

WINDOWS 11 22H2

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

在这背后藏着很多的隐藏故事，想解开这些，你准备好开始尝试封装了吗？

目录

- 一、部署映像
 - A. 先决条件
 - B. 语言包：提取
 - C. 自定义部署映像
 - D. ISO
- 二、常见问题
- 三、已知问题
- 四、组件：映像中已安装的所有包，对比

一、部署映像

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 获取相关包

学习[“附件：OS”](#)，查看[“Windows 11”](#)项，

1. 系统安装包

- 1.1. 准备：[en-us_windows_11_business_editions_version_22h2_x64_dvd_17a08ce3.iso](#)
- 1.2. 解压到：[D:\en-us_windows_11_business_editions_version_22h2_x64_dvd_17a08c](#)
- 1.3. 解压完成后，将目录[en-us_windows_11_business_editions_version_22h2_x64_dvd_17a08c](#)更改为 [D:\OS11](#)
- 1.4. 所有脚本、所有路径，已默认设置为 [D:\OS11](#) 为映像来源。

2. 语言包

2.1. 学习

2.1.1. 将语言添加到 Windows 11 映像

2.1.2. 语言和区域按需功能 (FOD)

2.1.2.1. 区域关联

区域关联是什么？

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

已知区域关联：

2.1.2.1.1. 区域：zh-TW，可选关联区域：zh-HK

2.1.2.2. 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

2.1.2.3. 其它区域特定的要求

2.1.2.3.1. 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。

触发已知区域时，需添加特定的“程序包”。

存在争议：

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

2.2. 语言包：下载

[22621.1.220506-1250.ni_release_amd64fre_CLIENT_LOF_PACKAGES_OEM.iso](#)

2.3. 语言包：修复

任选一个网站并打开：

- <https://uupdump.net>
- <https://uup.ee>

2.3.1. 打开后，搜索关键词：[22621.382](#)，在搜索结果选择：[Windows 11, version 22H2 \(22621.382\) amd64](#)

2.3.2. 打开后，选择“全部文件”；

2.3.3. 在“全部文件”页面里依次搜索绿色部分并下载：

2.3.3.1. 适用于：Install.wim（1项）

2.3.3.1.1. [Media](#)

2.3.3.2. 适用于：WinRE.wim，暂无

2.3.3.3. 适用于：Boot.wim，暂无

2.3.4. 下载所有文件后，拉动到页面下方，下载并运行“[生成重命名脚本（Windows）](#)”；

2.4. 使用 ISO 编辑软件，编辑 [22621.1.220506-1250.ni_release_amd64fre_CLIENT_LOF_PACKAGES_OEM.iso](#)，将下载的文件添加到 [\[ISO\]:\LanguagesAndOptionalFeatures](#) 目录里；

3. InBox Apps

3.1. 下载：[22621.1778.230511-2102.ni_release_svc_prod3_amd64fre_InboxApps.iso](#)

3.2. 下载：[22621.1.220506-1250.ni_release_amd64fre_InboxApps.iso](#) 后，提取以下文件后保存到桌面：

3.2.1. [Microsoft.HEVCVideoExtension_8wekyb3d8bbwe.x64.appx](#)

3.2.2. [Microsoft.HEVCVideoExtension_8wekyb3d8bbwe.x64.xml](#)

3.3. 使用 ISO 编辑工具，编辑 [22621.1778.230511-2102.ni_release_svc_prod3_amd64fre_InboxApps.iso](#)，将已提取文件添加到 [\[ISO\]:\packages](#) 目录里；

IV 运行 PS 命令行时：

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：

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。

B. 语言包：提取

II 准备语言包

挂载 22621.1.220506-1250.ni_release_amd64fre_CLIENT_LOF_PACKAGES_OEM.iso 或解压到任意位置；

III 提取语言包方案

1. 添加

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

2. 删除

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

IV 执行提取命令

- Auto = 自动搜索本地所有磁盘，默认；
- 自定义路径，例如指定为 E 盘：\$ISO = "E:\"

复制代码或查看源文件：Extract.ps1 (本地, Github)

```
$ISO = "Auto"
$SaveTo = "D:\OS11_Custom"
$Extract_language_Pack = @(
    @{ Tag = "zh-CN"; Act = "Add"; Scope = @( "Install\Install"; "Install\WinRE"; "Boot\Boot" ) }
    @{ Tag = "en-US"; Act = "Del"; Scope = @( "Install\Install"; "Install\WinRE"; "Boot\Boot" ) }
)

Function Extract_Language
{
    param( $Act, $NewLang, $Expand )

    Function Match_Required_Fonts
    {
        param( $Lang )

        $Fonts = @(
            @{ Match = @("as", "ar-SA", "ar", "ar-AE", "ar-BH", "ar-DJ", "ar-DZ", "ar-EG", "ar-ER", "ar-IL", "ar-IQ", "ar-JO", "ar-KM", "ar-KW", "ar-LB", "ar-
LY", "ar-MA", "ar-MR", "ar-OM", "ar-PS", "ar-QA", "ar-SD", "ar-SO", "ar-SS", "ar-SY", "ar-TD", "ar-TN", "ar-YE", "arz-Arab", "ckb-Arab", "fa", "fa-AF",
"fa-IR", "glk-Arab", "ha-Arab", "ks-Arab", "ks-Arab-IN", "ku-Arab", "ku-Arab-IQ", "mzn-Arab", "pa-Arab", "pa-Arab-PK", "pnb-Arab", "prs", "prs-AF",
```

```
"prs-Arab", "ps", "ps-AF", "sd-Arab", "sd-Arab-PK", "tk-Arab", "ug", "ug-Arab", "ug-CN", "ur", "ur-IN", "ur-PK", "uz-Arab", "uz-Arab-AF"); Name =
"Arab"; }

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

)

ForEach ($item in $Fonts) {
    if (($item.Match) -Contains $Lang) {
        return $item.Name
    }
}

return "Not_matched"
}

Function Match_Other_Region_Specific_Requirements
{
    param( $Lang )

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

    ForEach ($item in $Fonts) {
        if (($item.Match) -Contains $Lang) {
            return $item.Name
        }
    }

    return "Skip_specific_packages"
}

Function Extract_Process
{
```

```
param( $Package, $Name, $NewSaveTo )

$NewSaveTo = "$($SaveTo)\$($NewSaveTo)\Language\$($Act)\$($NewLang)"
New-Item -Path $NewSaveTo -ItemType Directory -ErrorAction SilentlyContinue | Out-Null

if ($ISO -eq "Auto") {
    Get-PSDrive -PSProvider FileSystem -ErrorAction SilentlyContinue | ForEach-Object {
        ForEach ($item in $Package) {
            $TempFilePath = Join-Path -Path $_.Root -ChildPath $item -ErrorAction SilentlyContinue
            if (Test-Path $TempFilePath -PathType Leaf) {
                Write-host "`n Find: " -NoNewLine; Write-host $TempFilePath -ForegroundColor Green
                Write-host " Copy to: " -NoNewLine; Write-host $NewSaveTo
                Copy-Item -Path $TempFilePath -Destination $NewSaveTo -Force
            }
        }
    }
} else {
    ForEach ($item in $Package) {
        $TempFilePath = Join-Path -Path $ISO -ChildPath $item -ErrorAction SilentlyContinue
        Write-host "`n Find: " -NoNewLine; Write-host $TempFilePath -ForegroundColor Green
        if (Test-Path $TempFilePath -PathType Leaf) {
            Write-host " Copy to: " -NoNewLine; Write-host $NewSaveTo
            Copy-Item -Path $TempFilePath -Destination $NewSaveTo -Force
        } else {
            Write-host " Not found"
        }
    }
}

Write-host "`n Verify the language pack file"
ForEach ($item in $Package) {
    $Path = "$($NewSaveTo)\$([IO.Path]::GetFileName($item))"
    if (Test-Path $Path -PathType Leaf) {
        Write-host " Discover: " -NoNewLine; Write-host $Path -ForegroundColor Green
    } else {
        Write-host " Not found: " -NoNewLine; Write-host $Path -ForegroundColor Red
    }
}

$AdvLanguage = @(
    @(
        Path = "Install\Install"
        Rule = @(
            "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-Fonts-{DiyLang}-Package~31bf3856ad364e35~AMD64~~.cab"
            "LanguagesAndOptionalFeatures\Microsoft-Windows-Client-Language-Pack_x64_{Lang}.cab"
            "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-Basic-{Lang}-Package~31bf3856ad364e35~AMD64~~.cab"
            "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-Handwriting-{Lang}-
Package~31bf3856ad364e35~AMD64~~.cab"
            "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-OCR-{Lang}-Package~31bf3856ad364e35~AMD64~~.cab"
            "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-Speech-{Lang}-Package~31bf3856ad364e35~AMD64~~.cab"
            "LanguagesAndOptionalFeatures\Microsoft-Windows-LanguageFeatures-TextToSpeech-{Lang}-
Package~31bf3856ad364e35~AMD64~~.cab"
            "LanguagesAndOptionalFeatures\Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"
            "LanguagesAndOptionalFeatures\Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"
            "LanguagesAndOptionalFeatures\Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~{Lang}~.cab"
            "LanguagesAndOptionalFeatures\Microsoft-Windows-MediaPlayer-Package-AMD64-{Lang}.cab"
            "LanguagesAndOptionalFeatures\Microsoft-Windows-MediaPlayer-Package-wow64-{Lang}.cab"
```

```

"LanguagesAndOptionalFeatures\Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"
"LanguagesAndOptionalFeatures\Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~{Lang}~.cab"
"LanguagesAndOptionalFeatures\Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"
"LanguagesAndOptionalFeatures\Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"
"LanguagesAndOptionalFeatures\Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~wow64~{Lang}~.cab"
"LanguagesAndOptionalFeatures\Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"
"LanguagesAndOptionalFeatures\Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~{Lang}~.cab"
"LanguagesAndOptionalFeatures\Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~AMD64~{Lang}~.cab"
"LanguagesAndOptionalFeatures\Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~{Lang}~.cab"
"LanguagesAndOptionalFeatures\Microsoft-Windows-InternationalFeatures-{Specific}-Package~31bf3856ad364e35~amd64~~.cab"
)
}
@{
    Path = "Install\WinRE"
    Rule = @(
        "Windows Preinstallation Environment\x64\WinPE_OCs\WinPE-FontSupport-{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\lp.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-securestartup_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-atbroker_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-audiocore_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-audiodrivers_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-enhancedstorage_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-narrator_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-scripting_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-speech-tts_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-srh_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-srt_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-wds-tools_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-wmi_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-appxdeployment_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-appxpackaging_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-storagewmi_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-wifi_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-windowsupdate_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-rejuv_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-opcservices_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-hta_{Lang}.cab"
    )
}
@{
    Path = "Boot\Boot"
    Rule = @(
        "Windows Preinstallation Environment\x64\WinPE_OCs\WinPE-FontSupport-{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\lp.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\WinPE-Setup_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\WINPE-SETUP-CLIENT_{Lang}.CAB"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-securestartup_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-atbroker_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-audiocore_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-audiodrivers_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-enhancedstorage_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-narrator_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-scripting_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-speech-tts_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-srh_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-srt_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-wds-tools_{Lang}.cab"
        "Windows Preinstallation Environment\x64\WinPE_OCs\{Lang}\winpe-wmi_{Lang}.cab"
    )
}

