

O plánované spuštění Excelu a sešitu se postará Plánovač úloh ve Windows, případně skript navázaný na událost. Pro spuštění maker můžeme jít dvojitou cestou:

1) přímým voláním ze skriptu s pomocí známé metody Application.Run (ta podporuje i použití parametrů)

2a) prostým spuštěním sešitu, v němž je volání maker umístěno do událostní procedury Workbook_Open (historicky mírně odlišné makro Auto_Open)

2b) spuštěním sešitu s uvedením parametru /m, lépe řečeno /mNazevMakra ([Command-line switches for Microsoft Office products](#))

Příklady níže aplikují první ze způsobů.

Jako skript se nabízí BAT soubory, VBScript či PowerShell. Zpracovány jsou druhé dva.

VBScript

VBScript (viz také pojmy Windows Scripting Host, CScript) je tu už od nepaměti. Sloužil jako programovací jazyk pro Internet Explorer a je tu rovněž ve formě samospustitelných skriptů. Operační systém u souborů .vbs nezobrazoval příponu, a ty byly proto zneužívány pro nejrůznější havěť, hlavně v přílohách e-mailů. Přesto je tu tato podoba skriptů dostupná dodnes a hodí se pro naplánované úlohy a obsluhu sešitů Excelu (s makry).

Uvedený výpis postačí uložit (a spustit poklepnáním) v prostém textovém editoru s příponou .vbs.

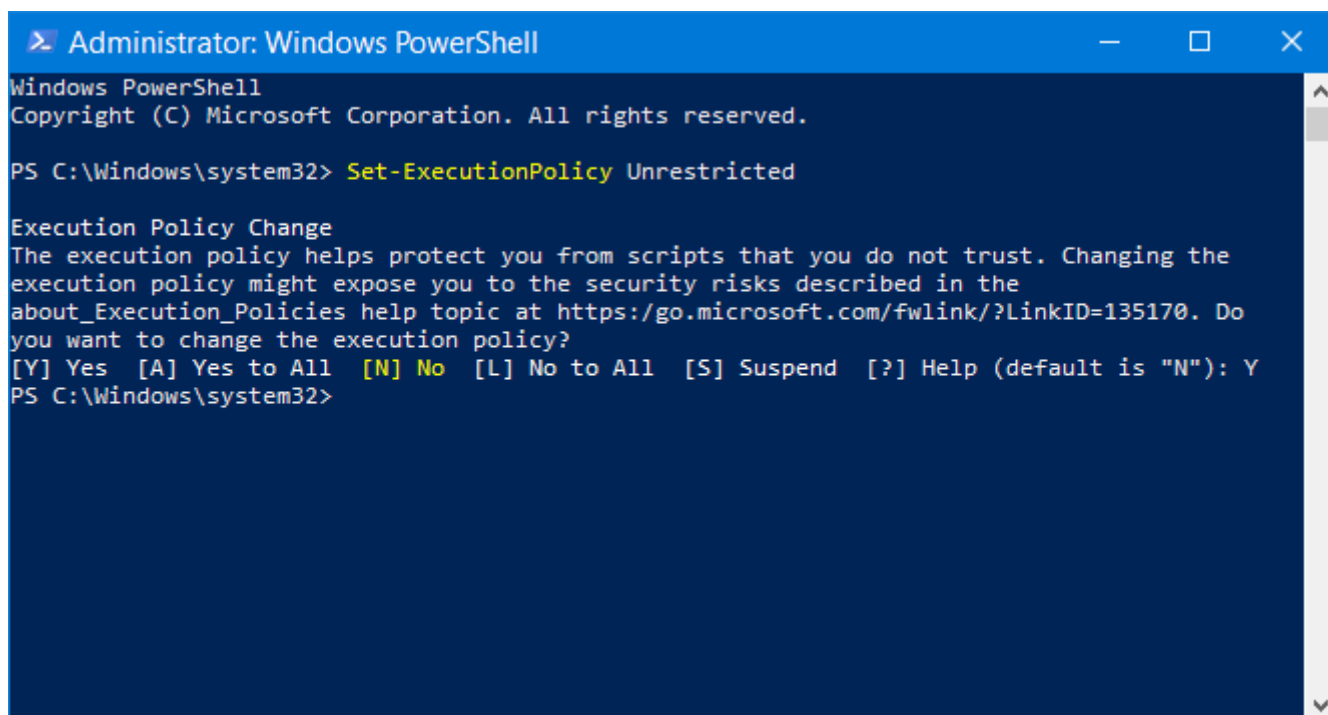
```
1 'VBScript
2
3 Set objShell = CreateObject("Wscript.Shell")
4 strCesta = objShell.CurrentDirectory
5
6 Set objExcel = CreateObject("Excel.Application")
7 Set objSesit = objExcel.Workbooks.Open(strCesta & "" & "sesit.xlsm")
8 Set objList = objSesit.Worksheets(1)
9
10 objExcel.DisplayAlerts = False
11
12 objExcel.Visible = True
13 objList.Cells(1, 1).Value = Now
14
15 objExcel.Run "MojeMakro"
16
17 objSesit.Save
18 objSesit.Close
19
20 objExcel.Quit
```

