Betriebsdokumentation

Von Marek von Rogall  
Modul 122  
Abläufe mit einer Scriptsprache automatisieren

Funktion des Skripts Seite 2

Zielsystem Seite 2

Auslöser für das Skript Seite 2

Benötigte Berechtigungen & Clients Seite 3

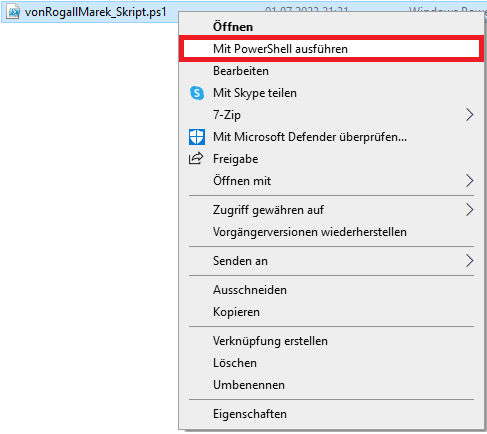
Verweis Entwicklerdokumentation Seite 3

Das Skript Seite 3-5

**Funktion des Skripts**

Das Skript dient dazu, Video-, Bild- und Audiodateien zu konvertieren. So können z.B. massenweise Videodateien in Audiodateien auf einmal konvertiert werden. Mithilfe der beigelegten Konfigurationsdatei können Sie den Ein- & Ausgabeordner, sowie den gewünschten Dateityp festlegen.  
  
**Zielsystem**

Das Zielsystem des Skripts ist Windows. Garantierte Kompatibilität wird mit Windows 10 Pro Version 22H (Build 19045.3086) gewährt.  
  
**Auslöser des Skripts**

Das Skript muss manuell ausgeführt werden, indem Sie auf das Skript rechts-klicken und anschliessend auf «Mit PowerShell ausführen» drücken.  


Alternativ können Sie auch die PowerShell Applikation auf Ihrem Rechner öffnen und zu dem Speicherort des Skripts navigieren. Anschliessend können Sie dieses ausführen.

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

**Benötigte Berechtigungen & Clients**

Um das Skript auszuführen sind ggf. zusätzliche Berechtigungen notwendig. Überprüfen Sie, dass Sie Ihre PowerShell ExecutionPolicy so eingestellt haben, dass das Ausführen von Skripts auf Ihrem Gerät erlaubt ist. Mehr über die ExecutionPolicy von PowerShell 5.1 erfahren sie [hier](https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_execution_policies?view=powershell-5.1).

Das Skript kann auf verschiedenen Clients gleichzeitig ausgeführt werden, benötigt aber nur ein Gerät um ausgeführt zu werden.

**Verweis Entwicklerdokumentation**

Mehr Informationen zum Skript finden Sie in der beigelegten Entwicklerdokumentation.

* vonRogallMarek\_Entwicklerdokumentation.docx

**Das Skript**

<# Skriptname: vonRogallMarek\_Skript.ps1

Autor: Marek von Rogall

Date: 2023-07-01

Version: 1.0

Beschreibung: Konvertiert Audio, Bild und Videodateien in ein gewünschtes Dateiformat.

#>

Clear-Host

$currentLocation = $PSScriptRoot

$global:errorCounter = 0

<#Funktion, welche überprüft ob ein Pfad existiert (und das Ergebnis in der Konsole ausgibt)#>

*Function* CheckPath($pathToCheck, [*string*]$name) {

    if ((Test-Path $pathToCheck) -eq $true) {

        Write-Host -NoNewLine "[OK] " -ForegroundColor green

        Write-Host -NoNewLine $name -ForegroundColor cyan

        Write-Host $pathToCheck -ForegroundColor white

    }

    else {

        $global:errorCounter++

        Write-Host -NoNewLine "[ERROR] " -ForegroundColor red

        Write-Host -NoNewLine $name -ForegroundColor cyan

        Write-Host -NoNewLine $pathToCheck -ForegroundColor white

        Write-Host " does not exist" -ForegroundColor cyan

    }

}

<#Überprüfen, ob sich die Konfigurationsdatei im Ordner des Skripts befindet.#>

$checkforconfig = Test-Path "$currentLocation\config.json"

if ($checkforconfig -eq $false) {

    Write-Host -NoNewLine "[ERROR] " -ForegroundColor red

    Write-Host -NoNewLine "Konfigurationsdatei: " -ForegroundColor cyan

    Write-Host "Es konnte keine Konfigurationsdatei gefunden werden. Bitte stelle sicher, dass sich config.json im selben Ordner wie das Skript befindet." -ForegroundColor white

