00.UWPDesktop		A, B
22.	Services.Store.Engagement		A, B
23.	Microsoft.Cortana	9NFFX4SZZ23L	A, B
24.	Microsoft.BingWeather	9WZDNCRFJ3Q2	A, B

25.	Microsoft.DesktopAppInstaller	9NBLGGH4NNS1	A, B
26.	Microsoft.GetHelp	9PKDZBMV1H3T	A, B
27.	Microsoft.Getstarted	9WZDNCRDTBJJ	A, B
28.	Microsoft.Microsoft3DViewer	9NBLGGH42THS	A, B
29.	Microsoft.MicrosoftOfficeHub	9WZDNCRD29V9	A, B
30.	Microsoft.MicrosoftSolitaireCollection	9WZDNCRFHWD2	A, B
31.	Microsoft.MicrosoftStickyNotes	9NBLGGH4QGHW	A, B
32.	Microsoft.MSPaint	9NBLGGH4QGHW	A, B
33.	Microsoft.Office.OneNote	XPFFZHVGQWWLHB	A, B
34.	Microsoft.People	9NBLGGH10PG8	A, B
35.	Microsoft.ScreenSketch	9MZ95KL8MR0L	A, B
36.	Microsoft.Wallet	9NBLGGH52CKV	A, B
37.	Microsoft.StorePurchaseApp	9NBLGGH4LS1F	A, B
38.	Microsoft.Windows.Photos	9WZDNCRFJBH4	A, B
39.	Microsoft.WindowsAlarms	9WZDNCRFJ3PR	A, B
40.	Microsoft.WindowsCalculator	9WZDNCRFHVN5	A, B
41.	Microsoft.WindowsCamera	9WZDNCRFJBBG	A, B
42.	Microsoft.Windows.CommunicationsApps	9WZDNCRFHVQM	A, B
43.	Microsoft.WindowsFeedbackHub	9NBLGGH4R32N	A, B
44.	Microsoft.WindowsMaps	9WZDNCRDTBVB	A, B
45.	Microsoft.WindowsStore	9WZDNCRFJBMP	A, B
46.	Microsoft.XboxApp	9MV0B5HZVK9Z	A, B
47.	Microsoft.XboxGameOverlay	9NBLGGH537C2	A, B
48.	Microsoft.XboxIdentityProvider	9WZDNCRD1HKW	A, B
49.	Microsoft.XboxSpeechToTextOverlay	9P086NHDNB9W	A, B
50.	Microsoft.YourPhone	9NMPJ99VJBWV	A, B

5.3. InBox Apps：已安装

5.3.1. 查看

Get-AppXProvisionedPackage -Path "D:\OS_10_Custom\Install\Install\Mount" | Out-GridView

5.3.2. 导出到 Csv

$SaveTo = "D:\OS_10_Custom\Install\Install\Report.$(Get-Date -Format "yyyyMMddHHmmss").csv"

Get-AppXProvisionedPackage -Path "D:\OS_10_Custom\Install\Install\Mount" | Export-CSV -NoType -Path $SaveTo

```
                    Write-host $SaveTo -ForegroundColor Green
```

5.4.   删除已安装的所有预应用程序

- Install.InBox.Appx.Clear.all.ps1

    o \Expand\Install\Install.InBox.Appx.Clear.all.ps1

    o https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/Expand/Install/Install.InBox.Appx.Clear.all.ps1

- 复制代码

```
Get-AppXProvisionedPackage -path "D:\OS_10_Custom\Install\Install\Mount" -ErrorAction SilentlyContinue | ForEach-Object {

  Write-host "`n  $($_.DisplayName)"

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

  try {

    Remove-AppxProvisionedPackage -Path "D:\OS_10_Custom\Install\Install\Mount" -PackageName $_.PackageName -ErrorAction
SilentlyContinue | Out-Null

    Write-host "Finish" -ForegroundColor Green

  } catch {

    Write-host "Failed" -ForegroundColor Red

  }

}
```

5.5.   区域标记：了解

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

查看脱机映像已安装的语言

Dism /Image:"D:\OS_10_Custom\Install\Install\Mount" /Get-Intl

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

5.6.   区域标记：添加方式

5.6.1.   执行"语言包：添加"

5.6.2. 安装"本地语言体验包（LXPs）"

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

5.6.2.1. 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_10_Custom\Install\Install\InBox.Appx，文件格式：LanguageExperiencePack.zh-CN.Neutral.Appx

5.6.2.2. 手动下载

5.6.2.2.1. 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	9NTJLXMXX35J	az-latn-az	9P5TFKZHQ5K8
be-by	9MXPBGNNDW3L	bg-bg	9MX54588434F
bn-bd	9PH7TKVXGGM8	bn-in	9P1M44L7W84T
bs-latn-ba	9MVFKLJ10MFL	ca-es	9P6JMKJQZ9S7
ca-es-valencia	9P9K3WMFSW90	chr-cher-us	9MX15485N3RK
cs-cz	9P3WXZ1KTM7C	cy-gb	9NKJ9TBML4HB
da-dk	9NDMT2VKSNL1	de-de	9P6CT0SLW589
el-gr	9N586B13PBLD	en-gb	9NT52VQ39BVN
es-es	9NWVGWLHPB1Z	es-mx	9N8MCM1X3928
et-ee	9NFBHFMCR30L	eu-es	9NMCHQHZ37HZ
fa-ir	9NGS7DD4QS21	fi-fi	9MW3PQ7SD3QK
fil-ph	9NWM2KGTDSSS	fr-ca	9MTP2VP0VL92
fr-fr	9NHMG4BJKMDG	ga-ie	9P0L5Q848KXT
gd-gb	9P1DBPF36BF3	gl-es	9NXRNBRNJN9B
gu-in	9P2HMSWDJDQ1	ha-latn-ng	9N1L95DBGRG3
he-il	9NB6ZFND5HCQ	hi-in	9NZC3GRX8LD3
hr-hr	9NW01VND4LTW	hu-hu	9MWN3C58HL87
hy-am	9NKM28TM6P67	id-id	9P4X3N4SDK8P
ig-ng	9PG4ZFJ48JSX	is-is	9NTHJR7TQXX1
it-it	9P8PQWNS6VJX	ja-jp	9N1W692FV4S1