Kód je snadno čitelný i vzhledem k tomu, že VBScript a VBA mají společný základ z Visual Basicu 5 a 6. Vypnutí upozornění (DisplayAlerts) je nutné, jinak se běh zastaví na výzvě o uložení nového sešitu/listu s názvem List1. Ten vzniká pravděpodobně v rámci metody CreateObject, podobně jako když poklepete na zástupce aplikace Excel.

PowerShell

Abychom ve Windows mohli spouštět nepodepsané skripty PowerShellu, musíme pro tuto činnost nastavit patřičná práva. To znamená například spustit konzoli Windows PowerShell s právy správce a v ní následující kód:

Set-ExecutionPolicy Unrestricted



```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> Set-ExecutionPolicy Unrestricted

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the
execution policy might expose you to the security risks described in the
about_Execution_Policies help topic at https://go.microsoft.com/fwlink/?LinkID=135170. Do
you want to change the execution policy?
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): Y
PS C:\Windows\system32>
```

PowerShell – práva ke spuštění skriptů

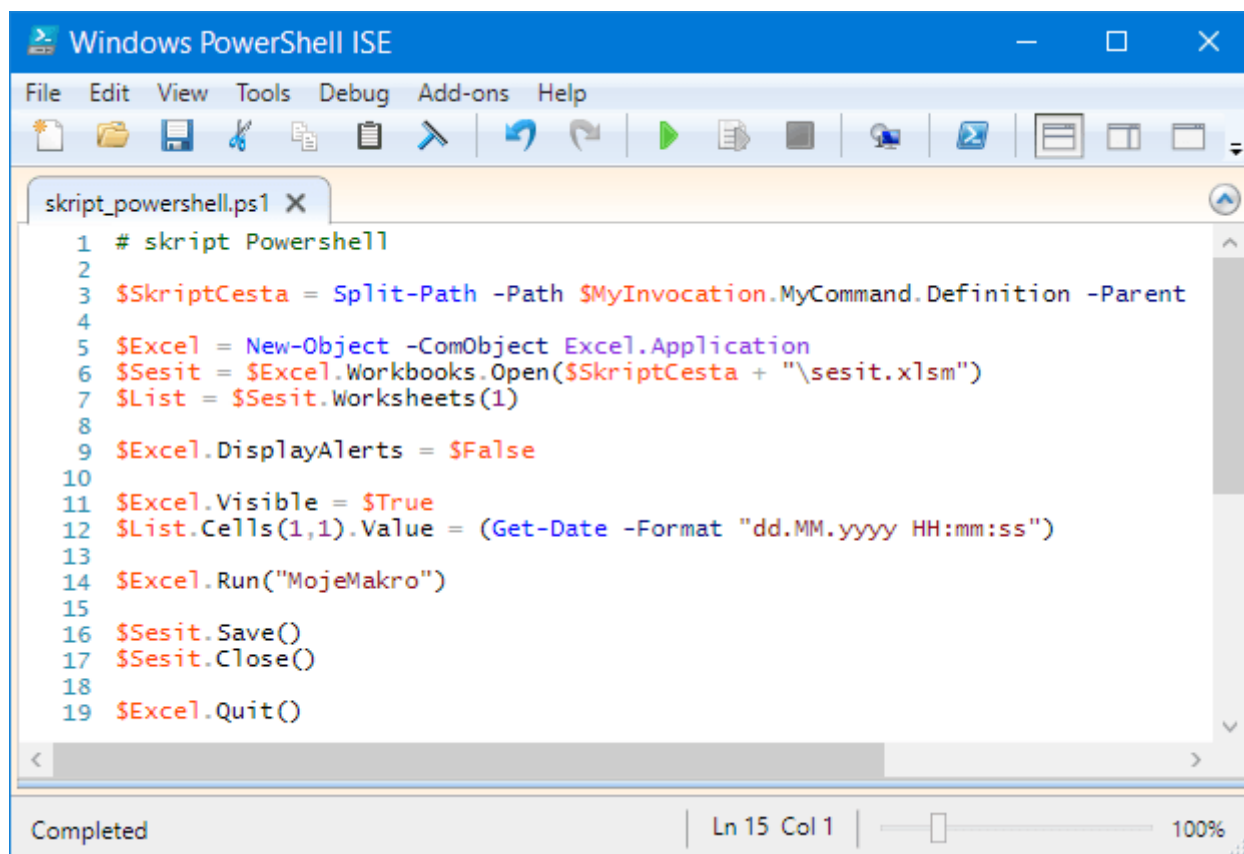
