Albert Herd

Just a notepad to scribble my thoughts

POWERSHELL / TUTORIAL

Group files into folder structure by date using PowerShell

OCTOBER 7, 2018 ALBERT HERD LEAVE A COMMENT

Lately I was sorting out around 5000 files worth of pictures and videos taken out from a mobile phone, to upload them on a cloud drive. To make this task more manageable, I wanted to separate these files out, firstly by file type and then by date. This will make the folders far more smaller and therefore easier to upload these files to the cloud folder by folder.

Grouping such several files manually will be very tedious and error-prone. This task is a perfect candidate to be scripted out; this is exactly what I did. Since this may be helpful to other people going through such task, I decided to share this script.

All the script requires is a source path and a target path – anything else is handled by the script. It has several assumptions, such as it always moves, never deletes a file. Customising the script is easy and only requires basic programming knowledge.

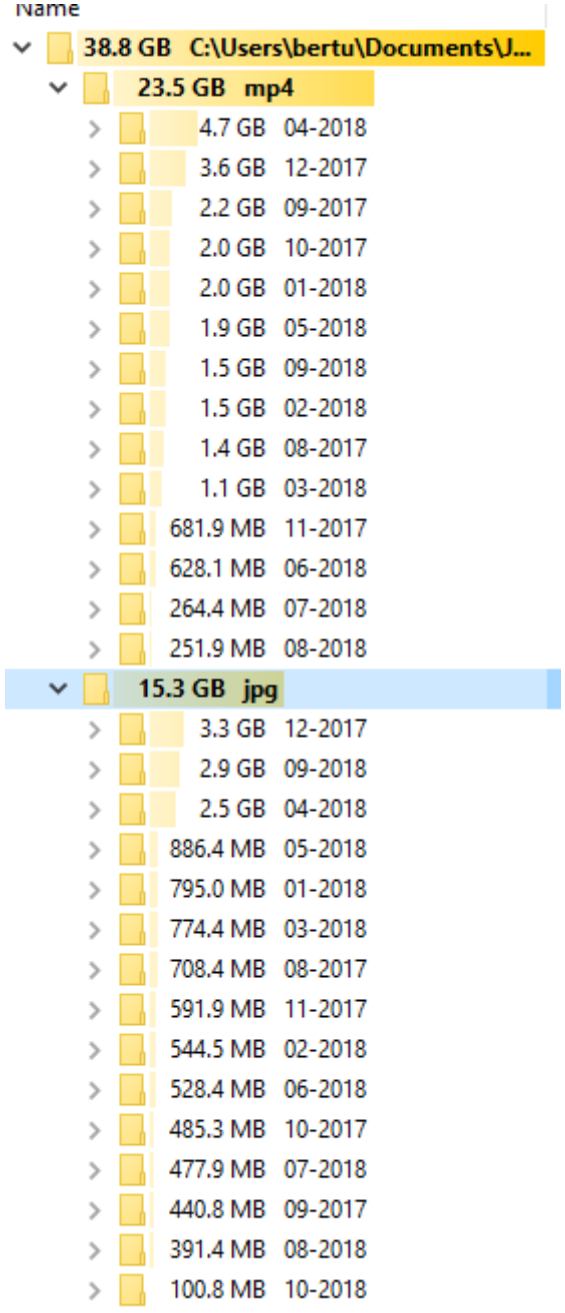
Check it out here or below: https://github.com/albertherd/GroupFilesPowershell/blob/master/GroupFiles.ps1 (https://github.com/albertherd/GroupFilesPowershell/blob/master/GroupFiles.ps1)

```
1  Param(
2      [Parameter(Mandatory=$true)]
3      [string]$sourcePath,
4      [Parameter(Mandatory=$true)]
5      [string]$targetPath
6  )
7
```

```
$global:fileTypeLookup = @{};
$folderDateTimeFormat = "MM-yyyy"

function Copy-FilesIntoFoldersByMonthAndType{
    param()

    $files = Get-ChildItem -Recurse -File -Path $sourcePath
    $filesProcessed = 0

    foreach($file in $files){
        $folder = Get-DirectoryForFile $file
        Copy-Item $file.FullName -Destination $folder

        $filesProcessed++
        Write-Progress -Activity "Grouping files" -Status "$($filesProcessed) out of $($files.Count) grou
    }
}
function Get-DirectoryForFile{
    param($file)

    $monthYearDirLookup = Get-FilePathDictionary $file
    $modifiedTimeMonthYearInternal = $file.LastWriteTime.ToString("MMyyyy")

    if($monthYearDirLookup.ContainsKey($modifiedTimeMonthYearInternal)){
        return $monthYearDirLookup[$modifiedTimeMonthYearInternal]
    }

    $extensionWithoutDot = $file.Extension.Substring(1, $file.Extension.Length - 1)
    $dateFolderFileName = $file.LastWriteTime.ToString($folderDateTimeFormat)
    $newPath = $targetPath + "\" + $extensionWithoutDot + "\" + $dateFolderFileName
    $path = New-Item -ItemType Directory $newPath -Force

    $monthYearDirLookup[$modifiedTimeMonthYearInternal] = $path.FullName
    return $path.FullName
}

function Get-FilePathDictionary{
    param($file)

    if($global:fileTypeLookup.ContainsKey($file.Extension)){
        return $global:fileTypeLookup[$file.Extension]
    }

```

```
51      $global:fileTypeLookup.Add($file.Extension, @{})
52      return $global:fileTypeLookup[$file.Extension]
53  }
54
55  Copy-FilesIntoFoldersByMonthAndType
```

In my case, it generated the below folder structure:



Far more manageable than one flat folder of 38.8 GB folder, for sure!

Until the next one.

FILE, FOLDER, GROUP, POWERSHELL, TUTORIAL