```

```
    )
  }
)
$NewFonts = Match_Required_Fonts -Lang $NewLang
$SpecificPackage = Match_Other_Region_Specific_Requirements -Lang $NewLang
Foreach ($item in $Expand) {
  $Language = @()
  Foreach ($itemList in $AdvLanguage) {
    if ($itemList.Path -eq $item) {
      Foreach ($PrintLang in $itemList.Rule) {
        $Language += "$($PrintLang)".Replace("{Lang}", $NewLang).Replace("{DiyLang}", $NewFonts).Replace("{Specific}", $SpecificPackage)
      }
      Extract_Process -NewSaveTo $itemList.Path -Package $Language -Name $item
    }
  }
}
}
ForEach ($item in $Extract_language_Pack) {
  Extract_Language -Act $item.Act -NewLang $item.Tag -Expand $item.Scope
}
```

C. 自定义部署映像

II 自定义部署映像：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:\OS11\Sources\Install.wim"
Get-WindowsImage -ImagePath $ViewFile | Foreach-Object {
  Get-WindowsImage -ImagePath $ViewFile -index $_.ImageIndex
}
```

[循环操作区域](#)，开始，

2. 指定挂载 Install 路径

```
New-Item -Path "D:\OS11_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:\OS11\sources\install.wim" -Index "1" -Path "D:\OS11_Custom\Install\Install\Mount"
```

[处理 Install.wim 映像内的文件](#)，可选项，开始，

3.1. 自定义部署映像：WinRE.wim

注意：

- WinRE.wim 属于 Install.wim 映像内的文件；

- Install.wim 有多个索引号时，仅处理任意一个 WinRE.wim 即可；
- 同步至所有索引号即可减少 Install.wim 体积；学习 “[如何批量替换 Install.wim 里的所有索引号里的 WinRE.wim](#)”

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:\OS11_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:\OS11_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:\OS11_Custom\Install\Install\Mount\Windows\System32\Recovery\WinRE.wim"
-Index "1" -Path "D:\OS11_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。

3.1.4.1. 语言包：添加

复制代码或查看源文件：[WinRE.Instl.lang.ps1](#) ([本地](#), [Github](#))

```
$Mount = "D:\OS11_Custom\Install\WinRE\Mount"
$Sources = "D:\OS11_Custom\Install\WinRE\Language\Add\zh-CN"

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

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

$Language_List = @(
    @{ Match = "*WinPE-LanguagePack-Package*"; File = "lp.cab"; }
    @{ Match = "*SecureStartup*"; File = "winpe-securestartup_zh-CN.cab"; }
    @{ Match = "*ATBroker*"; File = "winpe-atbroker_zh-CN.cab"; }
    @{ Match = "*AudioCore*"; File = "winpe-audiocore_zh-CN.cab"; }
    @{ Match = "*AudioDrivers*"; File = "winpe-audiodrivers_zh-CN.cab"; }
    @{ Match = "*EnhancedStorage*"; File = "winpe-enhancedstorage_zh-CN.cab"; }
    @{ Match = "*Narrator*"; File = "winpe-narrator_zh-CN.cab"; }
```

```
@{ Match = "*scripting*"; File = "winpe-scripting_zh-CN.cab"; }
@{ Match = "*Speech-TTS*"; File = "winpe-speech-tts_zh-CN.cab"; }
@{ Match = "*srh*"; File = "winpe-srh_zh-CN.cab"; }
@{ Match = "*srt*"; File = "winpe-srt_zh-CN.cab"; }
@{ Match = "*wds-tools*"; File = "winpe-wds-tools_zh-CN.cab"; }
@{ Match = "*-WMI-Package*"; File = "winpe-wmi_zh-CN.cab"; }
@{ Match = "*WinPE-AppxPackaging*"; File = "winpe-appxpackaging_zh-CN.cab"; }
@{ Match = "*StorageWMI*"; File = "winpe-storagewmi_zh-CN.cab"; }
@{ Match = "*WiFi*"; File = "winpe-wifi_zh-CN.cab"; }
@{ Match = "*rejuv*"; File = "winpe-rejuv_zh-CN.cab"; }
@{ Match = "*opcservices*"; File = "winpe-opcservices_zh-CN.cab"; }
@{ Match = "*hta*"; File = "winpe-hta_zh-CN.cab"; }
@{ Match = "*windowsupdate*"; File = "winpe-windowsupdate_zh-CN.cab"; }
@{ Match = "*appxdeployment*"; File = "winpe-appxdeployment_zh-CN.cab"; }
)

ForEach ($Rule in $Language_List) {
    Write-host "`n Rule name: $($Rule.Match)" -ForegroundColor Yellow; Write-host "  $('-' * 80)"
    ForEach ($Component in $Initl_install_Language_Component) {
        if ($Component -like "*$($Rule.Match)*") {
            Write-host "  Component name: " -NoNewline
            Write-host $Component -ForegroundColor Green
            Write-host "  Language pack file: " -NoNewline
            Write-host "$($Sources)\$($Rule.File)" -ForegroundColor Green

            Write-Host "  Installing ".PadRight(22) -NoNewline
            try {
                Add-WindowsPackage -Path $Mount -PackagePath "$($Sources)\$($Rule.File)" | Out-Null
                Write-host "Finish" -ForegroundColor Green
            } catch {
                Write-host "Failed" -ForegroundColor Red
            }
            break
        }
    }
}
}
```

3.1.4.2. 脱机映像语言：更改

3.1.4.2.1. 更改默认语言、区域设置和其他国际设置

```
区域： zh-CN

Dism /Image:"D:\OS11_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:\OS11_Custom\Install\WinRE\Mount" /Get-Intl
```

3.1.4.3. 语言包：删除，可选

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

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

复制代码或查看源文件： [WinRE.Del.Specified.lang.Tag.ps1](#) ([本地](#), [Github](#))

```
$Lang = "en-US"
$Mount = "D:\OS11_Custom\Install\WinRE\Mount"
$Sources = "D:\OS11_Custom\Install\WinRE\Language\Del\en-US"

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

$Language = @(
    @{ Match = "*windowsupdate*"; File = "winpe-windowsupdate_$(Lang).cab"; }
    @{ Match = "*appxdeployment*"; File = "winpe-appxdeployment_$(Lang).cab"; }
    @{ Match = "*hta*"; File = "winpe-hta_$(Lang).cab"; }
    @{ Match = "*opcservices*"; File = "winpe-opcservices_$(Lang).cab"; }
    @{ Match = "*rejuv*"; File = "winpe-rejuv_$(Lang).cab"; }
    @{ Match = "*WiFi*"; File = "winpe-wifi_$(Lang).cab"; }
    @{ Match = "*StorageWMI*"; File = "winpe-storagewmi_$(Lang).cab"; }
    @{ Match = "*WinPE-AppxPackaging*"; File = "winpe-appxpackaging_$(Lang).cab"; }
    @{ Match = "*-WMI-Package*"; File = "winpe-wmi_$(Lang).cab"; }
    @{ Match = "*wds-tools*"; File = "winpe-wds-tools_$(Lang).cab"; }
    @{ Match = "*srt*"; File = "winpe-srt_$(Lang).cab"; }
    @{ Match = "*srh*"; File = "winpe-srh_$(Lang).cab"; }
    @{ Match = "*Speech-TTS*"; File = "winpe-speech-tts_$(Lang).cab"; }
    @{ Match = "*scripting*"; File = "winpe-scripting_$(Lang).cab"; }
    @{ Match = "*Narrator*"; File = "winpe-narrator_$(Lang).cab"; }
    @{ Match = "*EnhancedStorage*"; File = "winpe-enhancedstorage_$(Lang).cab"; }
    @{ Match = "*AudioDrivers*"; File = "winpe-audiodrivers_$(Lang).cab"; }
    @{ Match = "*AudioCore*"; File = "winpe-audiocore_$(Lang).cab"; }
    @{ Match = "*ATBroker*"; File = "winpe-atbroker_$(Lang).cab"; }
    @{ Match = "*SecureStartup*"; File = "winpe-securestartup_$(Lang).cab"; }
    @{ Match = "*WinPE-LanguagePack-Package*"; File = "lp.cab"; }
)