V zápisu níže se lze snadno orientovat, až tak výrazně se příkazy neliší od předchozích. Nicméně od mladšího nástroje, než je VBScript, bych čekal nějakou přidanou hodnotu. Ale nějak mě nadšení mívá. I když vyřešíme práva, pro mě nepřekonatelnou překážkou se stala taková hloupost, jako je zápis datumu do buňky Excelu. Ať jsem dělal, co dělal, datum se vždy vložil jako text. Takže jsem nakonec nechal původní syntaxi a nezabýval se dalšími pokusy (možná by pomohlo předformátování buňky a

zápisu data a času ve formě desetinného čísla). Při ukládání a zavírání sešitu mi chvíli trvalo, než jsem pochopil, že metody Save a Close vyžadují závorky (jinak se dočkáme tzv. OverloadDefinitions s hláškou typu void Save()). A stejně jako u VBScriptu, i zde platí, že v případě nevyplnění DisplayAlerts zůstane instance Excelu „viset“ a aplikace se neukončí.

Výpis níže postačí uložit jako obyčejný textový soubor s příponou .ps1. Nespouští se ovšem poklepáním, ale příkazem v konzoli PowerShellu.

```
1 # Powershell
2
3 $SkriptCesta = Split-Path -Path $MyInvocation.MyCommand.Definition -Parent
4
5 $Excel = New-Object -ComObject Excel.Application
6 $Sesit = $Excel.Workbooks.Open($SkriptCesta + "\sesit.xlsm")
7 $List = $Sesit.Worksheets(1)
8
9 $Excel.DisplayAlerts = $False
10
11 $Excel.Visible = $True
12 $List.Cells(1,1).Value = (Get-Date -Format "dd.MM.yyyy HH:mm:ss")
13
14 $Excel.Run("MojeMakro")
15
16 $Sesit.Save()
17 $Sesit.Close()
18
19 $Excel.Quit()
```

