PowerShell für Einsteiger

Grundlagen und Lernsituationen

Inhaltsübersicht

- Begrüßung/Vorstellung
- Impulsvortrag PowerShell
- PowerShell Grundlagen (Cmdlets, Hilfe zur Selbsthilfe)
- Pipeline, Prozesse und Dienste verwalten
- Benutzer und Gruppen verwalten, Benutzerprofile
- Dateisystem und NTFS-Rechte, Freigaben, Netzlaufwerke
- Netzwerkkonfiguration
- Serveraufgaben, Loganalyse, Webzugriff, Jobs
- Programmierung mit PowerShell (ps1-Skripte, Zugriff auf .NET-Objekte)
- Klassenarbeiten und Lernsituationen mit PowerShell erstellen und vorstellen

Verzeichnisse und Dateien

Grundlegende Manipulation von NTFS-Objekten

Navigation im Dateisystem

- Cmdlets der Familie *Location*, aber nicht *WinHomeLocation*
- Man kann wie in cmd (*pushd*, *popd*) auch Pfade mit einem Stack verwalten

Operation	Cmdlet	Aliase (PS, cmd, bash)
Verzeichnis betreten	`Set-Location`	`sl, cd, chdir`
Aktuelles Verzeichnis ausgeben	`Get-Location`	`gl, pwd`
Verzeichnis auf Stack legen	`Push-Location`	`pushd`
Verzeichnis vom Stack holen	`Pop-Location`	`popd`

Operationen im Dateisystem

- CRUD-Pattern (Create, Read, Update, Delete) für Dateien
- Einige Cmdlets der Familie *Item*

- Achtung:
  - `Invoke-Item` ruft eine Datei mit Standard-App auf
  - File IO mit Familie *Content* (*Get*, *Set*, *Add*, *Clear*)
  - Datei und Ordner werden mit demselben Cmdlet angelegt, aber Ordner erhalten Parameter *-ItemType Directory*

Operationen im Dateisystem

Operation	Cmdlet	Aliase (PS, cmd, bash)
Datei erstellen	`New-Item`	`ni`
Ordner erstellen	`New-Item … -ItemType Directory`	`ni`
Zugriff auf Dateisystem-Objekt (Handle)	`Get-Item`	`gi`
Dateiinhalt ausgeben	`Get-Content`	`gc,type,cat`
Datei/Ordner verschieben	`Move-Item`	`mi,move,mv`
Datei/Ordner umbenennen	`Rename-Item`	`rni,ren`
Datei/Ordner kopieren	`Copy-Item`	`cpi,copy,cp`
In Datei schreiben	`Set-Content`	
Datei/Ordner löschen	`Remove-Item`	`ri,del,rd,rm, rmdir,erase`

Dateisystem durchsuchen

- Auflisten und Suchen mit `Get-ChildItem`
  - entspricht im Wesentlichen *dir* bzw. *ls*
  - diese sind auch als Aliase verfügbar
  - PowerShell Alias ist *gci*
  - wichtige Parameter: *-Recurse*, *-Include*, *-Exclude*
  - *System.IO.FileInfo* hat 100(!) Properties und Methoden

Übungen PS51, PS52
Dateisystem

- CRUD-Operationen im Dateisystem mit PowerShell (PS51)
- Dateisystem durchsuchen, Wildcards verwenden (PS52)

Rechte im Dateisystem

Arbeiten mit NTFS-Zugriffsrechten

NTFS-Rechte
Grundlagen

- Beschreiben Zugriffsrechte an Dateisystemobjekten mit ACLs (Access Control Lists)
- Rechte werden an *Prinzipale* vergeben
- Es gibt *Zulassen- (engl. Allow)* und *Verweigern-* (engl. *Deny*) Rechte
- Windows kennt grundlegende und erweiterte Berechtigungen
- Rechtevergabe im Großen ist eher ein Serverthema

NTFS-Rechte
Rechtestufen bzw. Rechtepyramide

Stufe	Grundlegende Berechtigungen
FullAccess (FA)	Vollzugriff; Spezielle Berechtigungen
ModifyAccess (MA)	Ändern; Schreiben
ReadAccess (RA)	Lesen, Ausführen; Ordnerinhalt anzeigen; Lesen
NoAccess (NA)	--

- Jede Stufe enthält die Rechte der Stufe darunter
  - Teilmengenrelation fungiert als Ordnungsrelation
  - Die Rechtestufen werden durch Teilmengenbeziehung geordnet

☞ Welche Eigenschaften hat eine Ordnungsrelation?