ForEach ($Rule in $Language) {
    Write-host "`n Rule name: $($Rule.Match)" -ForegroundColor Yellow; Write-host "  $('-' * 80)"
    ForEach ($Component in $Initl_install_Language_Component) {
        if ($Component -like "*$($Rule.Match)*$(Lang)*") {
            Write-host "  Component name: " -NoNewline
            Write-host $Component -ForegroundColor Green
            Write-host "  Language pack file: " -NoNewline
            Write-host "$($Sources)\$($Rule.File)" -ForegroundColor Green

            Write-Host "  Deleting ".PadRight(22) -NoNewline
            try {
                Remove-WindowsPackage -Path $Mount -PackagePath "$($Sources)\$($Rule.File)" -ErrorAction
                SilentlyContinue | Out-Null
            } catch {
                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:\OS11_Custom\Install\WinRE\Mount" | Out-GridView
```

3.1.4.4.2. 导出到 Csv

```
$SaveTo = "D:\OS11_Custom\Install\WinRE\Report.${(Get-Date -Format
"yyyyMMddHHmmss")}.csv"
Get-WindowsPackage -Path "D:\OS11_Custom\Install\WinRE\Mount" | Export-CSV -
NoType -Path $SaveTo
Write-host $SaveTo -ForegroundColor Green
```

3.1.5. 累积更新，可选

准备可用的累积更新文件，请更改示例文件名：KB_WinRE.cab

3.1.5.1. 添加

```
$KBPath = "D:\OS11_Custom\Install\WinRE\Update\KB_WinRE.cab"
Add-WindowsPackage -Path "D:\OS11_Custom\Install\WinRE\Mount" -PackagePath $KBPath
```

3.1.5.2. **删除**

```
$KBPath = "D:\OS11_Custom\Install\WinRE\Update\KB_WinRE.cab"
Remove-WindowsPackage -Path "D:\OS11_Custom\Install\WinRE\Mount" -PackagePath $KBPath
```

3.1.5.3. **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:\OS11_Custom\Install\WinRE\Mount" /cleanup-image /StartComponentCleanup
/ResetBase
```

3.1.5.3.1. **固化更新后清理组件，可选项**

```
$Mount = "D:\OS11_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，可选**

3.1.7. **保存映像**

```
Save-WindowsImage -Path "D:\OS11_Custom\Install\WinRE\Mount"
```

3.1.8. **卸载映像**

关闭所有可能正在访问映像中文件的应用程序，包括文件资源管理器。

```
Dismount-WindowsImage -Path "D:\OS11_Custom\Install\WinRE\Mount" -Discard
```

3.1.9. **重建 WinRE.wim 后，可缩小文件大小**

复制代码或查看源文件：[WinRE.Rebuild.ps1](#) ([本地](#), [Github](#))

```
$FileName = "D:\OS11_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

复制代码或查看源文件：[WinRE.Backup.ps1](#) ([本地](#), [Github](#))

```
$WimLibPath = "D:\OS11_Custom\Install\Install\Update\Winlib"
$FileName = "D:\OS11_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。

4.1. 语言包：添加

复制代码或查看源文件：[Install.Instl.lang.ps1](#) ([本地](#), [Github](#))

```
$Mount = "D:\OS11_Custom\Install\Install\Mount"
$Sources = "D:\OS11_Custom\Install\Install\Language\Add\zh-CN"
$Initl_install_Language_Component = @()
Get-WindowsPackage -Path $Mount | ForEach-Object {
    $Initl_install_Language_Component += $_.PackageName
}
Add-WindowsPackage -Path $Mount -PackagePath "$($Sources)\Microsoft-Windows-LanguageFeatures-Fonts-Hans-
Package~31bf3856ad364e35~amd64~~.cab"
$Language_List = @(
    @{ Match = "*Client-LanguagePack-Package*"; File = "Microsoft-Windows-Client-Language-Pack_x64_zh-CN.cab"; }
    @{ Match = "*LanguageFeatures-Basic*"; File = "Microsoft-Windows-LanguageFeatures-Basic-zh-CN-
Package~31bf3856ad364e35~amd64~~.cab"; }
    @{ Match = "*Handwriting*"; File = "Microsoft-Windows-LanguageFeatures-Handwriting-zh-CN-
Package~31bf3856ad364e35~amd64~~.cab"; }
    @{ Match = "*OCR*"; File = "Microsoft-Windows-LanguageFeatures-OCR-zh-CN-
Package~31bf3856ad364e35~amd64~~.cab"; }
    @{ Match = "*LanguageFeatures-Speech*"; File = "Microsoft-Windows-LanguageFeatures-Speech-zh-CN-
Package~31bf3856ad364e35~amd64~~.cab"; }
    @{ Match = "*TextToSpeech*"; File = "Microsoft-Windows-LanguageFeatures-TextToSpeech-zh-CN-
Package~31bf3856ad364e35~amd64~~.cab"; }
```

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

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

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

@{ Match = "*MediaPlayer*amd64*"; File = "Microsoft-Windows-MediaPlayer-Package-AMD64-zh-CN.cab"; }
@{ Match = "*MediaPlayer*wow64*"; File = "Microsoft-Windows-MediaPlayer-Package-wow64-zh-CN.cab"; }
@{ Match = "*PowerShell*amd64*"; File = "Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~zh-
CN~.cab"; }
@{ Match = "*PowerShell*wow64*"; File = "Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~zh-
CN~.cab"; }

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

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

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

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

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

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

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

@{ Match = "*Client*LanguagePack*zh-TW*"; File = "Microsoft-Windows-InternationalFeatures-Taiwan-
Package~31bf3856ad364e35~amd64~~~.cab"; }
)
ForEach ($Rule in $Language_List) {
    Write-host "`n Rule name: $($Rule.Match)" -ForegroundColor Yellow; Write-host "  $('-' * 80)"
    ForEach ($Component in $Initl_install_Language_Component) {
        if ($Component -like "*$($Rule.Match)*") {
            Write-host "  Component name: " -NoNewline
            Write-host $Component -ForegroundColor Green
            Write-host "  Language pack file: " -NoNewline
            Write-host "$($Sources)\$($Rule.File)" -ForegroundColor Green
            Write-Host "  Installing ".PadRight(22) -NoNewline
            try {
                Add-WindowsPackage -Path $Mount -PackagePath "$($Sources)\$($Rule.File)" | Out-Null
                Write-host "Finish" -ForegroundColor Green
            } catch {
                Write-host "Failed" -ForegroundColor Red
            }
            break
        }
    }
}
```

4.2. 脱机映像语言：更改

- 从 Windows 11 开始，DISM 设置的默认系统 UI 语言在所有版本中保持不变（家庭版除外）。对于所有商业版，在开箱即用体验 (OOBE) 期间选择的语言会设置为系统首选 UI 语言，Windows 将以此语言显示；对于家庭版，在 OOBE 期间选择的语言将继续用作默认系统 UI 语言。

- 从 Windows 10 版本 2004 开始，如果将基于 .appx 的语言体验包 (LXP) 支持的语言作为参数传递，则该语言将设置为系统首选 UI 语言，其父语言将设置为默认系统 UI 语言。在以前的版本中，仅支持基于 .cab 的语言包。

4.2.1. 更改默认语言、区域设置和其他国际设置

```
区域: zh-CN

Dism /Image:"D:\OS11_Custom\Install\Install\Mount" /Set-AllIntl:"zh-CN"
```

4.2.2. 查看可用的语言设置

```
Dism /Image:"D:\OS11_Custom\Install\Install\Mount" /Get-Intl
```

4.3. 语言包: 删除, 可选

- 添加语言后，如果要部署到非英语区域，可通过删除英语语言组件来节省空间。删除语言时，请按与添加语言组件相反的顺序卸载语言组件。
- 添加中文后，反向删除 “英语 - 美国”，区域: en-US，需提前提取语言包

复制代码或查看源文件: `Install.Del.Specified.lang.Tag.ps1` ([本地](#), [Github](#))

```
$Lang = "en-US"
$Mount = "D:\OS11_Custom\Install\Install\Mount"
$Sources = "D:\OS11_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:\OS11_Custom\Install\Install\Mount" | Out-GridView
```

4.4.2. 导出到 Csv

```
$SaveTo = "D:\OS11_Custom\Install\Install\Report.$(Get-Date -Format "yyyyMMddHHmmss").csv"
Get-WindowsPackage -Path "D:\OS11_Custom\Install\Install\Mount" | Export-CSV -NoType -Path $SaveTo
Write-host $SaveTo -ForegroundColor Green
```

5. InBox Apps

5.1. 学习 “附件：inBox Apps”

了解不同的操作系统预安装应用，应用商店连接。

5.2. InBox Apps：已安装

5.2.1. 查看

```
Get-AppXProvisionedPackage -Path "D:\OS11_Custom\Install\Install\Mount" | Out-GridView
```

5.2.2. 导出到 Csv

```
$SaveTo = "D:\OS11_Custom\Install\Install\Report.$(Get-Date -Format "yyyyMMddHHmmss").csv"
Get-AppXProvisionedPackage -Path "D:\OS11_Custom\Install\Install\Mount" | Export-CSV -NoType -Path $SaveTo
Write-host $SaveTo -ForegroundColor Green
```

5.3. 删除已安装的所有预应用程序

复制代码或查看源文件：[Install.InBox.Appx.Clear.all.ps1](#) ([本地](#), [Github](#))

```
Get-AppXProvisionedPackage -path "D:\OS11_Custom\Install\Install\Mount" -ErrorAction SilentlyContinue | ForEach-Object {
    Write-host "`n $_.DisplayName)"
    Write-Host "  Deleting ".PadRight(22) -NoNewline
    try {
        Remove-AppxProvisionedPackage -Path "D:\OS11_Custom\Install\Install\Mount" -PackageName $_.PackageName -
ErrorAction SilentlyContinue | Out-Null
        Write-host "Finish" -ForegroundColor Green
    } catch {
        Write-host "Failed" -ForegroundColor Red
    }
}
```

