Werkzeugkasten (Editoren)

- Kate: http://de.wikipedia.org/wiki/Kate_%28KDE%29
- Bluefish: http://de.wikipedia.org/wiki/Bluefish
- Eclipse mit Plugin: http://de.wikipedia.org/wiki/Eclipse_%28IDE%29 , http://sourceforge.net/projects/shelled/

Hello BASH

Neue Textdatei "hallo.sh" mit folgendem Text erstellen:

```
#!/bin/bash
echo Hallo BASH!
```

Ausführbar machen:

```
chmod +x hallo.sh
```

Ausführen (in aktuellem Verzeichnis):

```
./hallo.sh
```

Variablen

Neue Textdatei "variablen.sh" mit folgendem Text erstellen:

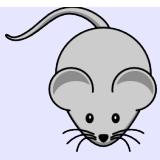
```
#!/bin/bash
zahl=10                          # Einfache Zahl
text=' hunde und katze'          # Ein Text
textzahl="$zahl$text"            # Text mit Variablenersetzung
textverlaengert="${text}n"       # Variablenersetzung + Text anhängen
genauso='$text$zahl'             # Text unverändert

echo "zahl:$zahl text:$text textzahl:$textzahl textverlaengert:
$textverlaengert genauso:$genauso"
```

Ausführen (in aktuellem Verzeichnis):

```
bash variablen.sh
```

Besondere Variablen



```
$?      # Rückgabewert von Programmen
$#      # Anzahl der Parameter bei Funktions- oder Skriptaufruf
$@      # Alle Parameter
$1      # 1. Parameter
$2      # 2. Parameter
...
```

Neue Textdatei "parameter.sh" mit folgendem Text erstellen:

```
#!/bin/bash
echo "Anzahl:$# 1.P:$1 2.P.:$2 3.P:$3 alle P.$@"
```

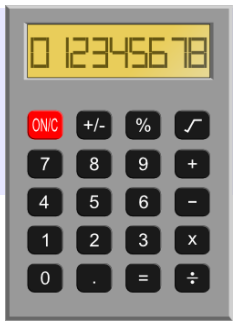
Ausführen (in aktuellem Verzeichnis):

```
bash parameter.sh hund katze maus
```

Vergleichen und Berechnen

```
[ $a -eq $b ]     # $a gleich (equal) $b?
[ $a -gt $b ]     # $a größer (greater then) $b?
[ $a -lt $b ]     # $a kleiner (less then) $b?
[ $a = $b ]       # Text $a identisch zu Text $b?
[ $a != $b ]      # Text $a nicht identisch zu Text $b?
[ $a ]            # $a auf etwas nicht-leeres gesetzt (Text oder Zahl)
```

Ergebnis in Variable $?: 0 = wahr, 1 = falsch



```
c=$[ $a + $b ]    # $a plus $b ist $c
c=$[ $a - $b ]    # $a minus $b ist $c
c=$[ $a * $b ]    # $a mal $b ist $c
c=$[ $a / $b ]    # $a geteilt durch $b ist $c
c=$[ $a % $b ]    # Rest  von $a geteilt durch $b ist $c
```

Weitere Verwendungsmöglichkeiten: **man test**

Verzweigungen

Neue Textdatei "verzweigungen.sh" mit folgendem Text erstellen:

```
#!/bin/bash
echo "Bitte etwas eingeben:"
read zahl  # Zahl von Tastatur einlesen und in $zahl speichern

if [ $zahl -gt 5 ]
then
      echo "Größer als 5"
else
    echo "Nicht größer als 5"
fi

case $zahl in
    1)
        echo "Zahl ist genau 1"
        exit;;
    10)
        echo "Zahl ist genau 10"
        exit;;
    'text')
        echo "Keine Zahl, sondern \"text\""
        exit;;
```

```
    *)
        echo "Zahl ist was anderes"
esac
```

Ausführen (in aktuellem Verzeichnis):

```
bash verzweigungen.sh
```

Umleitungen

```
>  # Standard-Ausgabe in Datei schreiben (Datei ggf. überschreiben)
2> # Fehler-Ausgabe in Datei schreiben (Datei ggf. überschreiben)
>> # Standard-Ausgabe in Datei schreiben (ggf. an vorhandene Datei
anhängen)
```


```
|       # Standard-Ausgabe in anderes Programm umlenken
2>&1 |  # Standard- und Fehler-Ausgabe in anderes Programm umlenken
```

