2.9 Skripte/Hilfsprogramme

2.9.1 Skript-Oberfläche mit `dialog`

Mit dem Programm `dialog` kann man einem Skript zu einer etwas gefälligeren Benutzeroberfläche verhelfen, die von weitem Ähnlichkeiten mit einem GUI hat.

2.9.1.1 Ausgabe mit `-msgbox` Das Programm `dialog` bietet zum Beispiel einen Ersatz für den Befehl `echo` an:

Terminal
```
schueler@debian964:~$ dialog --msgbox "Hello, World!" 0 0
```

Der erste Parameter bestimmt den auszugebenden Text, die anderen beiden die Höhe und die Breite des Textfensters (also die Y-Koordinate zuerst). Gibt man wie hier die Parameter mit 0 an, wird eine passende Größe gewählt.

2.9.1.2 Auswahl mit `-yesno` Mit dem folgenden Befehl kann man eine Frage stellen:

Terminal
```
schueler@debian964:~$ dialog --yesno "Haben Sie gut geschlafen?" 0 0
```

Mit der Tabulator-Taste kann man zwischen den beiden Auswahlknöpfen wechseln; mit der Return- oder der Enter-Taste schließt man die Wahl ab. Beim Auswählen von `Yes` wird der Wert 0 (=true) zurückgegeben, beim Auswählen von `No` ist der Rückgabewert 1.

2.9.1.3 Eingabe mit `-inputbox` Für die Eingabe gibt es die Option `inputbox`:

Terminal
```
schueler@debian964:~$ dialog --inputbox "Spannung U in Volt" 0 0
```

Jetzt kann man einen Wert eingeben; dieser Wert wird nach dem Abschicken mit `OK` auf die Fehlerausgabe (stderr, Kanal 2) ausgegeben. Für den Einsatz in einem Skript ist es oft einfacher, den Wert auf die Standardausgabe (stdout, Kanal 1) zu geben. Dazu dient die Option `--stdout`. Nun kann man das Ergebnis mit einer Kommandosubstitution in eine Variable packen:

Terminal
```
schueler@debian964:~$ U="$(dialog --stdout --inputbox \
>     "Spannung U in Volt" 0 0)"
schueler@debian964:~$ echo "$U"
50 # wenn man vorher 50 eingegeben hat
```

Wenn man die Eingabe nicht auf dem Bildschirm sehen soll, kann man statt `inputbox` die Option `passwortbox` benutzen.

2.9.1.4 Beispiel-Skript Hier ist ein Beispiel für ein mit `dialog` gestaltetes Skript:

```
1  #!/bin/bash
2
3  dialog --msgbox "Berechnung von P=U*I" 0 0
4  while [ "$?" -eq "0" ]
5  do
6     U="$(dialog --stdout --inputbox "Spannung U in Volt" 0 0)"
7     I="$(dialog --stdout --inputbox "Stromstaerke I in Ampere" 0 0)"
8     P=$((U*I))
9     dialog --msgbox "Leistung P = ${P} Watt" 0 0
10    dialog --yesno "Weitere Berechnung erwuenscht?" 0 0
11 done
```

2.9.1.5 Weitere Möglichkeiten

Genauso einfach ist es, ein Datum oder eine Uhrzeit auszuwählen:

Terminal

```
schueler@debian964:~$ DATUM="$(dialog --stdout --calendar \
>     Termin 0 0 24 12 1990)"
schueler@debian964:~$ UHRZEIT="$(dialog --stdout --timebox \
>     Uhrzeit 0 0 18 19 20)"
```

Die Eingabe von Datei- und Verzeichnisnamen ist für den Skript-Programmierer nicht schwer, die Anwendung für GUI-verwöhnte Nutzer schon:

Terminal

```
schueler@debian964:~$ DATEINAME=$(dialog --stdout --fselect "" 20 0 )
```

Wieder kann man mit der Tabulator-Taste zwischen den Fenstern springen und mit der Return- oder der Enter-Taste das Ergebnis abschicken. Mit den Pfeiltasten kann man im Verzeichnis- und im Dateifenster nach unten und nach oben laufen. Die Space-Taste kopiert den Inhalt an der aktuellen Position des Verzeichnis- oder Dateifensters in die Ergebniszeile (unten). Wenn man ein Verzeichnis in der Ergebniszeile stehen hat, kann man durch Anfügen eines Schrägstrichs in dieses Verzeichnis wechseln. Auf diese Art kann man mit etwas Übung jedes Objekt im Dateisystem auswählen.

2.9.2 GUI-Skriptoberflächen

Für einen Endanwender ist `dialog` eventuell zu spröde. In diesem Fall bietet sich das Programm `zenity` an (oder weitere Programme wie `xdialog` und `kdialog`). `zenity` ist die GUI-Ausführung von `dialog`.

2.9.2.1 Ausgabe mit `-info`

Der `echo`-Befehl sieht dann so aus:

Terminal

```
schueler@debian964:~$ zenity --info --text "Hello, World!"
```

Positiv ist, dass `zenity` keine Fenstergröße braucht; der einzige notwendige Parameter ist `--info`, der die Art des Fensters bestimmt. Negativ ist, dass die Namen der Optionen meistens nicht mit den Namen von `dialog` übereinstimmen.

Übrigens kann man in den Text ein HTML-ähnliches Markup einbauen (Pango heißt die Sprache, sie ist eine Teilmenge von HTML):

Terminal

```
schueler@debian964:~$ zenity --info --text \
>     "a<sup>2</sup>+b<sup>2</sup>=c<sup>2</sup>"
```

2.9.2.2 Auswahl mit `-question`

Die Auswahl zwischen zwei Alternativen funktioniert wie bei `dialog`:

Terminal

```
schueler@debian964:~$ zenity --question --text \
>     "Haben Sie gut geschlafen?"
```

2.9.2.3 Eingabe mit `-inputbox`

Bei der Eingabe von Text hat man es mit `zenity` leichter, weil es keine Konkurrenz zwischen der Nutzerausgabe des Programms und der Ergebnisausgabe gibt. Das Ergebnis wird hier immer auf Kanal 1 (stdout) ausgegeben:

Terminal

```
schueler@debian964:~$ U="$(zenity --entry --text "Spg. U in Volt")"
```

2.9.2.4 Weitere Möglichkeiten Auch hier gibt es wieder viele Möglichkeiten: Kalender, Farbauswahl, Menüs, Fortschrittsbalken usw. Die Eingabe eines Dateinamens ist für den Benutzer jedenfalls einfacher als bei `dialog`:

Terminal

```
schueler@debian964:~$ DATEINAME="$(zenity --file-selection)"
```

2.9.2.5 Beispiel-Skript Und hier ist ein Beispiel für die Anwendung von `zenity`:

```
1  #!/bin/bash
2  #
3  # Beispiele fuer zenity
4  #
5  TMPDATEI="/tmp/tmp$$.txt"
6  chmod 600 "$TMPDATEI"
7  DATEI="$(zenity --file-selection --title="Welche Datei editieren?")"
8
9  if [ -z "$DATEI" ]
10 then
11    zenity --info --text="Ende."
12 else
13    zenity --text-info --title="Datei editieren" \
14           --filename="$DATEI" \
15           --ok-label="Speichern" \
16           --cancel-label="Nix tun" \
17           --editable \
18           --checkbox="Ich weiss, was ich da tue. Ich tue es trotzdem." \
19           >"$TMPDATEI"
20
21    if [ "$?" == "0" ]
22    then
23       mv "$TMPDATEI" "$DATEI"
24    fi
25 fi
```