Pokud na uložený soubor .ps1 klepnete pravým tlačítkem myši, najdete v kontextové nápovědě volbu Upravit. Ta vás přesune do editoru Windows PowerShell ISE. Postupovat samozřejmě můžete i opačně. Nejprve spustíte editor a posléze vytváříte kód (viz View /Show Script Pane). Doporučuji použít také volbu Show Command Add-on.



```
Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
skript_powershell.ps1 X
1 # skript PowerShell
2
3 $SkriptCesta = Split-Path -Path $MyInvocation.MyCommand.Definition -Parent
4
5 $Excel = New-Object -ComObject Excel.Application
6 $Sesit = $Excel.Workbooks.Open($SkriptCesta + "\sesit.xlsm")
7 $List = $Sesit.Worksheets(1)
8
9 $Excel.DisplayAlerts = $False
10
11 $Excel.Visible = $True
12 $List.Cells(1,1).Value = (Get-Date -Format "dd.MM.yyyy HH:mm:ss")
13
14 $Excel.Run("MojeMakro")
15
16 $Sesit.Save()
17 $Sesit.Close()
18
19 $Excel.Quit()
Completed | Ln 15 Col 1 | 100%
```

PowerShell – editor a konzole

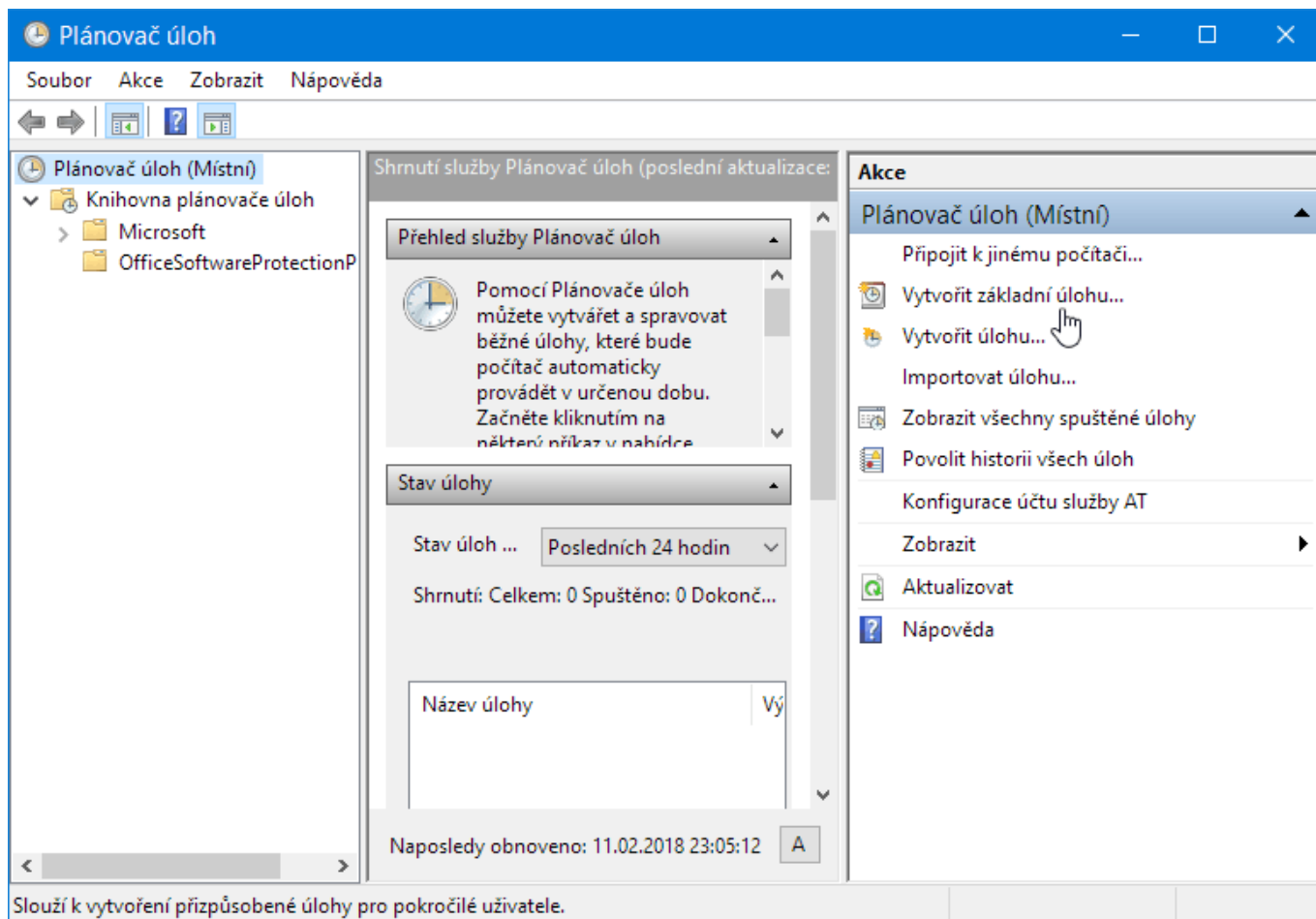
Plánovač úloh

Ať už jeden či druhý typ skriptu máte vytvořen a zbývá naplánovat jejich spuštění ve Windows. Následuje obrazový průvodce pro PowerShell. Pro samotné spuštění je použit příkaz

```
powershell -file „d:\skript_powershell.ps1“
```


V případě VBScriptu by se jednalo o příkaz

```
cscript skript_vbscript.vbs