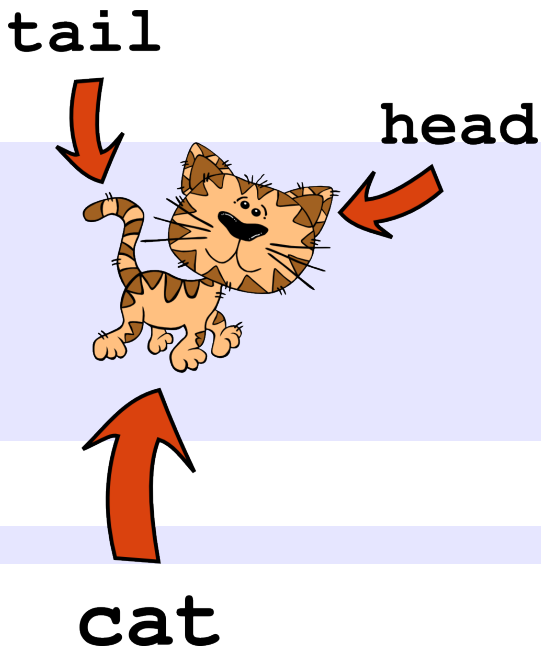
Neue Textdatei "umleitungen.sh" mit folgendem Text erstellen:

```
#!/bin/bash
seq 1 100 > /tmp/zahlen
seq 101 200 >> /tmp/zahlen
```

Ausführen (in aktuellem Verzeichnis):

```
bash umleitungen.sh
```

cat, head, tail: Dateien ausgeben



Neue Textdatei "katze.sh" mit folgendem Text erstellen:

```
#!/bin/bash

# Datei komplett ausgeben
cat /tmp/zahlen
# Die ersten 5 Zeilen von Datei ausgeben
head -5 /tmp/zahlen
# Die letzten 10 Zeilen von Datei ausgeben
tail -10 /tmp/zahlen
```

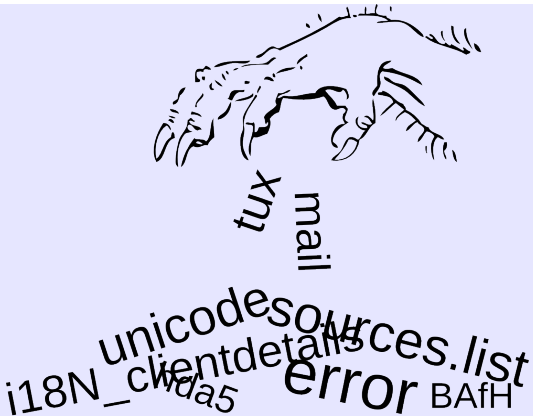
Ausführen (in aktuellem Verzeichnis):

```
bash katze.sh
```

(In) Dateien suchen

Neue Textdatei "suchen.sh" mit folgendem Text erstellen:

```
#!/bin/bash
echo "Alle Dateien unter /tmp/ auflisten:"
find /tmp/
echo 'Dateinamen finden, die "zahl" enthalten:'
find /tmp/ | grep zahl
echo 'Zeilen finden, die mit 10 aufhören:'
grep 10$ /tmp/zahlen
echo 'Zeilen finden, die genau 10 enthalten:'
grep ^10$ /tmp/zahlen
echo 'Zeilen finden, die mit 10 beginnen:'
grep ^10 /tmp/zahlen
```



Ausführen (in aktuellem Verzeichnis):

```
bash suchen.sh
```

Schleifen

Neue Textdatei "schleifen.sh" mit folgendem Text erstellen:

```
#!/bin/bash
echo "Mit for:"
for i in $(seq 1 10)
do
    echo $i
done

echo "Mit while:"
i=1
while [ $i -lt 11 ]
do
    echo $i
    i=$[ $i + 1 ]
done
```

Ausführen (in aktuellem Verzeichnis):

```
bash schleifen.sh
```

Texte bearbeiten

Neue Textdatei "MausInDaHaus.sh" mit folgendem Text erstellen:

```
#!/bin/bash
echo 'Eine Katze ist in der Küche, eine Katze auf dem Dachboden.' > ▸
haus.txt
sed 's/Katze/Maus/' haus.txt    # 1. Katze durch Maus ersetzen
sed 's/Katze/Maus/g' haus.txt   # Alle Katzen durch Mäuse ersetzen
cut -d' ' -f2 haus.txt          # 2. durch Leerzeichen getrennter Teil
sed 's/Katze/Maus/' haus.txt | cut -d' ' -f2   # Kombiniert
```

Ausführen (in aktuellem Verzeichnis):

```
bash MausInDaHaus.sh
```

Dinge zum Selbstnachschlagen

```
man <Befehl>
```

Scrollen mit den Pfeil- bzw. Bild-Tasten (nach oben/unten). Zum Anfang **springen** mit "Pos1", zum Ende mit "Ende". Suchen mit "/<Begriff>" + Return, nächstes Vorkommen suchen mit "/" + Return. **Beenden** mit "q".