5.4. 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:\OS11_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。

5.5. 区域标记：添加方式

5.5.1. 执行 “**语言包：添加**”

5.5.2. 安装 “本地语言体验包（LXPs）”

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

5.5.2.1. 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:\OS11_Custom\Install\Install\InBox.Appx**，文件格式：**LanguageExperiencePack.zh-CN.Neutral.Appx**

5.5.2.2. 手动下载

5.5.2.2.1. 查看 “**附件：LXPs**”

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

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

5.5.2.2.2.1. 搜索关键词：

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

5.5.2.2.2.2. 网页内搜索 22621 内容，搜索结果：

Microsoft.LanguageExperiencePackzh-CN_22621.*.neutral__8wekyb3d8bbwe.appx

5.5.2.2.2.3. 下载后保存到 **D:\OS11_Custom\Install\Install\InBox.Appx** 目录里，重命名：**LanguageExperiencePack.zh-cn.Neutral.Appx**

5.5.2.3. **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:\OS11_Custom\Install\Install\Mount" -PackagePath
"D:\OS11_Custom\Install\Install\InBox.appx\LanguageExperiencePack.zh-cn.Neutral.appx" -
SkipLicense
```

5.5.2.4. **InBox Apps：已安装的应用程序包**

5.5.2.4.1. 查看

```
Get-AppXProvisionedPackage -Path "D:\OS11_Custom\Install\Install\Mount" | Out-GridView
```

5.5.2.4.2. 导出出 Csv

```
$SaveTo = "D:\OS11_Custom\Install\Install\Report.$(Get-Date -Format
"yyyyMMddHHmmss").csv"
Get-AppXProvisionedPackage -Path "D:\OS11_Custom\Install\Install\Mount" | Export-CSV -
NoType -Path $SaveTo
Write-host $SaveTo -ForegroundColor Green
```

5.6. **InBox Apps：安装**

5.6.1. **挂载或解压 InBox Apps 安装文件**

挂载 `22621.1.220506-1250.ni_release_amd64fre_InboxApps.iso` 或解压到任意位置；

5.6.2. **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"`

复制代码或查看源文件：`Install.Inst.InBox.Appx.ps1` ([本地](#), [Github](#))

```
$ISO = "Auto"
$Mount = "D:\OS11_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.WebplImageExtension";
"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.WebplImageExtension";
"Microsoft.WindowsStore"; "Microsoft.GamingApp"; "Microsoft.Sticky.Notes"; "Microsoft.Paint";
"Microsoft.PowerAutomateDesktop"; "Microsoft.ScreenSketch"; "Microsoft.WindowsNotepad";
"Microsoft.WindowsTerminal"; "Clipchamp.Clipchamp"; "Microsoft.Solitaire.Collection"; "Microsoft.WindowsAlarms";
"Microsoft.WindowsFeedbackHub"; "Microsoft.WindowsMaps"; "Microsoft.ZuneMusic"; "Microsoft.BingNews";
"Microsoft.DesktopAppInstaller"; "Microsoft.WindowsCamera"; "Microsoft.Getstarted"; "Microsoft.Cortana";
"Microsoft.BingWeather"; "Microsoft.GetHelp"; "Microsoft.MicrosoftOfficeHub"; "Microsoft.People";
"Microsoft.StorePurchaseApp"; "Microsoft.Todos"; "Microsoft.WebMediaExtensions"; "Microsoft.Windows.Photos";
"Microsoft.WindowsCalculator"; "Microsoft.Windows.CommunicationsApps"; "Microsoft.WindowsSoundRecorder";
"Microsoft.Xbox.TCUI"; "Microsoft.XboxGameOverlay"; "Microsoft.XboxGamingOverlay";
"Microsoft.XboxIdentityProvider"; "Microsoft.XboxSpeechToTextOverlay"; "Microsoft.YourPhone";
"Microsoft.ZuneVideo"; "MicrosoftCorporationII.QuickAssist"; "MicrosoftWindows.Client.WebExperience";
"Microsoft.RawImageExtension";
        )
    }
    @{
        Name = @(
            "EnterpriseN"; "EnterpriseGN"; "EnterpriseSN"; "ProfessionalN"; "EducationN"; "ProfessionalWorkstationN";
"ProfessionalEducationN"; "CloudN"; "CloudEN"; "CloudEditionN"; "CloudEditionLN"; "StarterN";
        )
        Apps = @(
            "Microsoft.UI.Xaml.2.3"; "Microsoft.UI.Xaml.2.4"; "Microsoft.UI.Xaml.2.7";
"Microsoft.NET.Native.Framework.2.2"; "Microsoft.NET.Native.Runtime.2.2"; "Microsoft.VCLibs.140.00";
"Microsoft.VCLibs.140.00.UWPDesktop"; "Microsoft.SecHealthUI"; "Microsoft.WindowsStore";
"Microsoft.Sticky.Notes"; "Microsoft.Paint"; "Microsoft.PowerAutomateDesktop"; "Microsoft.ScreenSketch";
"Microsoft.WindowsNotepad"; "Microsoft.WindowsTerminal"; "Clipchamp.Clipchamp";
```

```
"Microsoft.Solitaire.Collection"; "Microsoft.WindowsAlarms"; "Microsoft.WindowsFeedbackHub";
"Microsoft.WindowsMaps"; "Microsoft.BingNews"; "Microsoft.DesktopAppInstaller"; "Microsoft.WindowsCamera";
"Microsoft.Getstarted"; "Microsoft.Cortana"; "Microsoft.BingWeather"; "Microsoft.GetHelp";
"Microsoft.MicrosoftOfficeHub"; "Microsoft.People"; "Microsoft.StorePurchaseApp"; "Microsoft.Todos";
"Microsoft.Windows.Photos"; "Microsoft.WindowsCalculator"; "Microsoft.Windows.CommunicationsApps";
"Microsoft.XboxGameOverlay"; "Microsoft.XboxIdentityProvider"; "Microsoft.XboxSpeechToTextOverlay";
"Microsoft.YourPhone"; "MicrosoftCorporationII.QuickAssist"; "MicrosoftWindows.Client.WebExperience";
    )
}
)
```

```
$Allow_Install_App = @()
ForEach ($item in $Pre_Config_Rules) {
    if ($item.Name -contains $Current_Edition_Version) {
        Write-host "`n  Match to: " -NoNewline
        Write-host $Current_Edition_Version -ForegroundColor Green

        $Allow_Install_App = $item.Apps
        break
    }
}
```

```
Write-host "`n  The app to install ( $($Allow_Install_App.Count) item )" -ForegroundColor Yellow
Write-host "  $('-' * 80)"
ForEach ($item in $Allow_Install_App) {
    Write-host "  $($item)" -ForegroundColor Green
}
```

```
$InBoxApps = @(
    @{ Name = "Microsoft.UI.Xaml.2.3"; File = "Microsoft.UI.Xaml.x64.2.3.appx"; License = ""; }
    @{ Name = "Microsoft.UI.Xaml.2.4"; File = "Microsoft.UI.Xaml.x64.2.4.appx"; License = ""; }
    @{ Name = "Microsoft.UI.Xaml.2.7"; File = "Microsoft.UI.Xaml.x64.2.7.appx"; License = ""; }
    @{ Name = "Microsoft.NET.Native.Framework.2.2"; File = "Microsoft.NET.Native.Framework.x64.2.2.appx"; License =
""; }
    @{ Name = "Microsoft.NET.Native.Runtime.2.2"; File = "Microsoft.NET.Native.Runtime.x64.2.2.appx"; License = ""; }
    @{ Name = "Microsoft.VCLibs.140.00"; File = "Microsoft.VCLibs.x64.14.00.appx"; License = ""; }
    @{ Name = "Microsoft.VCLibs.140.00.UWPDesktop"; File = "Microsoft.VCLibs.x64.14.00.UWPDesktop.appx";
License = ""; }
    @{ Name = "Microsoft.WindowsStore"; File = "Microsoft.WindowsStore_8wekyb3d8bbwe.msixbundle"; License =
"Microsoft.WindowsStore_8wekyb3d8bbwe.xml"; }
    @{ Name = "Microsoft.HEIFImageExtension"; File = "Microsoft.HEIFImageExtension_8wekyb3d8bbwe.appxbundle";
License = "Microsoft.HEIFImageExtension_8wekyb3d8bbwe.xml"; }
    @{ Name = "Microsoft.HEVCVideoExtension"; File = "Microsoft.HEVCVideoExtension_8wekyb3d8bbwe.x64.appx";
License = "Microsoft.HEVCVideoExtension_8wekyb3d8bbwe.x64.xml"; }
    @{ Name = "Microsoft.SecHealthUI"; File = "Microsoft.SecHealthUI_8wekyb3d8bbwe.x64.appx"; License =
"Microsoft.SecHealthUI_8wekyb3d8bbwe.x64.xml"; }
    @{ Name = "Microsoft.VP9VideoExtensions"; File = "Microsoft.VP9VideoExtensions_8wekyb3d8bbwe.x64.appx";
License = "Microsoft.VP9VideoExtensions_8wekyb3d8bbwe.x64.xml"; }
    @{ Name = "Microsoft.WebplImageExtension"; File = "Microsoft.WebplImageExtension_8wekyb3d8bbwe.x64.appx";
License = "Microsoft.WebplImageExtension_8wekyb3d8bbwe.x64.xml"; }
    @{ Name = "Microsoft.GamingApp"; File = "Microsoft.GamingApp_8wekyb3d8bbwe.msixbundle"; License =
"Microsoft.GamingApp_8wekyb3d8bbwe.xml"; }
    @{ Name = "Microsoft.Sticky.Notes"; File = "Microsoft.MicrosoftStickyNotes_8wekyb3d8bbwe.msixbundle"; License
= "Microsoft.MicrosoftStickyNotes_8wekyb3d8bbwe.xml"; }
    @{ Name = "Microsoft.Paint"; File = "Microsoft.Paint_8wekyb3d8bbwe.msixbundle"; License =
"Microsoft.Paint_8wekyb3d8bbwe.xml"; }
    @{ Name = "Microsoft.PowerAutomateDesktop"; File =
"Microsoft.PowerAutomateDesktop_8wekyb3d8bbwe.msixbundle"; License =
```



