LUZERN

KANTON LUZERN

05_PS_Funktionen_I

Modul 122

Abläufe mit einer Scriptsprache automatisieren

Berufsbildungszentrum
Wirtschaft, Informatik und Technik
bbzw.lu.ch

Handlungsnotwendige Kenntnisse

HANOK	Kenntnisse
1.6	Kennt das Vorgehen zur Realisierung von Scripts in der Systemadministration.
2.1	Kennt grundlegende Funktionalitäten der eingesetzten Scriptsprache.

Themen

- Kommentare
- Variablen
- Funktionen in Powershell
  (Herdt-Buch: Kap 8.2 - 8.5, S. 114 – 117)

Kommentare

Verwenden sie «Kommentar», damit PS diese «nicht als Anweisungen» versteht.
Sie können Ein- oder Mehrzeilen-Kommentar verwenden:

```
# Eine Zeile auskommentiert
<#  Mehrzeilen-Kommentar
    hier einfügen  #>
```

```
1   # <Titel>
2   # <Autor>
3   # <Datum/Zeit>
4   # <Filename>
```

Variablen

- Variable definieren
  - PS:

```
$message="hi"
$value=5
```

  - Bash:

```
message="hi"
value=5
```

Variablen

- Variable einlesen
  - PS: `$username = Read-Host`
  - PS: `$username = Read-Host "Username: "`
  - Bash: `read username`
  - Bash: `read -p "Username: " username`

- Variable ausgeben

```
echo $message
echo $value
```

Variablen

- Umgebungsvariablen
  - Beinhalten Variablen zum Betriebssystem
  - Beispiel: PATH, HOME, etc. (unterscheiden sich in den verschiedenen OS)
  - PS: `echo $env:PATH`
  - Bash: `echo $PATH`

Variablen. Übung 5’

- Erstelle ein Skript, in welchem ein Name für einen Ordner angegeben werden kann. Der Ordner wird anschliessend erstellt.
  - Erweitere das Skript, mit der Angabe des Pfads.

- Dokumentiere das Skript mit einem Kommentarheader, welches grob die Funktion des Skripts beschreibt.

Funktionen

- In **Funktionen** werden Code-Schnippsel zusammengefasst
- **Funktionen** sind abgeschlossene Einheiten, die aus mindestens einem PowerShell-Befehl bestehen und einen Namen tragen.
- Selbst erstellte Funktionen sind nur für die momentane Sitzung verfügbar (Über Profilskript → Dauerhaft gültig).

- **Cmdlets** sind (kompilierte) Funktionen.

Theorie: Herdt-Buch: Kap 8.2 - 8.5, S. 114 - 117

Funktionen Syntax (PS)

```
function <Name der Funktion> ($Input1, $Input2, $Input3)
{
  echo $Input1
  <weiterer Anweisungsblock>
}

# Call function
<Name der Funktion> ($Input1, $Input2, $Input3)
```

Funktionen Syntax (Bash)

```
function <Name der Funktion> ()
{
  echo $1
  <weiterer Anweisungsblock>
}

# Call function
<Name der Funktion> $Input1, $Input2, $Input3
```

Auftrag 1 (Funktion ohne Parameter)

Erstelle eine Funktion mit dem Namen «message», welche die Nachricht «Message from function» ausgibt.

Prüfe den Code, indem die Funktion direkt im Skript ausgeführt wird.

Auftrag 2 (Funktion mit Parameter)

Kopiere den Code aus Aufgabe 1 und passe ihn so an, dass die zu auszugebende Nachricht der Funktion „message“ als Parameter übergeben werden kann.

Prüfe den Code, indem die Funktion direkt im Skript ausgeführt wird.

Aufgabe B1 (mAdd)

1. Schreiben sie eine Funktion mAdd, die den User auffordert nacheinander 2 Zahlen einzugeben und deren Summe als Ergebnis ausgibt

Aufgabe B2 (mAdd2: Funktion mit Parameter)

1. Schreiben sie eine Funktion mAdd2, die 2 Zahlen als Inputparameter hat (`$z1` und `$z2`). In der Funktion wird die Summe der beiden Inputparameter gebildet und ausgegeben.

Aufgabe B3 (mOwn)

- Schreiben sie eine Funktion mOwn, die 3 Zahlen als Inputparameter akzeptiert.
- Beschreiben sie selbst eine Rechenaufgabe mit diesen 3 Zahlen als Text
- Schreiben sie den entsprechenden Code dazu. Geben sie das Resultat als Ergebnis aus.
- Fügen sie den Text der ihre selbstgewählte Aufgabenstellung beschreibt im Skript als Kommentar ein.

Hausaufgaben

- Fragen Sie Ihre Lehrperson

Vielen Dank für Ihre Aufmerksamkeit!

KANTON LUZERN

Berufsbildungszentrum Wirtschaft, Informatik und Technik BBZW

www.bbzw.lu.ch