ka-ge	9P60JZL05WGH	kk-kz	9PHV179R97LV
km-kh	9PGKTS4JS531	kn-in	9NC6DB7N95F9
kok-in	9MV3P55CMZ6P	ko-kr	9N4TXPCVRNGF
ku-arab-iq	9P1C18QL3D7H	ky-kg	9P7D3JJGZM48
lb-lu	9N0ST1WBZ9D9	lo-la	9N8X352G5NZV
lt-lt	9NWWD891H6HN	lv-lv	9N5CQDPH6SQT
mi-nz	9P2GDFB3JPSX	mk-mk	9P1X6XB1K3RN
ml-in	9NWDTV8FFV7L	mn-mn	9PG1DHC4VTZW
mr-in	9MWXCKHJVR1J	ms-my	9NPXL8ZSDDQ7
mt-mt	9PDG96SQ6BN8	nb-no	9N6J0M5DHCK0
ne-np	9P7CHPLWDQVN	nl-nl	9PF1C9NB5PRV
nn-no	9PK7KM3Z06KH	nso-za	9NS49QLX5CDV
or-in	9NTHCXCXSJDH	pa-arab-pk	9NJRL03WH6FM
pa-in	9NSNC0ZJX69B	pl-pl	9NC5HW94R0LD
prs-af	9P3NGC6X5ZQC	pt-br	9P8LBDM4FW35
pt-pt	9P7X8QJ7FL0X	quc-latn-gt	9P2V6MNNQZ0B
quz-pe	9NHTX8NVQ04K	ro-ro	9MWXGPJ5PJ3H
ru-ru	9NMJCX77QKPX	rw-rw	9NFW0M20H9WG
sd-arab-pk	9NB9JSCXW9X5	si-lk	9NVF9QSLGTL0
sk-sk	9N7LSNN099WB	sl-si	9NV27L34J4ST
sq-al	9MWLRGNMDGK7	sr-cyrl-ba	9MXGN7V65C7B
sr-cyrl-rs	9PPD6CCK9K5H	sr-latn-rs	9NBZ0SJDPPVT
sv-se	9P0HSNX08177	sw-ke	9NFF2M19DQ55
ta-in	9PDZB1WT1B34	te-in	9PMQJJGF63FW
tg-cyrl-tj	9MZHLBPPT2HC	th-th	9MSTWFRL0LR4
ti-et	9NC8C9RDNK2S	tk-tm	9NKHQ4GL6VLT
tn-za	9NFSXM123DHT	tr-tr	9NL1D3T5HG9R
tt-ru	9NV90Q1X1ZR2	ug-cn	9P52C5D7VL5S
uk-ua	9PPPMZRSGHR8	ur-pk	9NDWFTFW12BQ
uz-latn-uz	9P5P2T5P5L9S	vi-vn	9P0W68X0XZPT
wo-sn	9NH3SW1CR90F	xh-za	9NW3QWSLQD17
yo-ng	9NGM3VPPZS5V	zh-cn	9NRMNT6GMZ7
zh-tw	9PCJ4DHCQ1JQ	zu-za	9NNRM7KT5NB0

下载区域：zh-CN，应用程序 ID：9NRMNT6GMZ70，商店连接：

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

5.6.2.2.2.　打开网站：https://store.rg-adguard.net

5.6.2.2.2.1.　搜索关键词：

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

5.6.2.2.2.2.　网页内搜索 22621 内容，搜索结果：

Microsoft.LanguageExperiencePackzh-CN_22621.*.
neutral__8wekyb3d8bbwe.appx

5.6.2.2.2.3.　下载后保存到 D:\OS_10_Custom\Install\Install\InBox.Appx 目录里，重命
名：LanguageExperiencePack.zh-cn.Neutral.Appx

5.6.2.3.　执行安装命令安装本地语言体验包（LXPs）

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

Add-AppxProvisionedPackage -Path "D:\OS_10_Custom\Install\Install\Mount" -PackagePath
"D:\OS_10_Custom\Install\Install\InBox.appx\LanguageExperiencePack.zh-cn.Neutral.appx" -SkipLicense

5.6.2.4.　InBox Apps：已安装的应用程序包

5.6.2.4.1.　查看

Get-AppXProvisionedPackage -Path "D:\OS_10_Custom\Install\Install\Mount" | Out-GridView

5.6.2.4.2.　导到出 Csv

$SaveTo = "D:\OS_10_Custom\Install\Install\Report.$(Get-Date -Format
"yyyyMMddHHmmss").csv"

Get-AppXProvisionedPackage -Path "D:\OS_10_Custom\Install\Install\Mount" | Export-CSV -
NoType -Path $SaveTo

Write-host $SaveTo -ForegroundColor Green

5.7.　InBox Apps：安装

5.7.1.　挂载或解压 InBox Apps 安装文件

挂载 19041.3031.230508-1728.vb_release_svc_prod3_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"

- Install.Inst.InBox.Appx.ps1

  - \Expand\Install\Install.Inst.InBox.Appx.ps1

  - https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/Expand/Install/Install.Inst.InBox.Appx.ps1

- 复制代码

```
$ISO = "Auto"

$Mount = "D:\OS_10_Custom\Install\Install\Mount"

try {

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

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

    Write-Host $Current_Edition_Version -ForegroundColor Green

} catch {

    Write-Host "Error" -ForegroundColor Red

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

    return

}

$Pre_Config_Rules = @(

    @{

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

        Apps = @(

            "Microsoft.UI.Xaml.2.3"; "Microsoft.UI.Xaml.2.4"; "Microsoft.UI.Xaml.2.7"; "Microsoft.NET.Native.Framework.2.2";
"Microsoft.NET.Native.Runtime.2.2"; "Microsoft.VCLibs.140.00"; "Microsoft.VCLibs.140.00.UWPDesktop";
"Microsoft.HEIFImageExtension"; "Microsoft.HEVCVideoExtension"; "Microsoft.SecHealthUI";
"Microsoft.VP9VideoExtensions"; "Microsoft.WebpImageExtension"; "Microsoft.WindowsStore"; "Microsoft.GamingApp";
"Microsoft.Sticky.Notes"; "Microsoft.Paint"; "Microsoft.PowerAutomateDesktop"; "Microsoft.ScreenSketch";
"Microsoft.WindowsNotepad"; "Microsoft.WindowsTerminal"; "Clipchamp.Clipchamp"; "Microsoft.Solitaire.Collection";
"Microsoft.WindowsAlarms"; "Microsoft.WindowsFeedbackHub"; "Microsoft.WindowsMaps"; "Microsoft.ZuneMusic";
"Microsoft.BingNews"; "Microsoft.DesktopAppInstaller"; "Microsoft.WindowsCamera"; "Microsoft.Getstarted";
"Microsoft.Cortana"; "Microsoft.BingWeather"; "Microsoft.GetHelp"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.People";
"Microsoft.StorePurchaseApp"; "Microsoft.Todos"; "Microsoft.WebMediaExtensions"; "Microsoft.Windows.Photos";
"Microsoft.WindowsCalculator"; "Microsoft.Windows.CommunicationsApps"; "Microsoft.WindowsSoundRecorder";
"Microsoft.Xbox.TCUI"; "Microsoft.XboxGameOverlay"; "Microsoft.XboxGamingOverlay"; "Microsoft.XboxIdentityProvider";
"Microsoft.XboxSpeechToTextOverlay";

             "Microsoft.YourPhone"; "Microsoft.ZuneVideo"; "MicrosoftCorporationII.QuickAssist";
"MicrosoftWindows.Client.WebExperience"; "Microsoft.RawImageExtension"; "MicrosoftCorporationII.MicrosoftFamily";

        )
```

```
    }
    @{

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

       Apps = @(

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

       )

    }
    @{

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

       Apps = @(

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

       )

    }

)

$Allow_Install_App = @()

ForEach ($item in $Pre_Config_Rules) {

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

      Write-host "`n   Match to: "-NoNewline

      Write-host $Current_Edition_Version -ForegroundColor Green
```

```powershell
      $Allow_Install_App = $item.Apps

      break

    }

}

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

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

ForEach ($item in $Allow_Install_App) {

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

}

$InBoxApps = @(

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

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

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

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

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

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

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

    @{ Name = "Microsoft.WindowsStore"; File = "Microsoft.WindowsStore_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsStore_8wekyb3d8bbwe.xml"; }

    @{ Name = "Microsoft.HEIFImageExtension"; File = "Microsoft.HEIFImageExtension_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.HEIFImageExtension_8wekyb3d8bbwe.xml"; }

    @{ Name = "Microsoft.HEVCVideoExtension"; File = "Microsoft.HEVCVideoExtension_8wekyb3d8bbwe.x64.appx"; License = "Microsoft.HEVCVideoExtension_8wekyb3d8bbwe.x64.xml"; }

    @{ Name = "Microsoft.SecHealthUI"; File = "Microsoft.SecHealthUI_8wekyb3d8bbwe.x64.appx"; License = "Microsoft.SecHealthUI_8wekyb3d8bbwe.x64.xml"; }

    @{ Name = "Microsoft.VP9VideoExtensions"; File = "Microsoft.VP9VideoExtensions_8wekyb3d8bbwe.x64.appx"; License = "Microsoft.VP9VideoExtensions_8wekyb3d8bbwe.x64.xml"; }

    @{ Name = "Microsoft.WebpImageExtension"; File = "Microsoft.WebpImageExtension_8wekyb3d8bbwe.x64.appx"; License = "Microsoft.WebpImageExtension_8wekyb3d8bbwe.x64.xml"; }

    @{ Name = "Microsoft.GamingApp"; File = "Microsoft.GamingApp_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.GamingApp_8wekyb3d8bbwe.xml"; }

    @{ Name = "Microsoft.Sticky.Notes"; File = "Microsoft.MicrosoftStickyNotes_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.MicrosoftStickyNotes_8wekyb3d8bbwe.xml"; }

    @{ Name = "Microsoft.Paint"; File = "Microsoft.Paint_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.Paint_8wekyb3d8bbwe.xml"; }

    @{ Name = "Microsoft.PowerAutomateDesktop"; File = "Microsoft.PowerAutomateDesktop_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.PowerAutomateDesktop_8wekyb3d8bbwe.xml"; }

    @{ Name = "Microsoft.ScreenSketch"; File = "Microsoft.ScreenSketch_8wekyb3d8bbwe.msixbundle"; License =
```