```



Plánovač úloh - krok 1

Průvodce vytvořením základní úlohy

 Vytvořit základní úlohu

Vytvořit základní úlohu

Aktivační událost

Akce

Dokončit

Pomocí tohoto průvodce můžete rychle naplánovat běžnou úlohu. Pro podrobnější možnosti nebo nastavení (například více akcí nebo aktivační události úlohy) použijte příkaz Vytvořit úlohu v podokně Akce.


Název:

Popis:

< Zpět Další > Zrušit

Plánovač úloh - krok 2

Průvodce vytvořením základní úlohy ×

 Aktivační událost úlohy

Vytvořit základní úlohu

Aktivační událost

Akce

Dokončit


Kdy má být úloha spuštěna?

- Denně
- Týdně
- Měsíčně
- Jednou
- Při spuštění počítače
- Při přihlášení
- Při protokolování určité události

< Zpět Další > Zrušit

Plánovač úloh - krok 3

Průvodce vytvořením základní úlohy

 Denně

Vytvořit základní úlohu

Aktivační událost

Spustit: 12.02.2018 9:30:00 Synchronizace časových pásem

Denně

Opakování: 1 (dny)


Akce

Dokončit

< Zpět Další > Zrušit

Plánovač úloh - krok 4

Průvodce vytvořením základní úlohy

 Akce


Vytvořit základní úlohu

| | |
|-------------------|---|
| Aktivační událost | Jakou akci má úloha provést? |
| Denně | <input checked="" type="radio"/> Spustit program |
| Akce | <input type="radio"/> Odeslat e-mail (zastaralé) |
| Dokončit | <input type="radio"/> Zobrazit zprávu (zastaralé) |

