Ceci est un Script qui a été fait par mon collègue qui permet de créer des packages ZIP

```
<#
.Synopsis
Generate MUI Lang.zip from LanguagePack / FeatureOnDemand and LangOffice
folders

.DESCRIPTION
This script Build zip. file containing appropriate MUI source for the
master.

.NOTES
Name       : Build-MUIForMaster
Author     : Steven DURU
Version    : 1.0
DateCreated: 2019-11-26
DateUpdated:

.CHANGES
1.0 - 2019-11-26 - Creation script
#>

##############################################################################
#                          Variables declaration                             #
##############################################################################
# Return the directory of source files
If ($psISE) {
    $CurrentDir = Split-Path -Parent -Path $psISE.CurrentFile.FullPath
}
Else {
    if ($MyInvocation.MyCommand.CommandType -eq "ExternalScript") {
        $CurrentDir = Split-Path -Parent -Path
$MyInvocation.MyCommand.Definition
    }
    else {
        $CurrentDir = Split-Path -Parent -Path
([Environment]::GetCommandLineArgs()[0])
    }
}

$BuildPath = "$CurrentDir\Build"

##############################################################################
#                                  Script                                    #
##############################################################################

# Extract Content ISO and remove read only attributes.
Get-ChildItem $CurrentDir -recurse | Where-Object { $_.Extension -Match
"ISO" } | ForEach-Object {
```

```
    $Folder = "$($_.DirectoryName)\$($_.BaseName)"

    Try {
        Mount-DiskImage -ImagePath $($_.FullName) -PassThru | Out-Null
        $volume = Get-DiskImage -ImagePath $($_.FullName) | Get-Volume
        $source = $volume.DriveLetter + ":\*"
        $folder = mkdir $folder
        Write-Host "Extracting: $($_.Name) to '$Folder'..."
        copy-Item -Path $source -Destination $folder -Recurse -PassThru |
Set-ItemProperty -Name IsReadOnly -Value $False -ErrorAction
SilentlyContinue
        Write-Host "Exctraction completed !!"

        #Unmount the ISO
        Try {
            Write-Host "Unmounting the ISO..."
            Dismount-DiskImage -ImagePath $($_.FullName)
        }
        Catch {
            $_.Exception.Message
        }
    }
    Catch {
        $_.Exception.Message
    }
}


$csvData = @'
Arabic (saudi Arabia),ar-sa,ARAB
Bulgarian (Bulgaria),bg-bg
Chinese (PRC),zh-cn,HANS
Chinese (Hong Kong SAR),zh-tw,HANT
Croatian (Croatia),Hr-hr
Czech (Czech Republic),cs-cz
Danish (Denmark),da-dk
Dutch (Netherlands),nl-nl
Estonian (Estonia),et-ee
English (United Kingdom),en-gb,
Finnish (Finland),fi-fi
French (Canada),fr-ca,
French (France),fr-fr
German (Germany),de-de
Greek (Greece),el-gr
Hebrew (Israel),he-il,
Hungarian (Hungary),hu-hu
Italian (Italy),it-it
Japanese (Japan),ja-jp,Jpan
Korean (Korea),ko-kr,Kore
Latvian (Latvia),lv-lv
Lithuanian (Lithuania),lt-lt
```

```
Norwegian (Norway),nb-no
Polish (Poland),pl-pl
Portuguese (Brazil),pt-br
Portuguese (Portugal),pt-pt
Romanian (Romania),ro-ro
Russian (Russia),ru-ru
Serbian (Latin Serbia),sr-Latn-rs
Slovak (Slovakia),sk-sk
Slovenian (Slovenia),sl-si
Spanish (Mexico),es-mx
Spanish (Spain),es-es
Swedish (Sweden),sv-se
Thai (Thailand),th-th,Thai
Turkish (Turkey),tr-tr
Ukrainian (Ukraine),uk-ua
'@ | ConvertFrom-Csv  -Delimiter "," -Header
Language/region,CodeLang,FontsCode,LanguagePack,OfficeLang,Basic,Fonts,Handw
riting,OCR,Speech,TextToSpeech


$ArrayHeader =
@("Basic","Fonts","Handwriting","OCR","Speech","TextToSpeech")


Get-ChildItem "$CurrentDir\FeatureOnDemand" -recurse | Where-Object
{$_.Extension -eq ".cab" -and $_.Name -match "Microsoft-Windows-
LanguageFeatures" } | ForEach-Object {
    $FODName = $($_.Name)
    $FODFullPath = $($_.FullName)

    $csvData | ForEach-Object {
        # Check FeatureOnDemand match XX-XX
        If ( $FODName -match $($_.CodeLang) ) {

            If (!(Test-Path -Path
"$BuildPath\$($_.CodeLang)\FeaturePack\$($_.CodeLang)" )) {
                New-Item -Path "$BuildPath\$($_.CodeLang)\FeaturePack\" -
ItemType "Directory" -Name "$($_.CodeLang)" -Verbose | Out-Null
            }
            # Copy FOD.cab in Build\xx-xx\FeaturePack\xx-xx
            Copy-Item -Path $FODFullPath -Destination
"$BuildPath\$($_.CodeLang)\FeaturePack\$($_.CodeLang)" -Force