```
"Microsoft.PowerAutomateDesktop_8wekyb3d8bbwe.xml"; }

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
"Microsoft.WindowsSoundRecorder_8wekyb3d8bbwe.xml"; }

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

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

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

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

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

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

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

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

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

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

)
```

```
Function Install_Appx
{
    param
    (
        $File,
        $License
    )

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

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

    if (Test-Path -Path $File -PathType Leaf) {
        if (Test-Path -Path $License -PathType Leaf) {
            Write-host "    License: " -NoNewline
            Write-host $License -ForegroundColor Yellow

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

            try {
                Add-AppxProvisionedPackage -Path $Mount -PackagePath $File -LicensePath $License -ErrorAction
SilentlyContinue | Out-Null
                Write-Host "Done" -ForegroundColor Green
            } catch {
                Write-Host "Failed" -ForegroundColor Red
                Write-Host "  $($_) " -ForegroundColor Yellow
            }
        } else {
            Write-host "    No License".PadRight(22) -NoNewline -ForegroundColor Red
            Write-host "    Installing".PadRight(22) -NoNewline

            try {
                Add-AppxProvisionedPackage -Path $Mount -PackagePath $File -SkipLicense -ErrorAction SilentlyContinue |
```



```
Out-Null

        Write-Host "Done" -ForegroundColor Green
    } catch {
        Write-Host "Failed" -ForegroundColor Red
        Write-Host "  $($_) " -ForegroundColor Yellow
    }
}

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

}

ForEach ($Rule in $InBoxApps) {
    Write-host "`n  Name: " -NoNewline
    Write-host $Rule.Name -ForegroundColor Yellow
    Write-host "  $('-' * 80)"

    if($Allow_Install_App -contains $Rule.Name) {
        Write-host "  Search for apps: " -NoNewline
        Write-host $Rule.File -ForegroundColor Yellow
        Write-host "  Search for License: " -NoNewline
        Write-host $Rule.File -ForegroundColor Yellow

        if ($ISO -eq "Auto") {
            Get-PSDrive -PSProvider FileSystem -ErrorAction SilentlyContinue | ForEach-Object {
                $AppPath = Join-Path -Path $_.Root -ChildPath "packages\$($Rule.File)" -ErrorAction SilentlyContinue
                $LicensePath = Join-Path -Path $_.Root -ChildPath "packages\$($Rule.License)" -ErrorAction SilentlyContinue

                if (Test-Path $AppPath -PathType Leaf) {
                    Write-host "  $('-' * 80)"
                    Write-host "  Discover apps: " -NoNewLine
                    Write-host $AppPath -ForegroundColor Green

                    if (Test-Path $LicensePath -PathType Leaf) {
                        Write-host "  Discover License: " -NoNewLine
                        Write-host $LicensePath -ForegroundColor Green
                    } else {
                        Write-host "  License: " -NoNewLine
                        Write-host "Not found" -ForegroundColor Red
                    }

                    Install_Appx -File $AppPath -License $LicensePath
                    return
                }
            }
        } else {
            Install_Appx -File "$($ISO)\$($Rule.File)" -License "$($ISO)\$($Rule.License)"
        }
    } else {
        Write-host "  Skip the installation" -ForegroundColor Red
    }
}

}
```

5.7. 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。

复制代码或查看源文件: [Install.Clear.Flag.ps1](#) ([本地](#), [Github](#))

```
$Lang = "zh-CN"
Get-AppXProvisionedPackage -Path "D:\OS11_Custom\Install\Install\Mount" | Foreach-object {
    if ($_.DisplayName -Like "*LanguageExperiencePack*$( $Lang)*") {
        Write-host "  $_.DisplayName"
        Write-Host "  Deleting ".PadRight(22) -NoNewline
        try {
            Remove-AppxProvisionedPackage -Path "D:\OS11_Custom\Install\Install\Mount" -PackageName $_.PackageName -
ErrorAction SilentlyContinue | Out-Null
            Write-host "Finish" -ForegroundColor Green
        } catch {
            Write-host "Failed" -ForegroundColor Red
        }
    }
}
```

5.8. **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:\OS11_Custom\Install\Install\Mount" /Optimize-ProvisionedAppxPackages
```

6. **累积更新**

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

6.1. **初始版本**

6.1.1. **如何下载**

累积更新 [KB5016632](#) 已无法从 <https://www.catalog.update.microsoft.com/Search.aspx?q=KB5016632> 里搜索到，其它下载方法，任选一个网站并打开：

- <https://uupdump.net>
- <https://uup.ee>

6.1.1.1. 打开后，搜索关键词：[22621.382](#)，在搜索结果选择：[Windows 11, version 22H2 \(22621.382\) amd64](#)

6.1.1.2. 打开后，选择 “[全部文件](#)” ；

6.1.1.3. 在 “全部文件” 页面里搜索：[KB5016632](#)，下载：

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

6.1.1.4. **合并累积更新**

6.1.1.4.1. 前往 <https://github.com/abbodi1406/WHd/tree/master/scripts> 后下载 [PSFX_Repack_6.zip](#)

6.1.1.4.2. 下载后解压到：[D:\PSFX_Repack_6](#)

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

6.1.1.4.4. 运行： psfx2cab_GUI.cmd 后，将得到一个新的文件：
Windows11.0-KB5016632-x64-full_psfx.cab

6.1.1.4.5. 将 Windows11.0-KB5016632-x64-full_psfx.cab 保存到：
D:\OS11_Custom\Install\Install\Update\Windows11.0-KB5016632-x64-full_psfx.cab

6.1.2. 添加

```
$KBPath = "D:\OS11_Custom\Install\Install\Update\Windows11.0-KB5016632-x64-full_psfx.cab"
Add-WindowsPackage -Path "D:\OS11_Custom\Install\Install\Mount" -PackagePath $KBPath
```

6.2. 其它版本

6.2.1. 如何下载

查阅 “Windows 11 版本信息” ， 例如下载累积更新： KB5030219，前往下载页面：
<https://www.catalog.update.microsoft.com/Search.aspx?q=KB5030219> 或 "直连下载"，保存到：
D:\OS11_Custom\Install\Install\Update\windoOS11.0-kb5030219-
x64_659d407821da654d564a3108bf493594bd143ab1.msu

6.2.2. 添加

```
$KBPath = "D:\OS11_Custom\Install\Install\Update\windoOS11.0-kb5030219-  
x64_659d407821da654d564a3108bf493594bd143ab1.msu"  
Add-WindowsPackage -Path "D:\OS11_Custom\Install\Install\Mount" -PackagePath $KBPath
```

6.3. 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:\OS11_Custom\Install\Install\Mount" /cleanup-image /StartComponentCleanup /ResetBase
```

6.3.1. 固化更新后清理组件，可选项

```
$Mount = "D:\OS11_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，可选

8. 添加部署引擎，可选

- 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:\OS11_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:\OS11_Custom\Install\Install\Update\Winlib\WinRE.wim"
$CopyTo = "D:\OS11_Custom\Install\Install\Mount\Windows\System32\Recovery"
Copy-Item -Path $WinRE -Destination $CopyTo -Force
```

11. 保存映像

```
Save-WindowsImage -Path "D:\OS11_Custom\Install\Install\Mount"
```

12. 卸载映像

关闭所有可能正在访问映像中文件的应用程序，包括文件资源管理器。

```
Dismount-WindowsImage -Path "D:\OS11_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

复制代码或查看源文件：[Install.WinRE.Replace.wim.ps1](#) ([本地](#), [Github](#))

```
$Arguments = @(
    "extract",
    "D:\OS11\sources\install.wim", "1",
    "\Windows\System32\Recovery\Winre.wim",
    "--dest-dir=""D:\OS11_Custom\Install\Install\Update\Winlib""
)
New-Item -Path "D:\OS11_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

复制代码或查看源文件: [Install.WinRE.Replace.wim.ps1](#) ([本地](#), [Github](#))

```
Get-WindowsImage -ImagePath "D:\OS11\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:\OS11\sources\install.wim", $_.ImageIndex,  
        "--command=""add 'D:\OS11_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

复制代码或查看源文件: [Install.Rebuild.wim.ps1](#) ([本地](#), [Github](#))

```
$InstallWim = "D:\OS11\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 自定义部署映像: 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:\OS11\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:\OS11_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:\OS11\sources\boot.wim" -Index "2" -Path "D:\OS11_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。

4.1. 语言包：添加

复制代码或查看源文件：[Boot.Instl.lang.ps1](#) ([本地](#), [Github](#))

```
$Mount = "D:\OS11_Custom\Boot\Boot\Mount"
$Sources = "D:\OS11_Custom\Boot\Boot\Language\Add\zh-CN"

