HOWTO zum Bash-Eingabeprompt

(C) 2006-2024 T.Birnthaler/H.Gottschalk <howtos(at)ostc.de>
OSTC Open Source Training and Consulting GmbH
http://www.ostc.de

$Id: bashprompt-HOWTO.txt,v 1.16 2025/02/18 10:08:30 tsbirn Exp $

Dieses Dokument beschreibt die Konfiguration des Bash-Eingabeprompts.

INHALTSVERZEICHNIS

1) Primärer Prompt PS1
2) Sonderzeichen in Prompt-Variablen
3) Hinweise
4) Weitere Prompt-Variablen PS0, PS2, PS3, PS4

---

1) Primärer Prompt PS1

Die Variable "PS1" legt das Aussehen des Eingabeprompts (primärer Prompt, Eingabeaufforderung) fest, der immer dann ausgegeben wird, wenn eine interaktive Shell auf die Eingabe eines Befehls wartet. Standardwert ist:

```
PS1="\s-\v\$ "   # Shell-Name + "-" + Shell-Version + "$" oder "#" + Leerz.
```

Sie wird für alle Benutzer gemeinsam in "/etc/profile" (zuerst) oder für einzelne Benutzer in "~/.profile" (Login-Shell) bzw. "~/.bashrc" (interaktive Shell) gesetzt.

2) Sonderzeichen in Prompt-Variablen

Folgende Sonderzeichen in "PS1" (und den anderen Prompt-Variablen) setzen spezielle (variable) Komponenten in den Prompt ein:

Zeichen	Bedeutung	
\a	Klingeln	[alert]
\e	ESC-Zeichen	[escape]
\n	Zeilenumbruch	[newline]
\r	Wagenrücklauf	[carriage return]
\d	Datum in der Form "Sun Dec 24"	[date]
\D{FMT}	Datum formatiert gemäß "strftime(3)"-Format FMT	[Date]
\t	24h-Zeit in der Form "HH:MM:SS"	[time]
\T	12h-Zeit in der Form "HH:MM:SS AM/PM"	[Time]
\A	24h-Zeit in der Form "HH:MM"	
\@	12h-Zeit in der Form "HH:MM AM/PM"	
\h	Rechner-Name bis zum ersten "."	[host]
\H	Vollständiger Rechner-Name (FQHN)	[Host]
\j	Anzahl Hintergrundprozesse verwaltet von Shell	[jobs]
\l	Terminalname (ab letztem "/": "/dev/tty01" --> "tty01")	
\u	Benutzer-Name	[user]
\w	Vollständiges Arbeitsverzeichnis	[working directory]
\W	Arbeitsverzeichnis (Teil nach letztem "/")	[Working directory]
\s	Shell-Name (basename $0)	[shell]
\v	Bash-Version	[version]
\V	Bash-Version (inkl. Release und Patchlevel)	[Version]

```
  | \$      | "$"-Zeichen für normalen Benutzer, "#"-Zeichen für root         |
  | \#      | Befehlszeilennummer (in aktueller Sitzung)                      |
  | \!      | Befehlszeilennummer (über Sitzungen hinweg)                     |
  +---------+-----------------------------------------------------------------+
  | \[      | Beginn nicht darstellbare Zeichen (bei Kommandolänge ignoriert!) |
  | \]      | Ende nicht darstellbare Zeichen (bei Kommandolänge ignoriert!)   |
  | \\      | Backslash                                                       |
  +---------+-----------------------------------------------------------------+
```

Der Standardwert unter Linux ist für die root:

```
  PS1="\h:\w # "      # Host + Arbeitsverzeichnis + "#" + Leerzeichen
```

und für normale Benutzer:

```
  PS1="\u@\h:\w > "   # User + Host + Arbeitsverzeichnis + ">" + Leerzeichen
```

3) Hinweise

* Das Leerzeichen am Prompt-Ende ist wichtig, damit die Kommandoeingabe etwas abgesetzt vom Prompt beginnt.

* Der Prompt sollte nicht zu lang sein, da er sonst unübersichtlich wird und zu wenig Platz für die Eingabe der Kommandozeile übrig lässt.