            # Add empty ini file.
            New-Item -Path "$BuildPath\$($_.CodeLang)" -ItemType "File" -
Name "$($_.CodeLang).ini" -Force -Verbose | Out-Null

            For ($i=0; $i -lt $ArrayHeader.Length; $i++) {
                If ( $FODName -Match $_.CodeLang -and $FODName -match
$($ArrayHeader[$i]) ) {
                    $_.$($ArrayHeader[$i]) = "True"
```

```
                }
            }
        }
        # Check FeatureOnDemand match Thain ( for Fonts )
        If ( $($_.FontsCode) -and ($FODName -match $($_.FontsCode)) ) {
            # Copy FeatureOnDemande match XX-XX
            If (!(Test-Path -Path
"$BuildPath\$($_.CodeLang)\FeaturePack\$($_.CodeLang)" )) {
                New-Item -Path "$BuildPath\$($_.CodeLang)\FeaturePack\" -
ItemType "Directory" -Name "$($_.CodeLang)" -Verbose | Out-Null
            }
            Copy-Item -Path $FODFullPath -Destination
"$BuildPath\$($_.CodeLang)\FeaturePack\$($_.CodeLang)" -Force


            For ($i=0; $i -lt $ArrayHeader.Length; $i++) {
                If ( $FODName -Match $_.FontsCode -and $FODName -Match
$($ArrayHeader[$i]) ) {
                    $_.$($ArrayHeader[$i]) = "True"
                }
            }
        }

    }
}

# Extract LanguagePack
Get-ChildItem "$CurrentDir\LanguagePack" -recurse | Where-Object
{$_.Extension -eq ".cab" -and $_.Name -match "Microsoft-Windows-Client-
Language-Pack_x64" } | ForEach-Object {
    $LPName = $($_.Name)
    $LPFullPath = $($_.FullName)

    $csvData | ForEach-Object {
        # Check FeatureOnDemand match XX-XX
        If ( $LPName -match $($_.CodeLang) ) {

            If (!(Test-Path -Path
"$BuildPath\$($_.CodeLang)\os\$($_.CodeLang)" )) {
                New-Item -Path "$BuildPath\$($_.CodeLang)\os\" -ItemType
"Directory" -Name "$($_.CodeLang)" -Verbose | Out-Null
            }

            # Copy LP.cab in Build\xx-xx\os\xx-xx
            Copy-Item -Path $LPFullPath -Destination
"$BuildPath\$($_.CodeLang)\os\$($_.CodeLang)" -Force

            $_.LanguagePack = "True"
        }
    }
}
```

```
# Create Office Structure
Get-ChildItem "$CurrentDir\OfficeLang\" -Directory | ForEach-Object {
    $ParentFolder = $($_.Name)
    $ParentFullPath = $($_.FullName)

    $csvData | ForEach-Object {
        If ( $ParentFolder -match $($_.CodeLang) ) {
            Copy-Item $ParentFullPath -Destination "$BuildPath" -Recurse -
Force

            $_.OfficeLang = "True"
        }
    }
}

# Rename and compress Each folder
Get-ChildItem "$BuildPath" -Directory | ForEach-Object {
    $LangFolder = $($_.Name)
    $LangFullPath = $($_.FullName)

    $csvData | ForEach-Object {
        If ($LangFolder -match $($_.CodeLang)) {
            Compress-Archive -Path "$LangFullPath\*" -CompressionLevel
Fastest -DestinationPath
"$BuildPath\$($_."Language/region")_MUI_1909_X64.zip"
        }
    }
}

$csvData | FT

$Datas = $csvData
# Import CSVdata to Excel

$Excel = New-Object -ComObject excel.application
$Excel.visible = $True
$Excel.DisplayAlerts = $false

$workbook = $Excel.workbooks.add()
$Sheet = $workbook.ActiveSheet

$Sheet.cells.item(1,1) = “Language/region”
$Sheet.cells.item(1,2) = “CodeLang”
$Sheet.cells.item(1,3) = “FontsCode”
$Sheet.cells.item(1,4) = “LanguagePack”
$Sheet.cells.item(1,5) = “OfficeLang”
$Sheet.cells.item(1,6) = “Basic”
$Sheet.cells.item(1,7) = “Fonts”
$Sheet.cells.item(1,8) = “Handwriting”
$Sheet.cells.item(1,9) = “OCR”
$Sheet.cells.item(1,10) = “Speech”
```

```
$Sheet.cells.item(1,11) = “TextToSpeech”
$Header = $Sheet.Range("a1","k1")
$Header.Font.Size = '16'

$i = 2
$Datas | ForEach-Object {
    $Sheet.cells.item($i,1) = $_.'Language/region'
    $Sheet.cells.item($i,2) = $_.CodeLang
    $Sheet.cells.item($i,3) = $_.FontsCode
    $Sheet.cells.item($i,4) = $_.LanguagePack
    $Sheet.cells.item($i,5) = $_.OfficeLang
    $Sheet.cells.item($i,6) = $_.Basic
    $Sheet.cells.item($i,7) = $_.Fonts
    $Sheet.cells.item($i,8) = $_.Handwriting
    $Sheet.cells.item($i,9) = $_.OCR
    $Sheet.cells.item($i,10) = $_.Speech
    $Sheet.cells.item($i,11) = $_.TextToSpeech
 $i++
} #end foreach process

# make it a table
$ListObject =
$Excel.ActiveSheet.ListObjects.add(1,$Excel.ActiveSheet.UsedRange,0,1)
$Excel.ActiveSheet.UsedRange.EntireColumn.AutoFit()
$ListObject.Name = "TableData"
$ListObject.TableStyle = "TableStyleMedium13"


Read-Host 'Hit any key to close Excel'

$workbook.saveas("$CurrentDir\Export.xls")
$Excel.Quit()
```