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

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

$Language_List = @(
    @{ Match = "*WinPE*Setup*Client*Package*"; File = "WINPE-SETUP-CLIENT_zh-CN.CAB"; }
    @{ Match = "*WinPE*Setup*Package*"; File = "WinPE-Setup_zh-CN.cab"; }
    @{ Match = "*WinPE-LanguagePack*Package*"; File = "lp.cab"; }
    @{ Match = "*SecureStartup*"; File = "winpe-securestartup_zh-CN.cab"; }
    @{ Match = "*ATBroker*"; File = "winpe-atbroker_zh-CN.cab"; }
    @{ Match = "*AudioCore*"; File = "winpe-audiocore_zh-CN.cab"; }
    @{ Match = "*AudioDrivers*"; File = "winpe-audiodrivers_zh-CN.cab"; }
    @{ Match = "*EnhancedStorage*"; File = "winpe-enhancedstorage_zh-CN.cab"; }
    @{ Match = "*Narrator*"; File = "winpe-narrator_zh-CN.cab"; }
    @{ Match = "*scripting*"; File = "winpe-scripting_zh-CN.cab"; }
    @{ Match = "*Speech-TTS*"; File = "winpe-speech-tts_zh-CN.cab"; }
    @{ Match = "*srh*"; File = "winpe-srh_zh-CN.cab"; }
    @{ Match = "*srt*"; File = "winpe-srt_zh-CN.cab"; }
    @{ Match = "*wds-tools*"; File = "winpe-wds-tools_zh-CN.cab"; }
    @{ Match = "*-WMI-Package*"; File = "winpe-wmi_zh-CN.cab"; }
)

ForEach ($Rule in $Language_List) {
    Write-host "`n Rule name: $($Rule.Match)" -ForegroundColor Yellow; Write-host " $('-' * 80)"
    ForEach ($Component in $Initl_install_Language_Component) {
        if ($Component -like "*$($Rule.Match)*") {
            Write-host " Component name: " -NoNewline
            Write-host $Component -ForegroundColor Green
            Write-host " Language pack file: " -NoNewline
            Write-host "$($Sources)\$($Rule.File)" -ForegroundColor Green
        }
    }
}
```

```
Write-Host " Installing ".PadRight(22) -NoNewline
try{
    Add-WindowsPackage -Path $Mount -PackagePath "$($Sources)\ $($Rule.File)" | Out-Null
    Write-host "Finish" -ForegroundColor Green
} catch {
    Write-host "Failed" -ForegroundColor Red
}

break
}
}
}
```

4.2. 脱机映像语言：更改

4.2.1. 更改默认语言、区域设置和其他国际设置

区域标记：zh-CN

```
Dism /Image:"D:\OS11_Custom\Boot\Boot\Mount" /Set-AllIntl:zh-CN
```

4.2.2. 查看可用的语言设置

```
Dism /Image:"D:\OS11_Custom\Boot\Boot\Mount" /Get-Intl
```

4.3. 语言包：删除，可选

- 添加语言后，如果要部署到非英语区域，可通过删除英语语言组件来节省空间。删除语言时，请按与添加语言组件相反的顺序卸载语言组件。
- 添加中文后，反向删除“英语 - 美国”，区域：en-US，需提前提取语言包

复制代码或查看源文件：Boot.Del.Specified.lang.Tag.ps1 (本地, Github)

```
$Lang = "en-US"
$Mount = "D:\OS11_Custom\Boot\Boot\Mount"
$Sources = "D:\OS11_Custom\Boot\Boot\Language\Del\en-US"

$InitL_install_Language_Component = @()
Get-WindowsPackage -Path $Mount | ForEach-Object {
    $InitL_install_Language_Component += $_.PackageName
}

$Language = @(
    @{ Match = "*WordPad*wow64*"; File = "Microsoft-Windows-WordPad-FoD-
Package~31bf3856ad364e35~wow64~ $($Lang)~.cab"; }
    @{ Match = "*WordPad*amd64*"; File = "Microsoft-Windows-WordPad-FoD-
Package~31bf3856ad364e35~amd64~ $($Lang)~.cab"; }
    @{ Match = "*WMIC*FoD*Package*wow64*"; File = "Microsoft-Windows-WMIC-FoD-
Package~31bf3856ad364e35~wow64~ $($Lang)~.cab"; }
    @{ Match = "*WMIC*FoD*Package*amd64*"; File = "Microsoft-Windows-WMIC-FoD-
Package~31bf3856ad364e35~AMD64~ $($Lang)~.cab"; }
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

    ForEach ($Component in $Initl_install_Language_Component) {
        if ($Component -like "*$($Rule.Match)*$(Lang)*") {
            Write-host "  Component name: " -NoNewline
            Write-host $Component -ForegroundColor Green

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

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

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

                Write-host "Finish" -ForegroundColor Green
            } catch {
                Write-host "Failed" -ForegroundColor Red
                Write-host "  $($_) " -ForegroundColor Red
            }
            break
        }
    }
}
```

```
$InitlClearLanguagePackage = @()
Get-WindowsPackage -Path $Mount | ForEach-Object {
    if ($_.PackageName -like "*$(Lang)*") {
        $InitlClearLanguagePackage += $_.PackageName
    }
}
```



```
    }  
  }  
  
  if ($InitlClearLanguagePackage.count -gt 0) {  
    ForEach ($item in $InitlClearLanguagePackage) {  
      Write-Host "`n $($item)" -ForegroundColor Green  
      Write-Host "  Deleting ".PadRight(22) -NoNewline  
      try {  
        Remove-AppxProvisionedPackage -Path $Mount -PackageName $item -ErrorAction SilentlyContinue | Out-Null  
        Write-host "Finish" -ForegroundColor Green  
      } catch {  
        Write-host "Failed" -ForegroundColor Red  
        Write-host "  $($_) " -ForegroundColor Red  
      }  
    }  
  }  
}
```

4.4. 组件：映像中已安装的所有包

4.4.1. 查看

```
Get-WindowsPackage -Path "D:\OS11_Custom\Boot\Boot\Mount" | Out-GridView
```

4.4.2. 导出到 Csv

```
$SaveTo = "D:\OS11_Custom\Boot\Boot\Report.$(Get-Date -Format "yyyyMMddHHmmss").csv"  
Get-WindowsPackage -Path "D:\OS11_Custom\Boot\Boot\Mount" | Export-CSV -NoType -Path $SaveTo  
Write-host $SaveTo -ForegroundColor Green
```

4.5. 语言包：修复

4.5.1. 提取

打开：[D:\OS11_Custom\Install\Install\Language\Add\zh-CN\Microsoft-Windows-Client-Language-Pack_x64_zh-CN.cab](#)，进入目录：[Setup\sources\zh-cn\cli](#)，复制以下文件到桌面：

4.5.1.1. [arunres.dll.mui](#)

4.5.1.2. [spwizres.dll.mui](#)

4.5.1.3. [w32uires.dll.mui](#)

4.5.2. 复制

将提取的文件复制到：[D:\OS11_Custom\Boot\Boot\Mount\sources\zh-CN](#)

4.6. 语言包：同步到 ISO 安装程序

```
Copy-Item -Path "D:\OS11_Custom\Boot\Boot\Mount\sources\zh-CN" -Destination "D:\OS11\sources\zh-CN" -Recurse -Force
```

4.7. 重新生成 Lang.ini

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

4.7.1. 重新生成已挂载目录 lang.ini

重新生成的 Lang.ini 文件位置：D:\en-us_windows_server_2022_x64_dvd_620d7eac_Custom\Boot\Boot\Mount\Sources\lang.ini

```
Dism /image:"D:\OS11_Custom\Boot\Boot\Mount" /gen-langini /distribution:"D:\OS11_Custom\Boot\Boot\Mount"
```

4.7.2. 重新生成 lang.ini 后，同步到安装程序

重新生成的 Lang.ini 文件位置：D:\en-us_windows_server_2022_x64_dvd_620d7eac\Sources\lang.ini

```
Dism /image:"D:\OS11_Custom\Boot\Boot\Mount" /gen-langini /distribution:"D:\OS11"
```

5. 累积更新，可选

准备可用的累积更新文件，请更改示例文件名：KB_Boot.cab

5.1. 添加

```
$KBPath = "D:\OS11_Custom\Boot\Boot\Update\KB_Boot.cab"
Add-WindowsPackage -Path "D:\OS11_Custom\Boot\Boot\Mount" -PackagePath $KBPath
```

5.2. 删除

```
$KBPath = "D:\OS11_Custom\Boot\Boot\Update\KB_Boot.cab"
Remove-WindowsPackage -Path "D:\OS11_Custom\Boot\Boot\Mount" -PackagePath $KBPath
```

5.3. 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:\OS11_Custom\Boot\Boot\Mount" /cleanup-image /StartComponentCleanup /ResetBase
```

5.3.1. 固化更新后清理组件，可选项

```
$Mount = "D:\OS11_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，可选

7. 其它

7.1. 绕过 TPM

复制代码或查看源文件：[Boot.Bypass.TPM.ps1](#) ([本地](#), [Github](#))

```
$RegSystem = "D:\OS11_Custom\Boot\Boot\Mount\Windows\System32\Config\SYSTEM"
$RandomGuid = [guid]::NewGuid()
Write-Host " HKLM:\$( $RandomGuid )"
New-PSDrive -PSProvider Registry -Name OtherTasksTPM -Root HKLM -ErrorAction SilentlyContinue | Out-Null
Start-Process reg -ArgumentList "Load ""HKLM\$( $RandomGuid)"" ""$( $RegSystem)"" -Wait -WindowStyle Hidden -ErrorAction SilentlyContinue
New-Item "HKLM:\$( $RandomGuid)\Setup\LabConfig" -force -ea SilentlyContinue | Out-Null
New-ItemProperty -LiteralPath "HKLM:\$( $RandomGuid)\Setup\LabConfig" -Name "BypassCPUCheck" -Value 1 -PropertyType DWord -Force -ea SilentlyContinue | Out-Null
New-ItemProperty -LiteralPath "HKLM:\$( $RandomGuid)\Setup\LabConfig" -Name "BypassStorageCheck" -Value 1 -PropertyType DWord -Force -ea SilentlyContinue | Out-Null
New-ItemProperty -LiteralPath "HKLM:\$( $RandomGuid)\Setup\LabConfig" -Name "BypassRAMCheck" -Value 1 -PropertyType DWord -Force -ea SilentlyContinue | Out-Null
New-ItemProperty -LiteralPath "HKLM:\$( $RandomGuid)\Setup\LabConfig" -Name "BypassTPMCheck" -Value 1 -PropertyType DWord -Force -ea SilentlyContinue | Out-Null
New-ItemProperty -LiteralPath "HKLM:\$( $RandomGuid)\Setup\LabConfig" -Name "BypassSecureBootCheck" -Value 1 -PropertyType DWord -Force -ea SilentlyContinue | Out-Null
[gc]::collect()
Start-Process reg -ArgumentList "unload ""HKLM\$( $RandomGuid)"" -Wait -WindowStyle Hidden -ErrorAction SilentlyContinue
Remove-PSDrive -Name OtherTasksTPM
```

8. 保存映像