"Microsoft.ScreenSketch_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.WindowsNotepad"; File = "Microsoft.WindowsNotepad_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsNotepad_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.WindowsTerminal"; File = "Microsoft.WindowsTerminal_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsTerminal_8wekyb3d8bbwe.xml"; }

  @{ Name = "Clipchamp.Clipchamp"; File = "Clipchamp.Clipchamp_yxz26nhyzhsrt.msixbundle"; License = "Clipchamp.Clipchamp_yxz26nhyzhsrt.xml"; }

  @{ Name = "Microsoft.Solitaire.Collection"; File = "Microsoft.MicrosoftSolitaireCollection_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.MicrosoftSolitaireCollection_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.WindowsAlarms"; File = "Microsoft.WindowsAlarms_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsAlarms_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.WindowsFeedbackHub"; File = "Microsoft.WindowsFeedbackHub_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsFeedbackHub_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.WindowsMaps"; File = "Microsoft.WindowsMaps_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsMaps_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.ZuneMusic"; File = "Microsoft.ZuneMusic_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.ZuneMusic_8wekyb3d8bbwe.xml"; }

  @{ Name = "MicrosoftCorporationII.MicrosoftFamily"; File = "MicrosoftCorporationII.MicrosoftFamily_8wekyb3d8bbwe.msixbundle"; License = "MicrosoftCorporationII.MicrosoftFamily_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.BingNews"; File = "Microsoft.BingNews_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.BingNews_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.DesktopAppInstaller"; File = "Microsoft.DesktopAppInstaller_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.DesktopAppInstaller_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.WindowsCamera"; File = "Microsoft.WindowsCamera_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.WindowsCamera_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.Getstarted"; File = "Microsoft.Getstarted_8wekyb3d8bbwe.msixbundle"; License = "Microsoft.Getstarted_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.Cortana"; File = "Microsoft.CortanaApp_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.CortanaApp_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.BingWeather"; File = "Microsoft.BingWeather_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.BingWeather_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.GetHelp"; File = "Microsoft.GetHelp_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.GetHelp_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.MicrosoftOfficeHub"; File = "Microsoft.MicrosoftOfficeHub_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.MicrosoftOfficeHub_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.People"; File = "Microsoft.People_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.People_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.StorePurchaseApp"; File = "Microsoft.StorePurchaseApp_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.StorePurchaseApp_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.Todos"; File = "Microsoft.Todos_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.Todos_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.WebMediaExtensions"; File = "Microsoft.WebMediaExtensions_8wekyb3d8bbwe.appxbundle";

License = "Microsoft.WebMediaExtensions_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.Windows.Photos"; File = "Microsoft.Windows.Photos_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.Windows.Photos_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.WindowsCalculator"; File = "Microsoft.WindowsCalculator_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.WindowsCalculator_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.Windows.CommunicationsApps"; File = "Microsoft.WindowsCommunicationsApps_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.WindowsCommunicationsApps_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.WindowsSoundRecorder"; File = "Microsoft.WindowsSoundRecorder_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.WindowsSoundRecorder_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.Xbox.TCUI"; File = "Microsoft.Xbox.TCUI_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.Xbox.TCUI_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.XboxGameOverlay"; File = "Microsoft.XboxGameOverlay_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.XboxGameOverlay_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.XboxGamingOverlay"; File = "Microsoft.XboxGamingOverlay_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.XboxGamingOverlay_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.XboxIdentityProvider"; File = "Microsoft.XboxIdentityProvider_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.XboxIdentityProvider_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.XboxSpeechToTextOverlay"; File = "Microsoft.XboxSpeechToTextOverlay_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.XboxSpeechToTextOverlay_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.YourPhone"; File = "Microsoft.YourPhone_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.YourPhone_8wekyb3d8bbwe.xml"; }

  @{ Name = "Microsoft.ZuneVideo"; File = "Microsoft.ZuneVideo_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.ZuneVideo_8wekyb3d8bbwe.xml"; }

  @{ Name = "MicrosoftCorporationII.QuickAssist"; File = "MicrosoftCorporationII.QuickAssist_8wekyb3d8bbwe.appxbundle"; License = "MicrosoftCorporationII.QuickAssist_8wekyb3d8bbwe.xml"; }

  @{ Name = "MicrosoftWindows.Client.WebExperience"; File = "MicrosoftWindows.Client.WebExperience_cw5n1h2txyewy.appxbundle"; License = "MicrosoftWindows.Client.WebExperience_cw5n1h2txyewy.xml"; }

