HOWTO zum Aufruf von Shell-Skripten

(C) 2006-2017 T.Birnthaler/H.Gottschalk <howtos(at)ostc.de>
OSTC Open Source Training and Consulting GmbH
http://www.ostc.de

$Id: shell-call-HOWTO.txt,v 1.13 2017/11/25 23:04:32 tsbirn Exp $

Dieses Dokument beschreibt die verschiedenen Verfahren zum Aufruf von
Shell-Skripten.

INHALTSVERZEICHNIS

1) Aufrufarten

---

1) Aufrufarten

Für den Aufruf eines Shell-Skriptes gibt es folgende Möglichkeiten, je nach
Anwendungszweck ist eine geeignete auszuwählen:

Aufruf	"PATH" durchsucht	"x"-Recht nötig	Sub-Shell	Neuer Proz.	Env Var vererbt	Sh Var vererbt	Aliase+Funkt. vererbt	Rückkehr zum Aufrufer
cmd.sh	Ja	Ja	Ja	Ja	J	-	--	Ja
./cmd.sh	--	Ja	Ja	Ja	J	-	--	Ja
/PFAD/ZU/cmd.sh	--	Ja	Ja	Ja	J	-	--	Ja
sh [OPT] cmd.sh	--	--	Ja	Ja	J	-	--	Ja
. cmd.sh	Ja	--	--	--	J	J	Ja	Ja
source cmd.sh	Ja	--	--	--	J	J	Ja	Ja
. ./cmd.sh	--	--	--	--	J	J	Ja	Ja
source ./cmd.sh	--	--	--	--	J	J	Ja	Ja
exec cmd.sh	Ja	Ja	--	--	J	-	--	--
exec ./cmd.sh	--	Ja	--	--	J	-	--	--

* Das Kommando "." heißt auch "source"- oder "dot"-Kommando. Es liest die
  angegebene Datei in der aktuellen Shell ein (eine Art include-Anweisung),
  startet also KEINE Sub-Shell. Der Suchpfad wird dabei wie bei einem normalen
  Kommando-Aufruf durchsucht.

* Der Name von Shell-Skripten ist beliebig wählbar. Per Konvention wird aber
  meist die Extension ".sh" angehängt (die dann beim auch Aufruf anzugeben ist!).

* Shell-Skripte müssen immer das Lese-Recht ("r"-Bit) gesetzt haben. Sofern
  sie als Kommando (ohne "sh", ".", "source", "exec" davor) aufgerufen werden
  sollen, müssen sie auch ausführbar sein ("x"-Bit) und in einem Verzeichnis
  im Suchpfad $PATH liegen.

* Der Suchpfad $PATH wird immer dann durchsucht, wenn nur ein Kommandoname
  angegeben wurde. Da inzwischen bei allen Benutzern (insbesondere der
  "root"!) aus Sicherheitsgründen das aktuelle Verzeichnis "." im Pfad fehlt,
  muss man Skripten im aktuellen Verzeichnis immer per "./SKRIPT" aufrufen.

* Von den Shell-Variablen werden nur die EXPORTIERTEN "Umgebungs-Variablen"
  an das aufgerufene Skript vererbt (Befehl "export"). "Shell-Variablen"
  sowie Aliase und Funktionen werden NICHT an Shell-Skripte vererbt.

* Wird eine Sub-Shell gestartet (Standardfall), dann können Variablen und
  sonstige Einstellungen aus dem gestarteten Skript NICHT an das aufrufende
  Skript zurückgegeben werden (die alten Werte in der Ausgangs-Shell bleiben
  erhalten). Dies wirkt auf den ersten Blick widersinning, ist allerdings
  unter dem Aspekt der Unabhängigkeit von Skripten ein gutes Konzept.

* Das Standardverhalten von DOS/Windows, alle Skripten im gleichen Umgebungs-
  bereich auszuführen, hat dort zu vielen Problemen geführt (Speicherüberlauf,
  Variablen überschrieben, Variablen unabsichtlich gesetzt, ...)

* Ein Aufruf über "exec" spart einen Prozeß ein, eine Rückkehr zum aufrufenden
  Prozeß ist allerdings nicht mehr möglich. Ist dann sinnvoll, wenn sehr viele
  Prozesse gestartet werden oder wenn der aufgerufenen Befehl der letzte im
  Skript ist.

* Shell-Skripte sollten kein SUID- oder SGID-Bit gesetzt haben, da sie immer das
  Lese-Recht gesetzt haben müssen und daher evtl. vorhandene Sicherheitslöcher

sehr einfach durch Lesen des Skriptes zu entdecken sind.

* Unter LINUX sind aus diesem Grund beide Bits für Shell-Skripten nicht setzbar.
  Bei jeder Änderung an ausführbaren Dateien werden zudem diese beiden Bits aus
  Sicherheitsgründen zurückgesetzt und müssen explizit wieder gesetzt werden.