< Zpět Další > Zrušit

Plánovač úloh - krok 5

Průvodce vytvořením základní úlohy

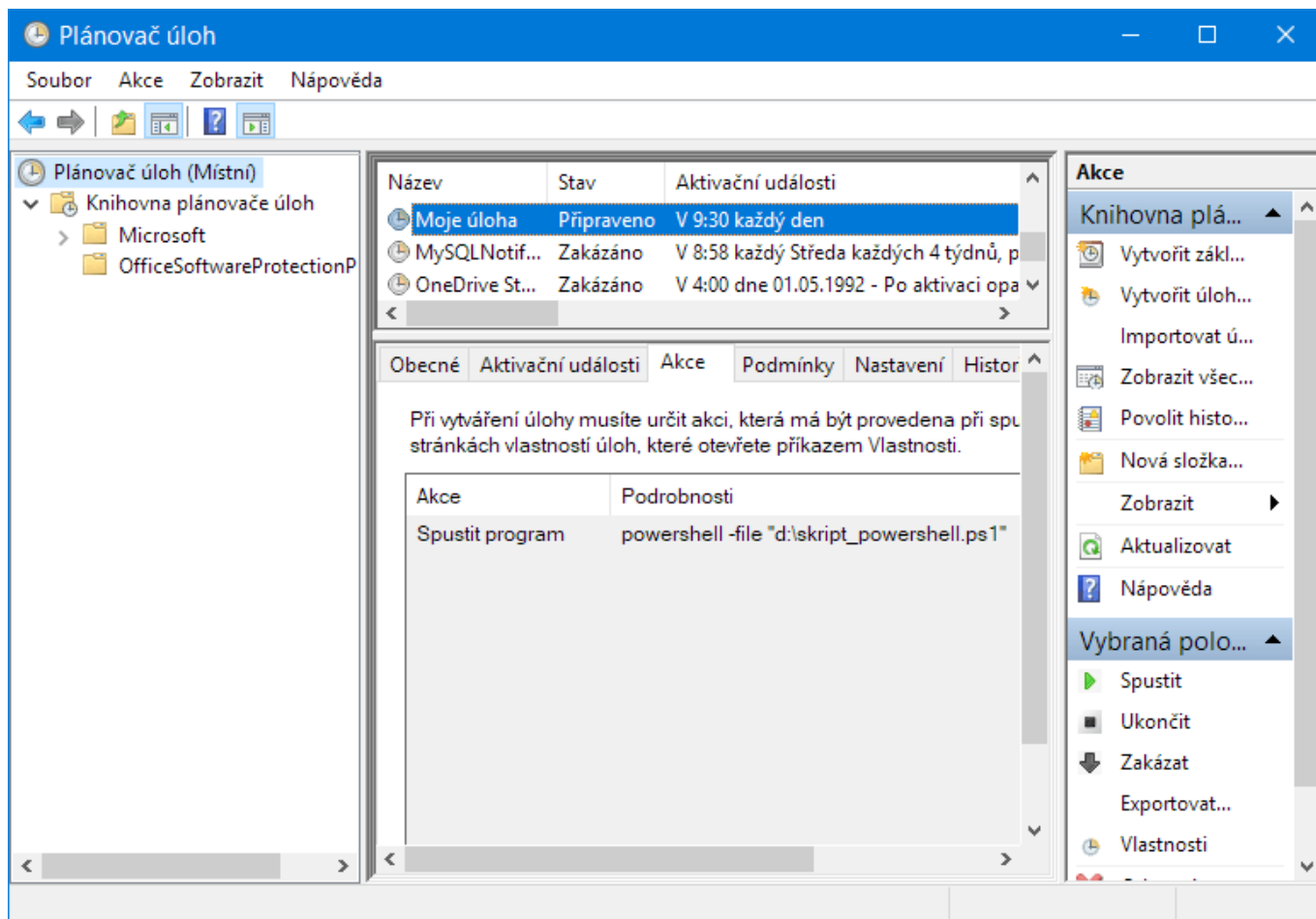
 Spustit program

Vytvořit základní úlohu

| | | | |
|------------------------|-------------------------------|--|---|
| Aktivační událost | Program či skript: | <input type="text" value="powershell"/> | <input type="button" value="Procházet..."/> |
| Denně | Přidat argumenty (volitelné): | <input type="text" value='-file "d:\skript_powershe'/> | |
| Akce | Spustit v (volitelné): | <input type="text"/> | |
| Spustit program | | | |
| Dokončit | | | |

< Zpět Další > Zrušit

Plánovač úloh - krok 6



Plánovač úloh - krok 7

Ke stažení

[excel_planovane_spusteni.zip](#)