```
Save-WindowsImage -Path "D:\OS11_Custom\Boot\Boot\Mount"
```

9. 卸载映像

关闭所有可能正在访问映像中文件的应用程序，包括文件资源管理器。

```
Dismount-WindowsImage -Path "D:\OS11_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：[1.1.0.4](#)， 选择下载源代码格式：[zip](#)， 下载完成后得到 [Multilingual-1.1.0.4.zip](#) 压缩包文件；
- 将已下载的 [main.zip](#) 或 [Multilingual-1.1.0.4.zip](#)， 解压到：[D:\Multilingual-1.1.0.4](#)， 重命名：[D:\Multilingual](#)
- 学习 “[无人值守 Windows 安装参考](#)” ， 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\Unattend\Mul.Unattend.xml 到 D:\OS11\Autounattend.xml

```
Copy-Item "D:\Multilingual\Unattend\Mul.Unattend.xml" -Destination "D:\OS11\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\Unattend\Mul.Unattend.xml 到 D:\OS11\Sources\Unattend.xml

```
Copy-Item "D:\Multilingual\Unattend\Mul.Unattend.xml" -Destination "D:\OS11\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:\OS11\sources\`$OEM$\`$$\Panther" -ItemType Directory
```

1.1.1.3.2. 复制

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

D:\OS11\Sources\\$OEM\$\Panther\Unattend.xml

```
Copy-Item "D:\Multilingual\Unattend\Mul.Unattend.xml" -Destination "D:\OS11\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:\OS11\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:\OS11\Sources\\$OEM\$\\$1\Yi\Engine

```
Copy-Item "D:\Multilingual\Engine" -Destination "D:\OS11\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

    "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:\OS11\sources\`$OEM$\`$1\Yi\Engine\Deploy\Allow" -ItemType Directory -

    ErrorAction SilentlyContinue | Out-Null

    Out-File -FilePath "D:\OS11\sources\`$OEM$\`$1\Yi\Engine\Deploy\Allow\$(Item)" -Encoding utf8 -

    ErrorAction SilentlyContinue

}
```

1.2. 添加到已挂载

通过 “自定义部署映像：Install.wim” ， 执行 “开始挂载 Install.wim” ， 挂载到： D:\OS11_Custom\Install\Install\Mount

1.2.1. 无人值守

复制 D:\Multilingual\Unattend\Mul.Unattend.xml 到 D:\OS11_Custom\Install\Install\Mount\Panther\Unattend.xml

```
Copy-Item "D:\Multilingual\Unattend\Mul.Unattend.xml" -Destination
"D:\OS11_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:\OS11_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:\OS11_Custom\Install\Install\Mount\Yi\Engine

```
Copy-Item "D:\Multilingual\Engine" -Destination "D:\OS11_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:\OS11_Custom\Install\Install\Mount\Yi\Engine\Deploy\Allow" -ItemType Directory -
ErrorAction SilentlyContinue | Out-Null

    Out-File -FilePath "D:\OS11_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:\OS11\Autounattend.xml
- D:\OS11\Sources\Unattend.xml
- D:\OS11\sources\\$\OEM\$\\$\\$Panther\unattend.xml
- D:\OS11_Custom\Install\Install\Mount\Panther\Unattend.xml

2.2.1. 多语言或单语

多语言时，单语时，可互相切换，替换时，请替换文件里所有相同的。

2.2.1.1. 多语言

```
<UILanguage>%OSDUILanguage%</UILanguage>
<InputLocale>%OSDInputLocale%</InputLocale>
<SystemLocale>%OSDSystemLocale%</SystemLocale>
<UILanguage>%OSDUILanguage%</UILanguage>
<UILanguageFallback>%OSDUILanguageFallback%</UILanguageFallback>
<UserLocale>%OSDUserLocale%</UserLocale>
```

2.2.1.2. 单语

单语需指定区域，例如指定区域：zh-CN

```
<UILanguage>zh-CN</UILanguage>
<InputLocale>zh-CN</InputLocale>
<SystemLocale>zh-CN</SystemLocale>
<UILanguage>zh-CN</UILanguage>
<UILanguageFallback>zh-CN</UILanguageFallback>
<UserLocale>zh-CN</UserLocale>
```

2.2.2. 用户方案

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

2.2.2.1. 自建用户 Administrator

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

```
<UserAccounts>
  <LocalAccounts>
    <LocalAccount wcm:action="add">
      <Password>
        <Value></Value>
        <PlainText>true</PlainText>
      </Password>
      <Description>Administrator</Description>
      <DisplayName>Administrator</DisplayName>
      <Group>Administrators</Group>
      <Name>Administrator</Name>
    </LocalAccount>
```

```
</LocalAccounts>
</UserAccounts>
<AutoLogon>
  <Password>
    <Value></Value>
    <PlainText>true</PlainText>
  </Password>
  <Enabled>true</Enabled>
  <Username>Administrator</Username>
</AutoLogon>
```

2.2.2.2. 自定义用户

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

2.2.2.2.1. 删除

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

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

2.2.2.2.2. 替换

从开始处 <OOBE> 到 </OOBE>

```
<OOBE>
  <ProtectYourPC>3</ProtectYourPC>
  <HideEULAPage>true</HideEULAPage>
  <HideWirelessSetupInOOBE>true</HideWirelessSetupInOOBE>
</OOBE>
```

D. ISO

II 生成 ISO

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

复制代码或查看源文件：ISO.ps1 (本地, Github)

```
$ISO = "D:\OS11"
$Volume = "OS11"
$SaveTo = "D:\OS11.iso"
$Arguments = @("-m", "-o", "-u2", "-udfver102", "-l" "$($Volume)", "-bootdata:2#p0,e,b" "$($ISO)\boot\etfsboot.com" "#pEF,e,b" "$($ISO)\efi\microsoft\boot\efisys.bin", $ISO, $SaveTo)
Start-Process -FilePath "OScdimg.exe" -ArgumentList $Arguments -wait -nonewwindow
```

III 绕过 TPM 安装检查，可选

1. 了解 <https://github.com/AveYo/MediaCreationTool.bat/tree/main/bypass11> 并下载：Quick_11_iso_esd_wim_TPM_toggle.bat

- 2. 将 D:\OS11.iso 文件托到 Quick_11_iso_esd_wim_TPM_toggle.bat 里，反序“添加”或“删除”TPM 安装时检查功能。

二、 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:\OS11_Custom\Install\Install\Mount" -Discard
```

```
Dismount-WindowsImage -Path "D:\OS11_Custom\Install\WinRE\Mount" -Discard
```

```
Dismount-WindowsImage -Path "D:\OS11_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:\OS11_Custom" -Force -Recurse
```