- nc: Daten über Netzwerke übertragen
- cp: Dateien kopieren
- mv: Dateien verschieben/umbenennen
- rm: Dateien/Verzeichnisse löschen

Lizenz (CC BY-SA 3.0)

http://creativecommons.org/licenses/by-sa/3.0/de/

Sie dürfen:

- das Werk bzw. den Inhalt vervielfältigen, verbreiten und öffentlich zugänglich machen
- Abwandlungen und Bearbeitungen des Werkes bzw. Inhaltes anfertigen
- das Werk kommerziell nutzen

Zu den folgenden Bedingungen:

- **Namensnennung** — Sie müssen den Namen des Autors/Rechteinhabers in der von ihm festgelegten Weise nennen.
- **Weitergabe unter gleichen Bedingungen** — Wenn Sie das lizenzierte Werk bzw. den lizenzierten Inhalt bearbeiten oder in anderer Weise erkennbar als Grundlage für eigenes Schaffen verwenden, dürfen Sie die daraufhin neu entstandenen Werke bzw. Inhalte nur unter Verwendung von Lizenzbedingungen weitergeben, die mit denen dieses Lizenzvertrages identisch oder vergleichbar sind.

Autor: Hauke Goos-Habermann (goos-habermann.de), Bilder (z.T. abgewandelt) von http://openclipart.org

LÖSUNG: Tux in der Waschmaschine: Animierte GIFs erstellen

```
#!/bin/bash

# Verzeichnis für temporäre Dateien erstellen
mkdir tmp

# Aufräumen
rm tmp/*.png

# Rahmen zu penguin.png hinzufügen und in penguin-rahmen.png speichern
convert penguin.png -bordercolor none -border 5x5 penguin-rahmen.png

i=10
# In 30°-Schritten rotieren
for grad in `seq 0 30 330`
do
    # penguin-rahmen.png rotieren und unter $i.png abspeichern
    convert -page 200x200 -background none -filter point -distort SRT ▸
$grad -bordercolor none penguin-rahmen.png -gravity Center $i.png
    # Bildnummer hochzählen
    i=$[$i + 1]
done

# PNGs zu einem animierten GIF zusammenfügen
convert -delay 100 -dispose Background -loop 0 -repage 144x144+3+3 *.png ▸
-scale 64x64 anim.gif
```

LÖSUNG: Die BAfH-Ausrede: Daten aus dem Internet lesen und aufbereiten

```
#!/bin/bash

# Einen Satz Ausreden in die Datei /tmp/bafh.txt herunterladen
wget "http://identi.ca/api/statuses/user_timeline.json?
screen_name=bastardausrede&count=200&callback=?" -O /tmp/bafh.txt

# /tmp/bafh.txt durchsuchen, auseinandernehmen und Umlaute konvertieren
sed 's/","/\n/g' /tmp/bafh.txt | grep '"text":"' | sed -e ▸
's/.*"text":"//g' -e 's#\\u00fc#ü#g' -e 's#\\u00f6#ö#g' -e ▸
's#\\u00e4#ä#g'  -e 's#\\u00df#ß#g' -e 's#\\/#/#g' -e 's#\\u00dc#Ü#g'
```

LÖSUNG: Verstreute Daten wiederfinden: Zentrale Suchdatenbank von Dateien auf diversen Medien (DVDs, CDs, Festplatten, ...) anlegen und darin suchen

DVDs einlesen

```
#!/bin/bash
# Parameter überprüfen
if [ $# -ne 1 ]
then
    echo "Benutzen $0 <Mountpunkt>"
    echo "<Mountpunkt> wo z.B. das DVD-Lauferk gemountet wird (z.B. ▸
/mnt/DVD)"
    exit
fi

# Mountpunkt setzen aus 1. Kommandozeilenparameter
dvdMountPoint=$1
# Verzeichnis zum Speichern der Index-Dateien
INDEXDIR="/tmp/index"

# Namen der DVD einlesen
echo "Namen der DVD eingeben:"
read dvdname

# Überprüfen, ob der Indexdateiname noch nicht benutzt ist
while [ -f $INDEXDIR/$dvdname ]
do
    echo "Name existiert bereits. Neuen Namen eingeben:"
    read dvdname
done
```

DVDs suchen

```
#!/bin/bash
# Verzeichnis zum Speichern der Index-Dateien
INDEXDIR="/tmp/index"

# Suchbefehl erstellen
s="grep -i -r $1 $INDEXDIR"

# 1. Kommandozeilenparameter wegwerfen
shift

# Weitere Parameter bearbeiten
for i in $@
do
    s="$s | grep -i $i" # und an Suchbefehl anhängen
done

# Suchbefehl ausführen
echo $s | bash
```