NTFS-Rechte
Rechtestufen

- Definiert im .NET Namespace System.Security.AccessControl
  - Verfügbar im `enum FileSystemRights`
- Wichtige Felder im enum (passend zu Rechtestufen)

Stufe	Enum-Feld	Zahlenwert
RA	Read	131209
	ReadAndExecute	131241
MA	Modify	197055
FA	FullControl	2032127

- https://learn.microsoft.com/en-us/dotnet/api/system.security.accesscontrol.filesystemrights?source=recommendations&view=net-7.0

NTFS-Rechte
Ändern von Zugriffsrechten

- In cmd Änderung mit *icacls* (funktioniert auch in PowerShell)
- Cmdlets für Änderungen von NTFS-Rechten
  - `Get-Acl` bzw. `Set-Acl`
  - Wichtige Parameter `-Path`, `-AclObject`

- Grundlegendes Konzept:
  - Jede Berechtigung ist ein *FileSystemAccessRule*-Objekt
  - Kann hinzugefügt oder entfernt werden
  - Vererbungsverhalten wird über eine ACL-Methode gesteuert

NTFS-Rechte
The PowerShell Way

Schritt	Beschreibung	Cmdlets
1	ACL (NTFS-Rechte) des Objekts holen und	`$acl=Get-Acl -Path <Folder>`
2	Vererbung unterbr vererbte Rechte lö	
3	Objekt mit neuen Zugriffsberechtigungen erstellen	`$rule=New-Object System.Security.AccessControl.FileSystemAccessRule('<computer>\<account>',<Rechtestufe>,<Vererbung>,<Propagation>,<Regeltyp>)`
4	Regel mit neuen Zugriffsberechtigungen zur ACL hinzufügen	`$acl.AddAccessRule($acl)`
5	Aktualisierte ACL auf das Objekt anwenden	`Set-Acl -Path <Folder> -AclObject $acl`

Ja, die .NET-Klasse heißt wirklich *System.Security.AccessControl.FileSystemAccessRule*

NTFS-Rechte
Beispiel

- Ordner *Test* soll FA Rechte für Benutzer *anr* und RA Rechte für Gruppe *Benutzer* erhalten. (Der Ordner existiert bereits.)

```
PS C:\Users\anr\Downloads> $acl = Get-Acl -Path .\Test\
PS C:\Users\anr\Downloads> $acl.SetAccessRuleProtection($True, $False)
PS C:\Users\anr\Downloads> $rule=New-Object System.Security.AccessControl.FileSystemAccess
Rule("anr","FullControl","ContainerInherit, ObjectInherit","None","Allow")
PS C:\Users\anr\Downloads> $acl.AddAccessRule($rule)
PS C:\Users\anr\Downloads> $rule=New-Object System.Security.AccessControl.FileSystemAccess
Rule("Benutzer","ReadAndExecute","ContainerInherit, ObjectInherit", "None", "Allow")
PS C:\Users\anr\Downloads> $acl.AddAccessRule($rule)
PS C:\Users\anr\Downloads> Set-Acl -Path Test -AclObject $acl
```

NTFS-Rechte
Beispiel

- Im Ordner *Test* soll das RA Recht für die Gruppe *Benutzer* wieder entfernt werden.

```
PS C:\Users\anr\Downloads> $acl = Get-Acl -Path .\Test\
PS C:\Users\anr\Downloads> $rule=New-Object System.Security.AccessControl.FileSystemAccess
Rule("Benutzer","ReadAndExecute","ContainerInherit, ObjectInherit", "None", "Allow")
PS C:\Users\anr\Downloads> $acl.RemoveAccessRule($rule)
True
PS C:\Users\anr\Downloads> Set-Acl -Path .\Test\ -AclObject $acl
```