```
Remove-Item "D:\OS11\Sources\` $OEM$" -Force -Recurse
```

```
Remove-Item -Path "D:\OS11\Autounattend.xml" -Force
```

```
Remove-Item -Path "D:\OS11\Sources\Unattend.xml" -Force
```

三、 已知问题

暂无

四、 组件：映像中已安装的所有包，对比

安装新的语言包后变化， 组件列表：

序号	类型	组件：原始	组件：安装 “简体中文 - 中国” 语言包后变化
1	OnDemandPack	Microsoft-OneCore-ApplicationModel-Sync-Desktop-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.317	Microsoft-OneCore-ApplicationModel-Sync-Desktop-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.317
2	OnDemandPack	Microsoft-OneCore-DirectX-Database-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-OneCore-DirectX-Database-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
3	LanguagePack	Microsoft-Windows-Client-LanguagePack-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.382	Microsoft-Windows-Client-LanguagePack-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.382
4	LanguagePack		Microsoft-Windows-Client-LanguagePack-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.382
5	OnDemandPack	Microsoft-Windows-Ethernet-Client-Intel-E1i68x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Ethernet-Client-Intel-E1i68x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
6	OnDemandPack	Microsoft-Windows-Ethernet-Client-Intel-E2f68-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Ethernet-Client-Intel-E2f68-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
7	OnDemandPack	Microsoft-Windows-Ethernet-Client-Realtek-Rtcx21x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Ethernet-Client-Realtek-Rtcx21x64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
8	OnDemandPack	Microsoft-Windows-Ethernet-Client-Vmware-Vmxnet3-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Ethernet-Client-Vmware-Vmxnet3-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
9	FeaturePack	Microsoft-Windows-FodMetadata-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-FodMetadata-Package~31bf3856ad364e35~amd64~~10.0.22621.1
10	Foundation	Microsoft-Windows-Foundation-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Foundation-Package~31bf3856ad364e35~amd64~~10.0.22621.1
11	OnDemandPack	Microsoft-Windows-Hello-Face-Package~31bf3856ad364e35~amd64~~10.0.22621.317	Microsoft-Windows-Hello-Face-Package~31bf3856ad364e35~amd64~~10.0.22621.317
12	OnDemandPack	Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~en-US~11.0.22621.1	Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~en-US~11.0.22621.1
13	OnDemandPack		Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~zh-CN~11.0.22621.1
14	OnDemandPack	Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~~11.0.22621.1	Microsoft-Windows-InternetExplorer-Optional-Package~31bf3856ad364e35~amd64~~11.0.22621.1
15	OnDemandPack	Microsoft-Windows-Kernel-LA57-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.382	Microsoft-Windows-Kernel-LA57-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.382
16	OnDemandPack	Microsoft-Windows-LanguageFeatures-Basic-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-LanguageFeatures-Basic-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1
17	OnDemandPack		Microsoft-Windows-LanguageFeatures-Basic-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1
18	OnDemandPack		Microsoft-Windows-LanguageFeatures-Fonts-Hans-Package~31bf3856ad364e35~amd64~~10.0.22621.1
19	OnDemandPack	Microsoft-Windows-LanguageFeatures-Handwriting-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-LanguageFeatures-Handwriting-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1
20	OnDemandPack		Microsoft-Windows-LanguageFeatures-Handwriting-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.317
21	OnDemandPack	Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-LanguageFeatures-OCR-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1
22	OnDemandPack		Microsoft-Windows-LanguageFeatures-OCR-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1
23	OnDemandPack	Microsoft-Windows-LanguageFeatures-Speech-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-LanguageFeatures-Speech-en-us-Package~31bf3856ad364e35~amd64~~10.0.22621.1
24	OnDemandPack		Microsoft-Windows-LanguageFeatures-Speech-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1
25	OnDemandPack	Microsoft-Windows-LanguageFeatures-TextToSpeech-en-	Microsoft-Windows-LanguageFeatures-TextToSpeech-en-

序号	类型	组件：原始	组件：安装“简体中文-中国”语言包后变化
		us-Package~31bf3856ad364e35~amd64~~10.0.22621.1	us-Package~31bf3856ad364e35~amd64~~10.0.22621.1
26	OnDemandPack		Microsoft-Windows-LanguageFeatures-TextToSpeech-zh-cn-Package~31bf3856ad364e35~amd64~~10.0.22621.1
27	OnDemandPack	Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.317	Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.317
28	OnDemandPack		Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.317
29	OnDemandPack	Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~~10.0.22621.382	Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~amd64~~10.0.22621.382
30	OnDemandPack	Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1	Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1
31	OnDemandPack		Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1
32	OnDemandPack	Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~~10.0.22621.1	Microsoft-Windows-MediaPlayer-Package~31bf3856ad364e35~wow64~~10.0.22621.1
33	OnDemandPack	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1
34	OnDemandPack		Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
35	OnDemandPack	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
36	OnDemandPack	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1
37	OnDemandPack		Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1
38	OnDemandPack	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1	Microsoft-Windows-Notepad-System-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1
39	OnDemandPack	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1
40	OnDemandPack		Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
41	OnDemandPack	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
42	OnDemandPack	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1
43	OnDemandPack		Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1
44	OnDemandPack	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~~10.0.22621.1	Microsoft-Windows-PowerShell-ISE-FOD-Package~31bf3856ad364e35~wow64~~10.0.22621.1
45	OnDemandPack	Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1	Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1
46	OnDemandPack		Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
47	OnDemandPack	Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Printing-PMCPPC-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
48	OnDemandPack	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1
49	OnDemandPack		Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
50	OnDemandPack	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-StepsRecorder-Package~31bf3856ad364e35~amd64~~10.0.22621.1
51	OnDemandPack	Microsoft-Windows-StepsRecorder-	Microsoft-Windows-StepsRecorder-

序号	类型	组件：原始	组件：安装“简体中文-中国”语言包后变化
		Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1	Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1
52	OnDemandPack		Microsoft-Windows-StepsRecorder- Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1
53	OnDemandPack	Microsoft-Windows-StepsRecorder- Package~31bf3856ad364e35~wow64~~10.0.22621.1	Microsoft-Windows-StepsRecorder- Package~31bf3856ad364e35~wow64~~10.0.22621.1
54	OnDemandPack	Microsoft-Windows-TabletPCMath- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-TabletPCMath- Package~31bf3856ad364e35~amd64~~10.0.22621.1
55	OnDemandPack	Microsoft-Windows-Wallpaper-Content-Extended-FoD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wallpaper-Content-Extended-FoD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
56	OnDemandPack	Microsoft-Windows-Wifi-Client-Broadcom-Bcmpeciedhd63- FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Broadcom-Bcmpeciedhd63- FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
57	OnDemandPack	Microsoft-Windows-Wifi-Client-Broadcom-Bcmwl63a-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Broadcom-Bcmwl63a-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
58	OnDemandPack	Microsoft-Windows-Wifi-Client-Broadcom-Bcmwl63al- FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Broadcom-Bcmwl63al- FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
59	OnDemandPack	Microsoft-Windows-Wifi-Client-Intel-Netwbw02-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Intel-Netwbw02-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
60	OnDemandPack	Microsoft-Windows-Wifi-Client-Intel-Netwew00-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Intel-Netwew00-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
61	OnDemandPack	Microsoft-Windows-Wifi-Client-Intel-Netwew01-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Intel-Netwew01-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
62	OnDemandPack	Microsoft-Windows-Wifi-Client-Intel-Netwlv64-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Intel-Netwlv64-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
63	OnDemandPack	Microsoft-Windows-Wifi-Client-Intel-Netwns64-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Intel-Netwns64-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
64	OnDemandPack	Microsoft-Windows-Wifi-Client-Intel-Netwsw00-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Intel-Netwsw00-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
65	OnDemandPack	Microsoft-Windows-Wifi-Client-Intel-Netwtw02-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Intel-Netwtw02-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
66	OnDemandPack	Microsoft-Windows-Wifi-Client-Intel-Netwtw04-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Intel-Netwtw04-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
67	OnDemandPack	Microsoft-Windows-Wifi-Client-Intel-Netwtw06-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Intel-Netwtw06-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
68	OnDemandPack	Microsoft-Windows-Wifi-Client-Intel-Netwtw08-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Intel-Netwtw08-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
69	OnDemandPack	Microsoft-Windows-Wifi-Client-Intel-Netwtw10-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Intel-Netwtw10-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
70	OnDemandPack	Microsoft-Windows-Wifi-Client-Marvel-Mrvlpcie8897-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Marvel-Mrvlpcie8897-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
71	OnDemandPack	Microsoft-Windows-Wifi-Client-Qualcomm-Athw8x-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Qualcomm-Athw8x-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
72	OnDemandPack	Microsoft-Windows-Wifi-Client-Qualcomm-Athwnx-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Qualcomm-Athwnx-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
73	OnDemandPack	Microsoft-Windows-Wifi-Client-Qualcomm-Qcamain10x64- FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Qualcomm-Qcamain10x64- FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
74	OnDemandPack	Microsoft-Windows-Wifi-Client-Ralink-Netr28x-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Ralink-Netr28x-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
75	OnDemandPack	Microsoft-Windows-Wifi-Client-Realtek-Rtl8187se-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Realtek-Rtl8187se-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
76	OnDemandPack	Microsoft-Windows-Wifi-Client-Realtek-Rtl8192se-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Realtek-Rtl8192se-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1
77	OnDemandPack	Microsoft-Windows-Wifi-Client-Realtek-Rtl819xp-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Realtek-Rtl819xp-FOD- Package~31bf3856ad364e35~amd64~~10.0.22621.1

序号	类型	组件：原始	组件：安装“简体中文-中国”语言包后变化
78	OnDemandPack	Microsoft-Windows-Wifi-Client-Realtek-Rtl85n64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Realtek-Rtl85n64-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
79	OnDemandPack	Microsoft-Windows-Wifi-Client-Realtek-Rtwlane-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Realtek-Rtwlane-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
80	OnDemandPack	Microsoft-Windows-Wifi-Client-Realtek-Rtwlane01-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Realtek-Rtwlane01-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
81	OnDemandPack	Microsoft-Windows-Wifi-Client-Realtek-Rtwlane13-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1	Microsoft-Windows-Wifi-Client-Realtek-Rtwlane13-FOD-Package~31bf3856ad364e35~amd64~~10.0.22621.1
82	OnDemandPack	Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1	Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1
83	OnDemandPack		Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
84	OnDemandPack	Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.382	Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.382
85	OnDemandPack	Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1	Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1
86	OnDemandPack		Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1
87	OnDemandPack	Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1	Microsoft-Windows-WMIC-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1
88	OnDemandPack	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~en-US~10.0.22621.1
89	OnDemandPack		Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~zh-CN~10.0.22621.1
90	OnDemandPack	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.317	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~amd64~~10.0.22621.317
91	OnDemandPack	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~en-US~10.0.22621.1
92	OnDemandPack		Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~zh-CN~10.0.22621.1
93	OnDemandPack	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1	Microsoft-Windows-WordPad-FoD-Package~31bf3856ad364e35~wow64~~10.0.22621.1
94	OnDemandPack	OpenSSH-Client-Package~31bf3856ad364e35~amd64~~10.0.22621.1	OpenSSH-Client-Package~31bf3856ad364e35~amd64~~10.0.22621.1
95	Update	Package_for_DotNetRollup_481~31bf3856ad364e35~amd64~~10.0.9065.6	Package_for_DotNetRollup_481~31bf3856ad364e35~amd64~~10.0.9065.6
96	SecurityUpdate	Package_for_RollupFix~31bf3856ad364e35~amd64~~22621.382.1.13	Package_for_RollupFix~31bf3856ad364e35~amd64~~22621.382.1.13
97	SecurityUpdate	Package_for_ServicingStack_378~31bf3856ad364e35~amd64~~22621.378.1.0	Package_for_ServicingStack_378~31bf3856ad364e35~amd64~~22621.378.1.0