  @{ Name = "Microsoft.RawImageExtension"; File = "Microsoft.RawImageExtension_8wekyb3d8bbwe.appxbundle"; License = "Microsoft.RawImageExtension_8wekyb3d8bbwe.xml"; }

)

Function Install_Appx

{

  param($File, $License)

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

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

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

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

```powershell
        Write-host "  License: " -NoNewline; Write-host $License -ForegroundColor Yellow

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

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

        try {

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

          Write-Host "Done" -ForegroundColor Green

        } catch {

          Write-Host "Failed" -ForegroundColor Red

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

        }

      } else {

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

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

        try {

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

          Write-Host "Done" -ForegroundColor Green

        } catch {

          Write-Host "Failed" -ForegroundColor Red

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

        }

      }

    } else {

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

    }

}

ForEach ($Rule in $InBoxApps) {

  Write-host "`n  Name: " -NoNewline

  Write-host $Rule.Name -ForegroundColor Yellow

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

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

    Write-host "  Search for apps: " -NoNewline

    Write-host $Rule.File -ForegroundColor Yellow

    Write-host "  Search for License: " -NoNewline
```

```
                    Write-host $Rule.File -ForegroundColor Yellow

                    if ($ISO -eq "Auto") {

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

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

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

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

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

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

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

                            Write-host $LicensePath -ForegroundColor Green

                          } else {

                            Write-host "  License: " -NoNewLine

                            Write-host "Not found" -ForegroundColor Red

                          }

                          Install_Appx -File $AppPath -License $LicensePath

                          return

                        }

                      }

                    } else {

                      Install_Appx -File "$($ISO)\$($Rule.File)" -License "$($ISO)\$($Rule.License)"

                    }

                  } else {

                    Write-host "  Skip the installation" -ForegroundColor Red

                  }

                }
```

5.8.   区域标记：删除

- 完成安装 InBox Apps 应用后，区域标记已不再重要，可删除或不删除，删除"本地语言体验包（LXPs），简体中文 - 中
  国）"，区域标记： zh-CN，可更改为其它区域标记。

- Install.Clear.Flag.ps1

  o   \Expand\Install\Install.Clear.Flag.ps1

  o   https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/Expand/Install/Install.Clear.Flag.ps1

- 复制代码

```
$Lang = "zh-CN"

Get-AppXProvisionedPackage -Path "D:\OS_10_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_10_Custom\Install\Install\Mount" -PackageName $_.PackageName -ErrorAction SilentlyContinue | Out-Null

      Write-host "Finish" -ForegroundColor Green

    } catch {

      Write-host "Failed" -ForegroundColor Red

    }

  }

}
```

5.9.　　InBox Apps：优化

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

Dism /Image:"D:\OS_10_Custom\Install\Install\Mount" /Optimize-ProvisionedAppxPackages

6.　累积更新

- 不同版本、旧版本升级到最新版时，需优先添加"功能启用包"后才能添加最新版累积更新；

- 添加语言包后，可安装与初始版本相同的累积更新，以解决安装后未刷新"组件：映像中已安装的所有包"状态的已知问题；

- 为保持最新，推荐您下载最新版。

6.1.　　功能启用包

- o　　学习：KB5015684：使用启用包对 Windows 10 版本 22H2 的特别推荐更新

- o　　注意：使用 Windows 10 22H2 19045.2006 相同版本和以上版本号时，可跳过功能启用包。

- o　　制作 Windows 10 版本 2004、20H2、21H1、21H2 时，需提前安装"功能启用包"后才能安装 19045.* 的累积更新，下载后保存到：D:\OS_10_Custom\Install\Install\Update，根据架构选择下载：

6.1.1.　　x64，默认

- 直连下载

https://catalog.s.download.windowsupdate.com/c/upgr/2022/07/windows10.0-kb5015684-x64_523c039b86ca98f2d818c4e6706e2cc94b634c4a.msu

- 安装

$KBPath = "D:\OS_10_Custom\Install\Install\Update\windows10.0-kb5015684-x64_523c039b86ca98f2d818c4e6706e2cc94b634c4a.msu"

Add-WindowsPackage -Path "D:\OS_10_Custom\Install\Install\Mount" -PackagePath $KBPath

6.1.2. x86

- 直连下载

https://catalog.s.download.windowsupdate.com/c/upgr/2022/07/windows10.0-kb5015684-x86_556015f4260dd75ca30b3e84484ea5b3ec97fa08.msu

- 安装

$KBPath = "D:\OS_10_Custom\Install\Install\Update\windows10.0-kb5015684-x86_556015f4260dd75ca30b3e84484ea5b3ec97fa08.msu"

Add-WindowsPackage -Path "D:\OS_10_Custom\Install\Install\Mount" -PackagePath $KBPath

6.1.3. Arm64

- 直连下载

https://catalog.s.download.windowsupdate.com/c/upgr/2022/07/windows10.0-kb5015684-arm64_a4cc6e95084da14adb972e1d67cf573a244ed7a3.msu

- 安装

$KBPath = "D:\OS_10_Custom\Install\Install\Update\windows10.0-kb5015684-arm64_a4cc6e95084da14adb972e1d67cf573a244ed7a3.msu"

Add-WindowsPackage -Path "D:\OS_10_Custom\Install\Install\Mount" -PackagePath $KBPath

6.2. 初始版本

累积更新 KB5017308 已无法从 https://www.catalog.update.microsoft.com/Search.aspx?q=KB5017308 里搜索到，通过直连方式下载后保存到：D:\OS_10_Custom\Install\Install\Update，根据架构选择下载：

6.2.1. x64，默认

- 直连下载

https://catalog.s.download.windowsupdate.com/c/msdownload/update/software/secu/2022/09/windows10.0-kb5017308-x64_2027053968a06948b45d139d95475ab4feee5654.msu

- 安装

  $KBPath = "D:\OS_10_Custom\Install\Install\Update\windows10.0-kb5017308-x64_2027053968a06948b45d139d95475ab4feee5654.msu"

  Add-WindowsPackage -Path "D:\OS_10_Custom\Install\Install\Mount" -PackagePath $KBPath

6.2.2.　x86

- 直连下载

  https://catalog.s.download.windowsupdate.com/d/msdownload/update/software/secu/2022/09/windows10.0-kb5017308-x86_ec788348202c33b871c8af028d045debd273576c.msu

- 安装

  $KBPath = "D:\OS_10_Custom\Install\Install\Update\windows10.0-kb5017308-x86_ec788348202c33b871c8af028d045debd273576c.msu"

  Add-WindowsPackage -Path "D:\OS_10_Custom\Install\Install\Mount" -PackagePath $KBPath

6.2.3.　Arm64

- 直连下载

  https://catalog.s.download.windowsupdate.com/d/msdownload/update/software/secu/2022/09/windows10.0-kb5017308-arm64_1ca51a0b3abdab7ee891bd1e1e6d3f03bc15e324.msu