NTFS-Rechte
Hinweise

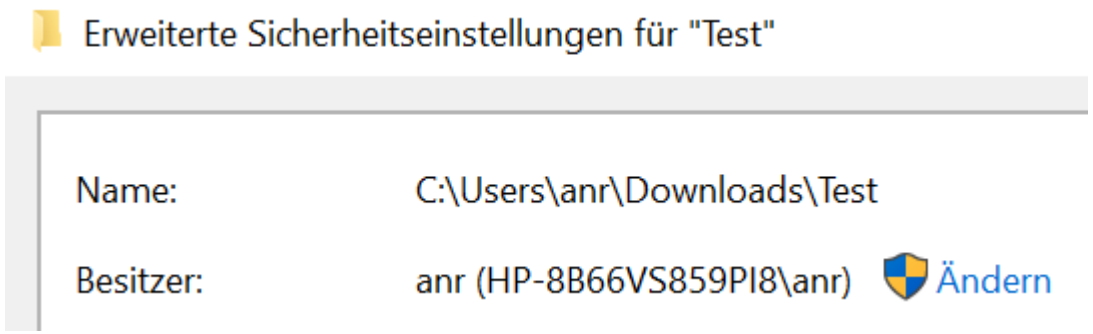
- Grundlegende OOP-Kenntnisse sind hilfreich für PowerShell-Methode
  - Objekte instantiieren `New-Object …`
  - Zugriff auf Properties `(Get-Acl <Folder>).Owner`
  - Arbeiten mit Methoden `$acl.AddAccessRule(…)`

- Im Zweifel mit SuS auch *icacls* verwenden (lassen)

- Nach Unterbrechung der Vererbung benötigt man eine Admin-PowerShell für `Set-Acl`, da auf dem Objekt keine expliziten Berechtigungen mehr vorhanden sind.

Besitz an einem Objekt übernehmen

- Jedes Dateisystemobjekt hat genau einen Besitzer
- Im Windows Explorer
  → Kontextmenü des Objekts aufrufen (Rechtsklick oder Button in Menüleiste)
  → Tab *Sicherheit*
  → Button *Erweitert*



Besitz an einem Objekt übernehmen

- Besitzerwechsel mit Button *Ändern* in GUI (Administratorrechte benötigt)

- In cmd mit `takeown /f <Ordner>`
  - funktioniert auch in PowerShell…
  - …aber natürlich gibt es auch Cmdlets

Besitz an einem Objekt übernehmen
The PowerShell Way

Schritt	Beschreibung	Cmdlets
1	ACL (NTFS-Rechte) des Objekts holen und speichern	`$acl=Get-Acl -Path <Folder>`
2	Benutzerobjekt für den neuen Besitzer erstellen	`$user=New-Object System.Security.Principal.Ntaccount('<computer>\<account>')`
3	Neuen Besitzer in ACL eintragen	`$acl.SetOwner($user)`
4	Aktualisierte ACL auf das Objekt anwenden	`Set-Acl -Path <Folder> -AclObject $acl`

NTFS Rechte

- PowerShell scheint erst einmal komplizierter als icacls, takeown, etc., aber
  - besser strukturiert, übersichtlicher, lesbarer
  - mehr Möglichkeiten durch Objektorientierung (viele Properties)
  - lässt sich gut in Skripte umwandeln

- Demo: Selbstgeschriebenes Cmdlet `Set-NtfsPermissions`

☞ Skripte, Programmierung und Funktionen kommen in Kapitel 7 ;-)

Übungen PS53
NTFS-Rechte

- NTFS-Rechte auslesen und vergeben
- Umgang mit ACLs für Dateisystemobjekte

Freigaben und Netzlaufwerke

Freigaben erstellen, Netzlaufwerke einbinden

Freigaben und Netzlaufwerke

- Ordner werden freigegeben, um den Zugriff von anderen Rechnern auf Ressourcen grundsätzlich zu ermöglichen.
  - es gibt einige Standardfreigaben *C$*, *Admin$*, …
  - spannender sind benutzerdefinierte Freigaben

- Netzlaufwerke sind Ressourcen, die eingebunden werden und sich wie ein lokales Laufwerk verhalten.
  - liegen häufig im LAN (dienen als Datenspeicher)
  - auch Ferneinbinden z.B. von WebDAV-Verzeichnissen möglich
  - Cloud-Speicher wie z.B. OneDrive, iCloud, Nextcloud,…

Freigaben und Netzlaufwerke
Informationsbeschaffung

- Auf dem System existierende Freigaben finden wir mit `Get-SmbShare`
  - *Smb* bedeutet dabei *Server Message Block*
  - Protokoll für Datei-, Druck- und Serverdienste in Netzwerken
  - Cmdlet entspricht grob Funktionalität von *net share*

```
PS C:\Users\anr> Get-SmbShare

Name    ScopeName Path        Description
----    --------- ----        -----------
ADMIN$  *         C:\windows  Remoteverwaltung
C$      *         C:\         Standardfreigabe
IPC$    *                     Remote-IPC
```

