# LU07a - Funktion definieren

ShellProg - Kapitel 6.1

Funktionen in Bash sind eine Möglichkeit, wiederverwendbaren Code zu schreiben. Sie ermöglichen es, eine Gruppe von Befehlen in einer benannten Einheit zu bündeln, die an beliebiger Stelle im Skript aufgerufen werden kann. Das verbessert die Lesbarkeit und Wartbarkeit von Shell-Skripten.

#### Definition einer Funktion Funktionen werden in Bash folgendermaßen definiert:

```
function name {
    # Befehle
}
```

Alternativ kannst du die `function`-Schlüsselwort weglassen:

```
name() {
    # Befehle
}
```

#### Aufrufen einer Funktion Eine Funktion wird durch ihren Namen aufgerufen, ohne Klammern:

```
name
```

#### Beispiel Hier ist ein einfaches Beispiel, das eine Funktion namens `greet` definiert:

```
#!/bin/bash

greet() {
    echo "Hallo, $1!"
}

greet "Welt"  # Aufruf der Funktion mit einem Argument
```

**Ausgabe:** </code> Hallo, Welt! </code>

#### Parameter in Funktionen Funktionen können Argumente wie Skripte selbst annehmen. Diese werden mit `$1`, `$2` usw. innerhalb der Funktion referenziert, wobei `$0` der Name des Skripts ist.

#### Rückgabewerte Funktionen können numerische Rückgabewerte verwenden, die mit dem Befehl `return` gesetzt werden:

```
summe() {
    return $(($1 + $2))
```

```
}

summe 3 5
echo $?  # Gibt 8 aus
```

Nicht-numerische Werte können über `echo` zurückgegeben und mit Command Substitution (`$(...)`) erfasst werden:

```
get_date() {
    echo $(date)
}

current_date=$(get_date)
echo "Aktuelles Datum: $current_date"
```

Mit Funktionen in Bash lassen sich also Skripte modularer und effizienter gestalten.

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
**https://wiki.bzz.ch/modul/m122/learningunits/lu07/funktionen?rev=1732599527**

Last update: **2024/11/26 06:38**