- 安装

  $KBPath = "D:\OS_10_Custom\Install\Install\Update\windows10.0-kb5017308-arm64_1ca51a0b3abdab7ee891bd1e1e6d3f03bc15e324.msu"

  Add-WindowsPackage -Path "D:\OS_10_Custom\Install\Install\Mount" -PackagePath $KBPath

6.3.　其它版本

查阅"Windows 10 版本信息"，例如下载累积更新：KB5032278，版本号：19045.4170，前往下载页面：https://www.catalog.update.microsoft.com/Search.aspx?q=KB5032278，下载后保存到：D:\OS_10_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.s.download.windowsupdate.com/d/msdownload/update/software/updt/2023/11/windows10.0-kb5032278-x64_3a2b384f690eaa92c5030c535ac4a7e47c71a635.msu

- 添加

```
$KBPath = "D:\OS_10_Custom\Install\Install\Update\windows10.0-kb5032278-
x64_3a2b384f690eaa92c5030c535ac4a7e47c71a635.msu"

Add-WindowsPackage -Path "D:\OS_10_Custom\Install\Install\Mount" -PackagePath $KBPath
```

6.3.2.　x86

- 直连下载

  https://catalog.s.download.windowsupdate.com/d/msdownload/update/software/updt/2023/11/windows10.0-kb5032278-x86_21e1e1c12d0f763d7c6e3ac6c3d7f8c6461e0004.msu

- 添加

```
$KBPath = "D:\OS_10_Custom\Install\Install\Update\windows10.0-kb5032278-
x86_21e1e1c12d0f763d7c6e3ac6c3d7f8c6461e0004.msu"

Add-WindowsPackage -Path "D:\OS_10_Custom\Install\Install\Mount" -PackagePath $KBPath
```

6.3.3.　Arm64

- 直连下载

  https://catalog.s.download.windowsupdate.com/c/msdownload/update/software/updt/2023/11/windows10.0-kb5032278-arm64_54422d085c9d2c64ca0ff6e88ace3ec9f8860b6a.msu

- 添加

```
$KBPath = "D:\OS_10_Custom\Install\Install\Update\windows10.0-kb5032278-
arm64_54422d085c9d2c64ca0ff6e88ace3ec9f8860b6a.msu"

Add-WindowsPackage -Path "D:\OS_10_Custom\Install\Install\Mount" -PackagePath $KBPath
```

6.4.　固化更新，可选项

固化后不可卸载，固化时将清理恢复映像并重置任何被取代的组件的基础。

```
Dism /Image:"D:\OS_10_Custom\Install\Install\Mount" /cleanup-image /StartComponentCleanup /ResetBase
```

6.4.1.　固化更新后清理组件

```
$Mount = "D:\OS_10_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_10_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_10_Custom\Install\Install\Update\Winlib\WinRE.wim"

    $CopyTo = "D:\OS_10_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_10_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_10_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

                       o \Expand\Install\Install.WinRE.Replace.wim.ps1

- 复制代码

```
$Arguments = @(

  "extract",

  "D:\OS_10\sources\install.wim", "1",

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

  "--dest-dir=""D:\OS_10_Custom\Install\Install\Update\Winlib"""

)

New-Item -Path "D:\OS_10_Custom\Install\Install\Update\Winlib" -ItemType Directory -ea SilentlyContinue

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

13.2.2. 获取 Install.wim 所有索引号后并替换旧的 WinRE.wim

- Install.WinRE.Replace.wim.ps1

- 复制代码

```
Get-WindowsImage -ImagePath "D:\OS_10\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_10\sources\install.wim", $_.ImageIndex,

    "--command=""add 'D:\OS_10_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.10/22H2/Expand/Install/Install.Rebuild.wim.ps1

- 复制代码

```
$InstallWim = "D:\OS_10\sources\install.wim"

Get-WindowsImage -ImagePath $InstallWim -ErrorAction SilentlyContinue | ForEach-Object {

    Write-Host "   Image name: " -NoNewline

    Write-Host $_.ImageName -ForegroundColor Yellow

    Write-Host "   The index number: " -NoNewline

    Write-Host $_.ImageIndex -ForegroundColor Yellow

    Write-Host "`n   Under reconstruction".PadRight(28) -NoNewline

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

    Write-Host "Finish`n" -ForegroundColor Green

}

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

    Remove-Item -Path $InstallWim

    Move-Item -Path "$($InstallWim).New" -Destination $InstallWim

    Write-Host "Finish" -ForegroundColor Green

} else {

    Write-host "Failed" -ForegroundColor Red

}
```

III   自定义封装：boot.wim

1.   查看 Boot.wim 文件信息

映像名称、映像描述、映像大小、架构、版本、索引号等；

```
$ViewFile = "D:\OS_10\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_10_Custom\Boot\Boot\Mount" -ItemType directory -ea SilentlyContinue
```

3.   开始挂载 Boot.wim

默认索引号：2

Mount-WindowsImage -ImagePath "D:\OS_10\sources\boot.wim" -Index "2" -Path "D:\OS_10_Custom\Boot\Boot\Mount"

4. 语言包

- 自动安装语言包：获取"组件：映像中已安装的所有包"后进行匹配，匹配到对应的名称后，再安装本地对应的语言包文件，查看
  报告"语言安装包适用于 Boot.wim"。

- 添加语言时，须对应不同的架构版本，未对应时，添加过程中报错等提示。

4.1. 语言包：添加

- Boot.Instl.lang.ps1

  o \Expand\Boot\Boot.Instl.lang.ps1

  o https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/Expand/Boot/Boot.Instl.lang.ps1

- 复制代码

$Mount = "D:\OS_10_Custom\Boot\Boot\Mount"

$Sources = "D:\OS_10_Custom\Boot\Boot\Language\Add\zh-CN"

$Initl_install_Language_Component = @()

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

    $Initl_install_Language_Component += $_.PackageName

}

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

$Language = @(

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

    @{ Match = "*WinPE*Setup*Package*"; File = "WinPE-Setup_zh-CN.cab"; }

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

    @{ Match = "*SecureStartup*"; File = "winpe-securestartup_zh-CN.cab"; }

    @{ Match = "*ATBroker*"; File = "winpe-atbroker_zh-CN.cab"; }

    @{ Match = "*AudioCore*"; File = "winpe-audiocore_zh-CN.cab"; }

    @{ Match = "*AudioDrivers*"; File = "winpe-audiodrivers_zh-CN.cab"; }

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

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

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

    @{ Match = "*Speech-TTS*"; File = "winpe-speech-tts_zh-CN.cab"; }

    @{ Match = "*srh*"; File = "winpe-srh_zh-CN.cab"; }

```
        @{ Match = "*srt*"; File = "winpe-srt_zh-CN.cab"; }

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

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

    )