* Ein INVERSER PROMPT kann durch das Kommando "tput" zur Ansteuerung von Terminal-Attributen erreicht werden. Dazu ist die Ausgabe des "tput"-Kommandos per Kommando-Substitution `...` oder $(...) in die Prompt-Definition einzufügen:

```
    PS1="\[`tput smso`\]\u@\h:\w \! >\[`tput rmso`\] "     # alt
    PS1="\[$(tput smso)\]\u@\h:\w \! >\[$(tput rmso)\] "   # modern
```

  "tput smso" schaltet den inversen Modus ein (set mode stand out), "tput rmso" schaltet ihn wieder ab (remove mode stand out).

  Damit die von "tput" erzeugten ANSI-Steuersequenzen nicht als Zeichen mitgezählt werden, sollten sie in "\[...\]" eingeschlossen werden (werden dann ignoriert). Vergisst man diese Klammern, dann lässt sich beim Editieren eines Kommandos der Cursor nicht korrekt mit den Cursortasten steuern.

* Analog lässt sich ein FARBIGER PROMPT erzeugen (z.B. "tput setaf 7" = weiße Schrift, "tput setab 4" = blauer Hintergrund, "tput sgr0" = normale Schrift):

```
    PS1="\[`tput setaf 7; tput setab 4`\]\u@\h:\w \!\[`tput sgr0`\] "
    PS1="\[$(tput setaf 7; tput setab 4)\]\u@\h:\w \!\[$(tput sgr0)\] "
```

* Soll nur der Name des aktuellen Verzeichnisses sichtbar sein (ohne Pfad davor), "\W" statt "\w" verwenden:

```
    PS1="\[`tput smso`\]\u@\h:\W \!\[`tput rmso`\] "
    PS1="\[$(tput smso)\]\u@\h:\W \!\[$(tput rmso)\] "
```

* Soll der absolute Pfad sichtbar sein, "\$(pwd)" statt "\w" verwendet (Kommando-Substitution):

```
    PS1="\[`tput smso`\]\u@\h:`pwd` \!\[`tput rmso`\] "
    PS1="\[$(tput smso)\]\u@\h:\$(pwd) \!\[$(tput rmso)\] "
```

* Der Wert der Variable "PROMPT_DIRTRIM" (ganze Zahl > 0) legt fest, wie viele Pfadkomponenten vom aktuellen Verzeichnis bei \w und \W übrig bleiben sollen.

4) Weitere Prompt-Variablen PS0, PS2, PS3, PS4

* Die Variable "PS0" wird zwischen dem Lesen des eingetippten Kommandos und seiner Ausführung ausgegeben. Standard ist:

  PS0=""

* Die Variable "PS2" legt das Bereitschaftzeichen fest, das immer dann ausgegeben wird, wenn Return gedrückt wird, die Shell das (mehrzeilige) Kommando aber noch nicht als abgeschlossen betrachtet (z.B. "for VAR in WERT1 WERT2 ..." <RETURN>). Standard ist:

  PS2="> "

  HINWEIS: Erscheint überhaupt kein Prompt nach der Eingabe von <RETURN>, so wartet das aufgerufene Kommando meist auf Eingabe von "stdin". Diese Eingabe kann mit <Strg-D> oder "." ABGESCHLOSSEN werden (Dateiende), mit <Strg-C> kann das Kommando ABGEBROCHEN werden (Cancel).

* Die Variable "PS3" steht für den Prompt beim "select"-Kommando. Standard ist:

  PS3="#? "

* Die Variable "PS4" wird vor Debug-Meldungen ausgegeben, die durch die Shell-Option "-x" [execute trace] ausgelöst werden. Standard ist:

  PS4="+"

* Der Inhalt der Variable "PROMPT_COMMAND" wird jedesmal als Kommando ausgeführt, bevor ein Eingabe-Prompt angezeigt wird. Die folgende Belegung gibt z.B. nach jedem eingegeben Kommando dessen Exit-Status aus.

  PROMPT_COMMAND="echo 'Exit-Status: $?'"