}

else {

    Write-Host -NoNewLine "[OK] " -ForegroundColor green

    Write-Host -NoNewLine "Konfigurationsdatei: " -ForegroundColor cyan

    Write-Host "$currentLocation\config.json" -ForegroundColor white

}

<#Überprüfen, ob sich ffmpeg.exe im Ordner des Skripts befindet.#>

$checkforffmpeg = Test-Path "$currentLocation\ffmpeg.exe"

if ($checkforffmpeg -eq $false) {

    Write-Host -NoNewLine "[ERROR] " -ForegroundColor red

    Write-Host -NoNewLine "FFMPEG-Installation: " -ForegroundColor cyan

    Write-Host "Es konnte keine ffmpeg Installation gefunden werden. Bitte stelle sicher, dass sich ffmpeg.exe im selben Ordner wie das Skript befindet." -ForegroundColor white

}

else {

    Write-Host -NoNewLine "[OK] " -ForegroundColor green

    Write-Host -NoNewLine "FFMPEG-Installation: " -ForegroundColor cyan

    Write-Host "$currentLocation\ffmpeg.exe" -ForegroundColor white

}

<#Konfigurationsdatei und/oder ffmpeg.exe nicht vorhanden = Skript beenden, ansonsten weiter ausführen.#>

if ($checkforconfig -eq $false -or $checkforffmpeg -eq $false) {

    Write-Host "`nAufgrund diverser Fehler kann das Skript nicht ausgefuehrt werden." -ForegroundColor red

    Pause

    exit

}

<#Probieren Input- & Outputordner, sowie gewünschtes Dateiformat aus der Konfigurationsdatei auszulesen. Bei Fehlern wird das Skript beendet.#>

try {

    $jsonFilePath = "$currentLocation\config.json"

    $jsonContent = Get-Content -Raw -Path $jsonFilePath  | ConvertFrom-Json

    $inputfolder = $jsonContent.inputfolder

    $outputfolder = $jsonContent.outputfolder

    $fileformat = $jsonContent.fileformat

    Write-Host -NoNewLine "[OK] " -ForegroundColor green

    Write-Host "Einlesen der Konfigurationsdatei." -ForegroundColor yellow

}

catch {

    Write-Host -NoNewLine "[ERROR] " -ForegroundColor red

    Write-Host "Beim Einlesen der Konfigurationsdatei ist ein Fehler aufgetreten. Hinweis: Pfade muessen mit `"\\`" angegeben werden." -ForegroundColor yellow

    Pause

    exit

}

<#Überprüfen ob die eingelesenen Pfade wirklich existieren.#>

CheckPath "$inputfolder" "Inputfolder: "

CheckPath "$outputfolder" "Outputfolder: "

Write-Host -NoNewLine "[SET]" -ForegroundColor Yellow

Write-Host $fileformat -ForegroundColor white

<#Existiert mindestens eines der Pfade nicht, so wird das Skript beendet.#>

if ($errorCounter -gt 0) {

    Write-Host "`n$errorCounter Pfad(e) konnte(n) nicht gefunden werden. Daher kann das Skript nicht ausgefuert werden." -ForegroundColor red

    Pause

    exit

}

<#Dateipfad (inkl. Name) & Einzelnen Dateinamen von allen Dateien im Inputordner herausfinden und in ein Array speichern.#>

$filePathNames = @()

$fileNames = @()

$filePathNames += Get-ChildItem $inputfolder | % { $\_.FullName }

$fileNames += Get-ChildItem $inputfolder | % { $\_.BaseName }

<#Meldung welche Dateien konvertiert werden sollen.#>

Write-Host "$("`nDiese Dateien werden zu ")$fileformat$(" konvertiert:")" -ForegroundColor Red

ForEach ( $item in $filePathNames) {

    Write-Host $item

}

<#Fragen, ob diese wirklich konvertiert werden sollen.#>

Write-Host "Fortfahren? [Y/N]" -ForegroundColor Red

$answer = Read-Host

if ($answer -ne "Y") { exit }

Clear-Host

<#Generiere Output: $OutputOrdner\$Dateiname\$Dateiformat --> Z.B. "C:\users\Marek\Videos\video.mov"#>

$ffmpegOutputFilePathNameExtension = @()

ForEach ($file in $fileNames) {

    $ffmpegOutputFilePathNameExtension += "$outputfolder\$file$fileformat"

}

Write-Host "CONVERTING..." -ForegroundColor Red

<#Konvertiere InputFile1 zu OutputFile1, dann InputFile2 zu OutputFile2 usw...#>

For ($i = 0; $i -lt $ffmpegOutputFilePathNameExtension.Length; $i++) {

    Write-Host $filePathNames[$i]" to "$ffmpegOutputFilePathNameExtension[$i] -ForegroundColor Yellow

    & "$currentLocation\ffmpeg.exe" -i $filePathNames[$i] $ffmpegOutputFilePathNameExtension[$i]

}

Clear-Host

<#Meldung, dass die Konvertierung abgeschlossen wurde#>

Write-Host "Konvertierung abgeschlossen!" -ForegroundColor Cyan

Pause

exit