    ForEach ($Rule in $Language) {

      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

        }

      }

    }
```

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_10_Custom\Boot\Boot\Mount" /Set-AllIntl:zh-CN

4.2.2. 查看可用的语言设置

Dism /Image:"D:\OS_10_Custom\Boot\Boot\Mount" /Get-Intl

4.3. 语言包：删除

- 添加语言后，如果要部署到非英语区域，可通过删除英语语言组件来节省空间。 删除语言时，请按与添加语言组件相反的顺序卸载语言组件。

- 添加中文后，反向删除"英语 - 美国"，区域：en-US，需提前提取语言包

- Boot.Del.Specified.lang.Tag.ps1

    o \Expand\Boot\Boot.Del.Specified.lang.Tag.ps1

    o https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/Expand/Boot/Boot.Del.Specified.lang.Tag.ps1

- 复制代码

```
$Lang = "en-US"

$Mount = "D:\OS_10_Custom\Boot\Boot\Mount"

$Sources = "D:\OS_10_Custom\Boot\Boot\Language\Del\en-US"

$Initl_install_Language_Component = @()

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

  $Initl_install_Language_Component += $_.PackageName

}

$Language = @(

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

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

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

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

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

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

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

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

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

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

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

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

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

  @{ Match = "*InternetExplorer*"; File = "Microsoft-Windows-InternetExplorer-Optional-
```

```powershell
Package~31bf3856ad364e35~AMD64~$($Lang)~.cab"; }

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

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

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

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

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

    @{ Match = "*Client-LanguagePack-Package*"; File = "Microsoft-Windows-Client-Language-Pack_x64_$($Lang).cab"; }

)

ForEach ($Rule in $Language) {

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

   ForEach ($Component in $Initl_install_Language_Component) {

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

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

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

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

       try {

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

         Write-host "Finish" -ForegroundColor Green

       } catch {

         Write-host "Failed" -ForegroundColor Red

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

       }

       break

     }

   }

}

$InitlClearLanguagePackage = @()

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

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

     $InitlClearLanguagePackage += $_.PackageName

   }

}
```

```
if ($InitlClearLanguagePackage.count -gt 0) {

  ForEach ($item in $InitlClearLanguagePackage) {

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

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

    try {

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

      Write-host "Finish" -ForegroundColor Green

    } catch {

      Write-host "Failed" -ForegroundColor Red

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

    }

  }

}
```

4.4. 组件：映像中已安装的所有包

4.4.1. 查看

```
Get-WindowsPackage -Path "D:\OS_10_Custom\Boot\Boot\Mount" | Out-GridView
```

4.4.2. 导出到 Csv

```
$SaveTo = "D:\OS_10_Custom\Boot\Boot\Report.$(Get-Date -Format "yyyyMMddHHmmss").csv"

Get-WindowsPackage -Path "D:\OS_10_Custom\Boot\Boot\Mount" | Export-CSV -NoType -Path $SaveTo

Write-host $SaveTo -ForegroundColor Green
```

4.5. 语言包：同步到 ISO 安装程序

```
Copy-Item -Path "D:\OS_10_Custom\Boot\Boot\Mount\sources\zh-CN" -Destination "D:\OS_10\sources\zh-CN" -Recurse -Force
```

4.6. 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。

4.6.1. 重新生成已挂载目录 lang.ini

重新生成的 Lang.ini 文件位置：D:\OS_10_Custom\Boot\Boot\Mount\Sources\lang.ini

```
Dism /image:"D:\OS_10_Custom\Boot\Boot\Mount" /gen-langini /distribution:"D:\OS_10_Custom\Boot\Boot\Mount"
```

4.6.2. 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_10\Sources\lang.ini

Dism /image:"D:\OS_10_Custom\Boot\Boot\Mount" /gen-langini /distribution:"D:\OS_10"

5. 累积更新

准备可用的累积更新文件，请更改示例文件名：KB_Boot.cab

5.1. 添加

$KBPath = "D:\OS_10_Custom\Boot\Boot\Update\KB_Boot.cab"

Add-WindowsPackage -Path "D:\OS_10_Custom\Boot\Boot\Mount" -PackagePath $KBPath

5.2. 删除

$KBPath = "D:\OS_10_Custom\Boot\Boot\Update\KB_Boot.cab"

Remove-WindowsPackage -Path "D:\OS_10_Custom\Boot\Boot\Mount" -PackagePath $KBPath

5.3. 固化更新

固化后不可卸载，固化时将清理恢复映像并重置任何被取代的组件的基础。

Dism /image:"D:\OS_10_Custom\Boot\Boot\Mount" /cleanup-image /StartComponentCleanup /ResetBase

5.3.1. 固化更新后清理组件

$Mount = "D:\OS_10_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. 保存映像：Boot.wim

Save-WindowsImage -Path "D:\OS_10_Custom\Boot\Boot\Mount"

8. 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_10_Custom\Boot\Boot\Mount" -Discard
```

IV   部署引擎

- 了解"自动添加 Windows 系统已安装的语言"，学习：https://github.com/ilikeyi/Multilingual，如何下载：

  ○   进入网站后，点击 "代码"，"下载压缩包"，下载完成后得到 main.zip 压缩包文件。

  ○   前往 https://github.com/ilikeyi/Multilingual/releases 下载页面，选择可用版本：1.1.1.1，选择下载源代码格式：zip，下载完成后得到 Multilingual-1.1.1.1.zip 压缩包文件；

- 将已下载的 main.zip 或 Multilingual-1.1.1.1.zip，解压到：D:\Multilingual-1.1.1.1，重命名：D:\Multilingual

- 学习"无人值守 Windows 安装参考"，通过无人值守来干预安装过程。

1.   添加方式

　　1.1.   添加到 ISO 安装介质

　　　　1.1.1.   无人值守

　　　　　　1.1.1.1.   添加到：[ISO]:\Autounattend.xml

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

　　　　　　复制 D:\Multilingual_Learn\Unattend\Mul.Unattend.xml 到 D:\OS_10\Autounattend.xml

```
Copy-Item "D:\Multilingual\_Learn\Unattend\Mul.Unattend.xml" -Destination "D:\OS_10\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_10\Sources\Unattend.xml

```
Copy-Item "D:\Multilingual\_Learn\Unattend\Mul.Unattend.xml" -Destination "D:\OS_10\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_10\sources\`$OEM$\`$$\Panther" -ItemType Directory
```

　　　　　　　　1.1.1.3.2.   复制

复制 D:\Multilingual_Learn\Unattend\Mul.Unattend.xml 到
D:\OS_10\Sources\\OEM\Panther\Unattend.xml

Copy-Item "D:\Multilingual_Learn\Unattend\Mul.Unattend.xml" -Destination
"D:\OS_10\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_10\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_10\Sources\\$OEM$\\$1\Yi\Engine

Copy-Item "D:\Multilingual\Engine" -Destination "D:\OS_10\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_10\sources\`$OEM$\`$1\Yi\Engine\Deploy\Allow" -ItemType Directory -ErrorAction
SilentlyContinue | Out-Null

    Out-File -FilePath "D:\OS_10\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_10_Custom\Install\Install\Mount

1.2.1. 无人值守

复制 D:\Multilingual_Learn\Unattend\Mul.Unattend.xml 到 D:\OS_10_Custom\Install\Install\Mount\Panther\Unattend.xml

