

Shell-Variablen (1)



BASH
THE BOURNE-AGAIN SHELL



Inhaltsverzeichnis

- [Beschreibung](#)
- [Variablen-Definition](#)
- [Variablen-Substitution](#)
- [Standard-Variablen](#)
- [Kürzel](#) ~ [für](#) \$HOME

Beschreibung

- Variablen sind Platzhalter für Werte.
- Man kann selbst Variablen definieren, d.h. einen Namen vergeben und einen Wert zuweisen.
- Bei der Verwendung von Variablen (Variablen-Substitution) wird der Wert der Variablen von der Shell an die Stelle des Variablennamens eingesetzt.

Variablen-Definition

Definition der Variablen `foo` mit dem Wert `bar`:

```
hermann@debian:~$ foo=bar
```

oder

```
hermann@debian:~$ foo="bar"
```

oder

```
hermann@debian:~$ foo='bar'
```

Regeln:

- Der Variablenname darf nur aus Buchstaben, Ziffern und Unterstrichen bestehen.
- Der Variablenname darf nicht mit einer Ziffer beginnen.
- Vor und nach dem Gleichheitszeichen (`=`) dürfen keine Leerzeichen stehen.
- Bei einfachen Werten darf der Variablenwert in Anführungszeichen stehen, muss aber nicht.
- Enthält der Variablenwert Leerzeichen (oder andere Shell-Sonderzeichen), dann muss er in Anführungszeichen stehen.

```
hermann@debian:~$ hello="Hallo Welt"
```

Variablen-Substitution

```
hermann@debian:~$ echo $foo  
bar
```

```
hermann@debian:~$ echo $hello  
Hallo Welt
```

```
hermann@debian:~$ echo "Der Wert von foo ist: $foo"  
Der Wert von foo ist: bar
```

```
hermann@debian:~$ echo 'Der Wert von foo ist: $foo'  
Der Wert von foo ist: $foo
```

Um den Wert einer Variablen zu verwenden (d.h. den Variablennamen durch den Wert zu ersetzen), muss dem Variablennamen ein Dollarzeichen (`$`) vorangestellt werden.

Um den Wert einer Variablen innerhalb einer Zeichenkette zu verwenden, muss die Zeichenkette in doppelte Anführungszeichen gesetzt werden.

Dies funktioniert nur mit doppelten Anführungszeichen, nicht mit einfachen Anführungszeichen.

MERKE: Variablen-Substitution (und Kommando-Substitution) ist **nur innerhalb von doppelten Anführungszeichen** möglich.

Standard-Variablen

Es gibt einige Standard-Variablen, die von der Shell automatisch belegt werden.

Die meisten dieser Standard-Variablen werden beim Login des Benutzers gesetzt. `$PWD` wird bei jedem Verzeichniswechsel mit `cd` neu gesetzt.

Die wichtigsten Standard-Variablen sind:

Variable	Wert
<code>\$USER</code>	Benutzername
<code>\$HOME</code>	Pfad zum Heimat-Verzeichnis des Benutzers
<code>\$PWD</code>	Pfad zum aktuellen Verzeichnis (gleiche Ausgabe wie das Kommando <code>pwd</code>)
<code>\$OLDPWD</code>	Pfad zum zuvor besuchten Verzeichnis
<code>\$PATH</code>	durch Doppelpunkte getrennte Verzeichnis-Pfade, in denen die Shell nach ausführbaren Dateien (Programmen oder Skripten) sucht.

\$USER und \$HOME

```
hermann@debian:~$ echo $USER
hermann
hermann@debian:~$ echo Ich bin $USER.
Ich bin hermann.
hermann@debian:~$ echo $HOME
/home/hermann
hermann@debian:~$ echo Hier wohne ich: $HOME
Hier wohne ich: /home/hermann
hermann@debian:~$ echo "Hier wohnt $USER:      $HOME"
Hier wohnt hermann:      /home/hermann
```

\$PWD und \$OLDPWD

```
hermann@debian:~$ cd Dokumente
hermann@debian:~/Dokumente$ pwd
/home/hermann/Dokumente
hermann@debian:~/Dokumente$ echo "previous workdir: $OLDPWD, current workdir: $PWD"
previous workdir: /home/hermann, current workdir: /home/hermann/Dokumente
```

```
hermann@debian:~/Dokumente$ cd ..
hermann@debian:~$ pwd
/home/hermann
hermann@debian:~$ echo "previous workdir: $OLDPWD, current workdir: $PWD"
previous workdir: /home/hermann/Dokumente, current workdir: /home/hermann
```

\$PATH

```
hermann@debian:~$ echo $PATH  
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
```

Kürzel `~` für `$HOME`

- Das Kürzel `~` steht für das Heimat-Verzeichnis des Benutzers.
- Es wird auch im Prompt verwendet.
- Es kann in der Shell häufig anstelle von `$HOME` verwendet werden.

```
hermann@debian:~$ echo ~  
/home/hermann
```

```
hermann@debian:~$ echo Hier wohne ich: $HOME  
Hier wohne ich: /home/hermann  
hermann@debian:~$ echo Hier wohne ich: ~  
Hier wohne ich: /home/hermann
```

- Anders als `$HOME` wird das Kürzel `~` nicht in doppelten Anführungszeichen substituiert.

```
hermann@debian:~$ echo "Hier wohne ich: $HOME"
Hier wohne ich: /home/hermann
hermann@debian:~$ echo "Hier wohne ich: ~"
Hier wohne ich: ~
```