Freigaben und Netzlaufwerke
Informationsbeschaffung

- Auf dem System existierende Laufwerke finden wir mit `Get-PSDrive`
  - Laufwerke auf dem eigenen Rechner sind ein *PSDrive*,
  - die Umgebungsvariablen aber z.B. auch…
  - …zwei Hives der Registry sind ebenfalls als Laufwerke verfügbar

- Der Typ des Laufwerks wird in der Property *Provider* gespeichert
  - *FileSystem* sind gewöhnliche Dateisysteme (lokal oder remote)
  - *Environment* sind Umgebungsvariablen
  - *Registry* sind Hives oder Schlüssel aus der Windows Registry

Freigaben und Netzlaufwerke
Informationsbeschaffung

```
PS C:\Users\anr> Get-PSDrive

Name           Used (GB)     Free (GB) Provider      Root
----           ---------     --------- --------      ----
Alias                                  Alias
C                 175,94         61,53 FileSystem    C:\
Cert                                   Certificate   \
Env                                    Environment
Function                               Function
HKCU                                   Registry      HKEY_CURRENT_USER
HKLM                                   Registry      HKEY_LOCAL_MACHI…
Temp              175,94         61,53 FileSystem    C:\Users\anr\App…
Variable                               Variable
WSMan                                  WSMan
```

Freigaben und Netzlaufwerke
Informationsbeschaffung

- Mit dem System verbundene Netzlaufwerke finden wir mit `Get-SmbMapping`
  - *Smb* bedeutet dabei *Server Message Block*
  - Cmdlet entspricht grob der Funktionalität von *net use*

Freigaben und Netzlaufwerke
Vorgehen

- Das Einbinden von Netzlaufwerken erfolgt typischerweise in drei Schritten:

(1) Anlegen des Ordners im Dateisystem, NTFS Rechte vergeben

(2) Freigabe anlegen, Freigaberechte vergeben

(3) Freigegebenen Ordner als Netzlaufwerk einbinden

Freigaben und Netzlaufwerke
Beispiel

- Erstellen eines Tauschordners mit Schreibrechten für alle Benutzer
- ☞ Welcher Befehl legt einen neuen Ordner an ?

```
PS C:\Users\anr\Downloads> ni Tausch -ItemType Directory

    Directory: C:\Users\anr\Downloads

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d----          21.02.2023    18:42                Tausch
```

- Anlegen des Ordners im Dateisystem, NTFS Rechte vergeben
  - `Get-Acl`, ggf. Vererbung unterbrechen, Regeln definieren, `Set-Acl`

Freigaben und Netzlaufwerke
Beispiel

- Freigabe anlegen, Freigaberechte vergeben

```
PS C:\Users\anr\Downloads> New-SmbShare -Name "Tauschlaufwerk" -Path "C:\Users\anr\
Downloads\Tausch\" -ConcurrentUserLimit 10 -ChangeAccess "Jeder"

Name           ScopeName Path                          Description
----           --------- ----                          -----------
Tauschlaufwerk *         C:\Users\anr\Downloads\Tausch
```

- Parameter *-Path* enthält den absoluten Pfad zum Ordner
- Maximale Anzahl gleichzeitiger Zugriffe durch *-ConcurrentUserLimit* setzen
- Das Schreibrecht heißt bei den Freigaberechten *ChangeAccess*
  - Bei NTFS-Rechten ist es *Modify*
  - Alle Zugreifenden erfassen wir in der Freigabe mit *Jeder*, nicht mit *Benutzer*

Freigaben und Netzlaufwerke
Beispiel

- Freigegebenen Ordner als Netzlaufwerk einbinden

```
PS C:\Users\anr> New-SmbMapping -LocalPath "T:" -RemotePath "\\localhost\Tauschlaufwerk"

Status Local Path Remote Path
------ ---------- -----------
OK     T:         \\localhost\Tauschlaufwerk
```

- Der *LocalPath* muss kein Laufwerksbuchstabe sein; bietet sich aber an
- Der *RemotePath* wird im UNC Format angegeben

Freigaben und Netzlaufwerke
Beispiel

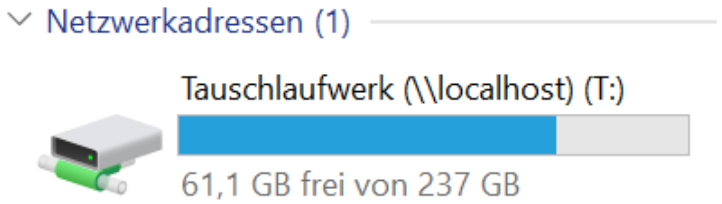
- In der PowerShell ist das Laufwerk unter T: sofort verfügbar

```
PS C:\Users\anr> T:
PS T:\>
```