```
Copy-Item "D:\Multilingual\_Learn\Unattend\Mul.Unattend.xml" -Destination "D:\OS_10_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_10_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_10_Custom\Install\Install\Mount\Yi\Engine

```
Copy-Item "D:\Multilingual\Engine" -Destination "D:\OS_10_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_10_Custom\Install\Install\Mount\Yi\Engine\Deploy\Allow" -ItemType Directory -
    ErrorAction SilentlyContinue | Out-Null

        Out-File -FilePath "D:\OS_10_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_10\Autounattend.xml

- D:\OS_10\Sources\Unattend.xml

- D:\OS_10\sources\$OEM$\$$\Panther\unattend.xml

- D:\OS_10_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

```xml
<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

```xml
<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。

```xml
<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 OOBE 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.　　生成 ISO

1.　　下载 OScdimg

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

1.1.　　x64

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

1.2. x86

1.3. arm64

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

- ISO.ps1

    o \Expand\ISO.ps1

    o https://github.com/ilikeyi/solutions/blob/main/_Learn/Packaging.tutorial/OS.10/22H2/Expand/ISO.ps1

- 复制代码

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

$ISO = "D:\OS_10"

$Volume = "OS_10"

$SaveTo = "D:\OS_10.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
```

章节 2 报告

- 编组缩写：

    - 群组 A：Windows 10 Education, Windows 10 Enterprise, Windows 10 Pro Education, Windows 10 Pro for Workstations, Windows 10 IoT Enterprise

    - 群组 B：Windows 10 Education N, Windows 10 Enterprise N, Windows 10 Pro, Windows 10 Pro N, Windows 10 Pro Education N, Windows 10 Pro N for Workstations

    - 群组 C：Windows 10 Home, Windows 10 Home Single Language

    - 群组 D：Windows 10 家庭版 N

II 不同版本

1.	Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~~10.0.19041.2006	A, C
2.	Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~~10.0.19041.1806	A, B
	Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.19041.1	
3.	Microsoft-Windows-Printing-WFS-FoD-Package~31bf3856ad364e35~amd64~~10.0.19041.2006	A, B, C, D
	Microsoft-Windows-Printing-WFS-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.19041.906	

4.	Microsoft-OneCore-ApplicationModel-Sync-Desktop-FOD-Package~31bf3856ad364e35~amd64~~10.0.19041.1949	A, B, C, D
5.	Microsoft-OneCore-DirectX-Database-FOD-Package~31bf3856ad364e35~amd64~~10.0.19041.1	A, B, C, D
6.	Microsoft-Windows-Client-LanguagePack-Package~31bf3856ad364e35~amd64~en-US~10.0.19041.2006	A, B, C, D
7.	Microsoft-Windows-FodMetadata-Package~31bf3856ad364e35~amd64~~10.0.19041.1	A, B, C, D
8.	Microsoft-Windows-Foundation-Package~31bf3856ad364e35~amd64~~10.0.19041.1	A, B, C, D
9.	Microsoft-Windows-Hello-Face-Migration-Package~31bf3856ad364e35~amd64~~10.0.19041.1889	A, B, C, D
10.	Microsoft-Windows-Hello-Face-Package~31bf3856ad364e35~amd64~~10.0.19041.1889	A, B, C, D
11.	Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~~11.0.19041.1566	A, B, C, D
12.	Microsoft-Windows-LanguageFeatures-Basic-en-us-Package~31bf3856ad364e35~amd64~~10.0.19041.1	A, B, C, D
13.	Microsoft-Windows-LanguageFeatures-Handwriting-en-us-Package~31bf3856ad364e35~amd64~~10.0.19041.1	A, B, C, D
14.	Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~10.0.19041.1	A, B, C, D
15.	Microsoft-Windows-LanguageFeatures-Speech-en-us-Package~31bf3856ad364e35~amd64~~10.0.19041.1	A, B, C, D
16.	Microsoft-Windows-LanguageFeatures-TextToSpeech-en-us-Package~31bf3856ad364e35~amd64~~10.0.19041.1	A, B, C, D
17.	Microsoft-Windows-MSPaint-FoD-Package~31bf3856ad364e35~amd64~~10.0.19041.746	A, B, C, D
	Microsoft-Windows-MSPaint-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.19041.1	A, B, C, D
18.	Microsoft-Windows-MSPaint-FoD-Package~31bf3856ad364e35~wow64~~10.0.19041.1	A, B, C, D
	Microsoft-Windows-MSPaint-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.19041.1	A, B, C, D
19.	Microsoft-Windows-Notepad-FoD-Package~31bf3856ad364e35~amd64~~10.0.19041.1865	A, B, C, D
	Microsoft-Windows-Notepad-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.19041.488	A, B, C, D
20.	Microsoft-Windows-Notepad-FoD-Package~31bf3856ad364e35~wow64~~10.0.19041.1	A, B, C, D
	Microsoft-Windows-Notepad-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.19041.1	A, B, C, D
21.	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~~10.0.19041.1	A, B, C, D
	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~en-US~10.0.19041.1	A, B, C, D
22.	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~~10.0.19041.1	A, B, C, D
	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~en-US~10.0.19041.1	A, B, C, D
23.	Microsoft-Windows-QuickAssist-Package~31bf3856ad364e35~amd64~~10.0.19041.1949	A, B, C, D
24.	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~~10.0.19041.746	A, B, C, D
	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~en-US~10.0.19041.1	A, B, C, D
25.	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~~10.0.19041.1	A, B, C, D
	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~en-US~10.0.19041.1	A, B, C, D
26.	Microsoft-Windows-TabletPCMath-Package~31bf3856ad364e35~amd64~~10.0.19041.1865	A, B, C, D
27.	Microsoft-Windows-UserExperience-Desktop-Package~31bf3856ad364e35~amd64~~10.0.19041.1741	A, B, C, D
28.	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~~10.0.19041.1682	A, B, C, D
	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.19041.1	A, B, C, D

29.	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~~10.0.19041.1	A, B, C, D
	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.19041.1	A, B, C, D
30.	OpenSSH-Client-Package~31bf3856ad364e35~amd64~~10.0.19041.964	A, B, C, D
31.	Package_for_DotNetRollup~31bf3856ad364e35~amd64~~10.0.4556.3	A, B, C, D
32.	Package_for_KB5015684~31bf3856ad364e35~amd64~~19041.1799.1.2	A, B, C, D
33.	Package_for_RollupFix~31bf3856ad364e35~amd64~~19041.2006.1.7	A, B, C, D
34.	Package_for_ServicingStack_1704~31bf3856ad364e35~amd64~~19041.1704.1.4	A, B, C, D
35.	Package_for_ServicingStack_1940~31bf3856ad364e35~amd64~~19041.1940.1.0	A, B, C, D

III   语言安装包适用于：Install.wim

1.	Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab	A, B
	Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1	
2.	Microsoft-Windows-LanguageFeatures-Fonts-Hans-Package~31bf3856ad364e35~AMD64~~.cab	A, B, C, D
	Microsoft-Windows-LanguageFeatures-Fonts-Hans-Package~31bf3856ad364e35~amd64~~10.0.19041.1	
3.	Microsoft-Windows-Client-Language-Pack_x64_zh-CN.cab	A, B, C, D
	Microsoft-Windows-Client-LanguagePack-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1	
4.	Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~AMD64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~zh-CN~11.0.19041.1	
5.	Microsoft-Windows-LanguageFeatures-Basic-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab	A, B, C, D
	Microsoft-Windows-LanguageFeatures-Basic-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.19041.1	
