Microsoft Windows

PowerShell

Arbeitsumgebung einrichten

Dr. Tobias Weltner
Microsoft MVP PowerShell

Microsoft®
PowerShell

PowerShell 7

- Als App installieren
  - Microsoft Store
  - nur als Preview (derzeit)
  - Auto-Update, kann jederzeit restarten
- Selbstverwaltet
  - versorgen Cmdlets mit Zusatzinformation

Microsoft®
PowerShell

PowerShell 7

- MSI downloaden und installieren:

```
# Download installation script
$uri = 'https://aka.ms/install-powershell.ps1'
$code = Invoke-RestMethod -Uri $uri

# Dynamically create PowerShell function
New-Item -Path function: -Name Install-PowerShell -Value $code

# Run PowerShel function and install the latest PowerShell
Install-PowerShell -UseMSI -Preview
```

- Startbefehl: pwsh

Microsoft® PowerShell

PowerShell 7 Erste Schritte

- Unterschiede
  - ersetzte Befehle (z.B. Get-WMIObject)
  - eingeschränkte Techniken (Workflows)
  - neue Operatoren (z.B. ??)
  - geänderte Profile ($profile)

  - Parallele Schleifen
  - Experimentelle Features

DEMO

Windows Terminal

- Workaround "Als Admin":

```
PS> Start-Process powershell -Verb runas -ArgumentList '-noprofile
-command wt'
PS> Start-Process cmd -Verb runas -ArgumentList '/c wt'

# am Rande:
start ms-windows-store:
```

Microsoft® PowerShell

Windows Terminal

- Als App installieren
  - Microsoft Store
  - "Preview" auswählen (mind. V 1.3)
  - Command Palette: STRG+SHIFT+P
- Einschränkungen
  - Als Admin ausführen
    - erfordert identisches Konto
    - keine MS-Accounts

Microsoft® PowerShell

Windows Terminal

- Workaround "Als Admin":

```
PS> Start-Process powershell -Verb runas -ArgumentList '-noprofile
-command wt'
PS> Start-Process cmd -Verb runas -ArgumentList '/c wt'

# am Rande:
start ms-windows-store:
```

- siehe auch: Open_Settings.ps1

Microsoft® PowerShell

Mit Windows Terminal arbeiten

- Parallel mit Konsolen arbeiten
- Persönliche Anpassungen
- Als Admin ausführen

DEMO

Github

- Austausch unserer Materialien
  - kostenloser Dienst
  - Teamarbeit an Quellcode
  - öffentlich oder privat
  - benötigt kostenloses Github-Konto

- Web-Frontend
  - https://github.com/TobiasPSP/ExpertClass2020

Microsoft® PowerShell

Github

- Grundsätzliche Idee
- Code über Frontend pflegen

- Zugriff aller Teilnehmer prüfen
- Grundlagen Push, Commit, Pull

DEMO

Github Desktop

- GUI für Github-Repos
  - sehr einfache Bedienung
  - keine kryptischen Befehle
  - wichtig für Austausch unserer Materialien

- Installation
  - https://desktop.github.com/

Microsoft® PowerShell

Github Desktop

- Lokales Werkzeug
- Einfacherer Umgang als Web-Interface

DEMO

VSCode

- Universaleditor
  - cross-platform, open-source
  - enthält keinen PowerShell-Host
  - funktioniert daher mit WPS und PS
  - "Editor Services" stellen Verbindung her

- Installation
  - https://code.visualstudio.com/
  - Palette: CTRL+SHIFT+P

Microsoft® PowerShell

VSCode

- Einrichten
  - CTRL+SHIFT+P
  - Open Settings

```
{
    "powershell.powerShellDefaultVersion": "PowerShell (x64)",
    "workbench.colorTheme": "PowerShell ISE",
    "files.defaultLanguage": "powershell",
    "editor.minimap.enabled": false,
    "workbench.activityBar.visible": true,
    "debug.openDebug": "neverOpen",
    "editor.tabCompletion": "on",
    "powershell.integratedConsole.focusConsoleOnExecute": false,

    "[powershell]": {
        "editor.renderWhitespace": "all",
        "editor.renderControlCharacters": true,
        "files.trimTrailingWhitespace": true,
        "files.encoding": "utf8bom",
        "files.autoGuessEncoding": true
    }
}
```

PowerShell in VSCode

- Bedienung und Konzept
- Command Palette

- Neue Skripte anlegen
- PowerShell Code interaktiv ausführen
- Skripte ausführen
- Code Completion und Script Analyzer

DEMO

VSCode

- Github-Unterstützung einrichten
  - View / Open View / Source Control
  - Git installieren (Link im Panel)
  - https://github.com/TobiasPSP/ExpertClass2020

  - Kurs-Repo klonen

Teamarbeit mit Github & VSCode

- Git installieren
- Repository klonen
- Eigene Dateien "committen"
- Im Team an PowerShell arbeiten

DEMO