- Zum Verwenden des Netzlaufwerks im Windows Explorer:
  - Explorer-Prozess neu starten

```
PS C:\Users\anr> Stop-Process -ProcessName "explorer"
```

  - Explorer-Fenster öffnen



Freigaben und Netzlaufwerke
Entfernen

- Zuerst Netzlaufwerk auflösen mit `Remove-SmbMapping`
- Dann Freigabe entfernen (Administrator PowerShell benötigt) mit `Remove-SmbShare`
- Gegebenfalls Dateisystemobjekt (Ordner) entfernen mit `Remove-Item` (ggf. Administrator PowerShell benötigt)

Freigaben und Netzlaufwerke
Hinweise

- Zum Anlegen und für die Freigaberechte wird ggf. eine Administratoren-PowerShell benötigt.
- Zum Einbinden eine normale PowerShell verwenden.
  - Sonst wird das Laufwerk ggf. nicht im Windows Explorer angezeigt.
- Da kein Output erzeugt wird, von dessen Objektstruktur wir profitieren, kann auch *net share*, *net use* benutzt werden.
- Bei WebDAV Laufwerken eher GUI oder *net*-Befehle nutzen
  - In PowerShell teilweise nicht realisierbar
  - Umgang mit URLs problematisch; UNC funktioniert meist aber

Übung PS54, PS55
Freigaben und Netzlaufwerke

- Freigaben (SMB-Shares) erstellen und verwalten (PS54)
- Netzlaufwerke erstellen, einbinden und verwalten (PS55)

Arbeiten mit der Windows Registry

Schlüssel auslesen und anlegen

Windows Registry

- Die Windows Registry ist die zentrale Verwaltung von Konfigurationsinformationen von Windows und installierten Programmen
- Besteht aus 5 sog. *hives* als Wurzeln von Schlüsselbäumen
  - HKEY_CLASSES_ROOT (HCR)
  - HKEY_CURRENT_USER (HKCU)
  - HKEY_LOCAL_MACHINE (HKLM)
  - HKEY_USERS (HKU)
  - HKEY_CURRENT_CONFIG (HCC)
- HKEY bedeutet *handle (to a) key*

Windows Registry

- Zugriff über PowerShell:
  - die Hives HKCU und HKLM sind als PSDrive standardmäßig verfügbar
  - erkennbar am Provider *Registry*
  - verhalten sich wie ein Dateisystem

```
PS C:\Users\anr\Downloads> Get-PSDrive -PSProvider Registry

Name           Used (GB)     Free (GB) Provider      Root
----           ---------     --------- --------      ----
HKCU                                   Registry      HKEY_CURRENT_USER
HKLM                                   Registry      HKEY_LOCAL_MACHINE
```

- Cmdlets speziell für Manipulation von Registry Schlüsseln
  - `Get-ItemProperty, Set-ItemProperty`

Windows Registry
Beispiel

- Abrufen einiger Basisinformationen zur aktuellen User Session

```
PS C:\Users\anr> Set-Location HKCU:
PS HKCU:\> Get-ItemProperty "Volatile Environment"

LOGONSERVER                : \\HP-8B66VS859PI8
USERDOMAIN                 : HP-8B66VS859PI8
USERNAME                   : anr
USERPROFILE                : C:\Users\anr
HOMEPATH                   : \Users\anr
HOMEDRIVE                  : C:
APPDATA                    : C:\Users\anr\AppData\Roaming
LOCALAPPDATA               : C:\Users\anr\AppData\Local
USERDOMAIN_ROAMINGPROFILE  : HP-8B66VS859PI8
PSPath                     : Microsoft.PowerShell.Core\Registry::HKEY_CURR
                             ENT_USER\Volatile Environment
PSParentPath               : Microsoft.PowerShell.Core\Registry::HKEY_CURR
                             ENT_USER
PSChildName                : Volatile Environment
PSDrive                    : HKCU
PSProvider                 : Microsoft.PowerShell.Core\Registry
```

Übung PS56
Windows Registry

- Registry als PSDrive verwenden
- Schlüssel und deren Werte anzeigen
- Schlüssel in der Registry anlegen