6.	Microsoft-Windows-LanguageFeatures-Handwriting-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab	A, B, C, D
	Microsoft-Windows-LanguageFeatures-Handwriting-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.19041.1	
7.	Microsoft-Windows-LanguageFeatures-OCR-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab	A, B, C, D
	Microsoft-Windows-LanguageFeatures-OCR-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.19041.1	
8.	Microsoft-Windows-LanguageFeatures-Speech-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab	A, B, C, D
	Microsoft-Windows-LanguageFeatures-Speech-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.19041.1	
9.	Microsoft-Windows-LanguageFeatures-TextToSpeech-zh-CN-Package~31bf3856ad364e35~AMD64~~.cab	A, B, C, D
	Microsoft-Windows-LanguageFeatures-TextToSpeech-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.19041.1	
10.	Microsoft-Windows-MSPaint-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-MSPaint-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1	
11.	Microsoft-Windows-MSPaint-FoD-Package~31bf3856ad364e35~wow64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-MSPaint-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.19041.1	
12.	Microsoft-Windows-Notepad-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-Notepad-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1	

13.	Microsoft-Windows-Notepad-FoD-Package~31bf3856ad364e35~wow64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-Notepad-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.19041.1	
14.	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1	
15.	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.19041.1	
16.	Microsoft-Windows-Printing-WFS-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-Printing-WFS-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1	
17.	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~AMD64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1	
18.	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~zh-CN~10.0.19041.1	
19.	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~AMD64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1	
20.	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~zh-CN~.cab	A, B, C, D
	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.19041.1	

IV　　语言安装包适用于：WinRE.wim

1. WinPE-FontSupport-zh-CN.cab

   Microsoft-Windows-WinPE-FontSupport-ZH-CN-Package~31bf3856ad364e35~amd64~~10.0.19041.1

2. lp.cab

   Microsoft-Windows-WinPE-LanguagePack-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

3. winpe-appxpackaging_zh-CN.cab

   Microsoft-Windows-WinPE-AppxPackaging-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

4. winpe-atbroker_zh-CN.cab

   WinPE-ATBroker-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

5. winpe-audiocore_zh-CN.cab

   WinPE-AudioCore-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

6. winpe-audiodrivers_zh-CN.cab

   WinPE-AudioDrivers-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

7. winpe-enhancedstorage_zh-CN.cab

   WinPE-EnhancedStorage-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

8. winpe-hta_zh-CN.cab

   WinPE-HTA-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

9. winpe-narrator_zh-CN.cab

WinPE-Narrator-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

10. winpe-opcservices_zh-CN.cab

WinPE-OpcServices-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

11. winpe-rejuv_zh-CN.cab

WinPE-Rejuv-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

12. winpe-scripting_zh-CN.cab

WinPE-Scripting-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

13. winpe-securestartup_zh-CN.cab

WinPE-SecureStartup-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

14. winpe-speech-tts_zh-CN.cab

WinPE-Speech-TTS-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

15. winpe-srh_zh-CN.cab

WinPE-SRH-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

16. winpe-srt_zh-CN.cab

WinPE-SRT-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

17. winpe-storagewmi_zh-CN.cab

WinPE-StorageWMI-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

18. winpe-wds-tools_zh-CN.cab

WinPE-WDS-Tools-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

19. winpe-wifi_zh-CN.cab

WinPE-WiFi-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

20. winpe-wmi_zh-CN.cab

WinPE-WMI-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

V 语言安装包适用于：Boot.wim

1. WinPE-FontSupport-zh-CN.cab

Microsoft-Windows-WinPE-FontSupport-ZH-CN-Package~31bf3856ad364e35~amd64~~10.0.19041.1

2. lp.cab

Microsoft-Windows-WinPE-LanguagePack-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

3. winpe-atbroker_zh-CN.cab

WinPE-ATBroker-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

4. winpe-audiocore_zh-CN.cab

WinPE-AudioCore-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

5.　winpe-audiodrivers_zh-CN.cab

WinPE-AudioDrivers-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

6.　winpe-enhancedstorage_zh-CN.cab

WinPE-EnhancedStorage-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

7.　winpe-narrator_zh-CN.cab

WinPE-Narrator-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

8.　winpe-scripting_zh-CN.cab

WinPE-Scripting-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

9.　winpe-securestartup_zh-CN.cab

WinPE-SecureStartup-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

10.　WINPE-SETUP-CLIENT_zh-CN.CAB

WinPE-Setup-Client-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

11.　WinPE-Setup_zh-CN.cab

WinPE-Setup-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

12.　winpe-speech-tts_zh-CN.cab

WinPE-Speech-TTS-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

13.　winpe-srh_zh-CN.cab

WinPE-SRH-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

14.　winpe-srt_zh-CN.cab

WinPE-SRT-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

15.　winpe-wds-tools_zh-CN.cab

WinPE-WDS-Tools-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

16.　winpe-wmi_zh-CN.cab

WinPE-WMI-Package~31bf3856ad364e35~amd64~zh-CN~10.0.19041.1

章节 3　　常见问题

II　　清理所有挂载到

关闭所有可能正在访问映像中文件的应用程序，包括文件资源管理器。

Dismount-WindowsImage -Path "D:\OS_10_Custom\Install\Install\Mount" -Discard

Dismount-WindowsImage -Path "D:\OS_10_Custom\Install\WinRE\Mount" -Discard

Dismount-WindowsImage -Path "D:\OS_10_Custom\Boot\Boot\Mount" -Discard

III　　修复挂载出现异常的问题

1.　查看已挂载

```
Get-WindowsImage -Mounted
```

2. 删除保存在注册表里的 DISM 挂载记录

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

3. 删除与已损坏的已装载映像关联的所有资源。

```
Clear-WindowsCorruptMountPoint
```

```
Dism /cleanup-wim
```

IV   清理目录

```
Remove-Item "D:\OS_10_Custom" -Force -Recurse
```

```
Remove-Item "D:\OS_10\Sources\`$OEM$" -Force -Recurse
```

```
Remove-Item -Path "D:\OS_10\Autounattend.xml" -Force
```

```
Remove-Item -Path "D:\OS_10\Sources\Unattend.xml" -Force
```

章节 4   已知问题

暂无

Yi' Solutions

此副本封装教程隶属于 Yi's Solutions 内容，学习更多：

- Yi 的官方网站 | https://fengyi.tel/solutions

- Github | https://github.com/ilikeyi/solutions

作者： Yi

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

文档版本： 1.1

文档模型： 完整版

更新日期： 2024 - 6

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