

Übungsblatt #04

Bearbeitungszeitraum: bis Donnerstag, 12.11.2009

- Was will uns dieses Übungsblatt sagen?*
1. Shellscripte sind letztendlich nichts anderes als eine Aneinanderreihung einzelner Kommandos. Man könnte alles, was man in ein Shellscript schreiben, genauso gut auch selber nacheinander in eine Shell eingeben.
 2. Deshalb spricht man auch davon, dass Shell-Skripte interpretiert werden, wohingegen Java- und C-Programme (sowie wie Programme anderer Sprachen auch) kompiliert (übersetzt) werden.
 3. Wenn man ein und dieselbe Aufgabe mehrfach machen muss, lohnt es sich, ein Shellscript zu schreiben und es
 4. Das test-Kommando kann insbesondere auch prüfen, ob Dateien oder Verzeichnisse bereits existieren.
 5. Außerdem kann es prüfen, ob ein Eintrag im Verzeichnis eine (reguläre) Datei oder ein weiteres Verzeichnis ist.
 6. Das, was wir bislang über reguläre Ausdrücke gelernt haben, können wir auch verwenden, um Dateiparitäten wie z.B. das Umbenennen auszuführen.
- durch Kommandozeilen-Argumente jeweils beim Aufruf anzupassen (in unserem Beispiel entspricht dies der Angabe der URL zur ZIP-Datei).

Unordnung im Dateisystem

Ihr kennt ganz sicher das Problem, dass die Verzeichnisse auf Eurem Computer (oder in Eurem Heimatverzeichnis auf unseren Maschinen) im Laufe der Zeit immer unübersichtlicher werden, weil sich jede Menge „Zug“ ansammelt. Für diese Aufgabe simulieren wir mal ein etwas außer Kontrolle geratenes Verzeichnis mit Bild-Dateien.

Auf unserer Homepage findet Ihr die Datei „unordnung.zip“. Wenn Ihr diese auspackt, so erhaltet Ihr über 100 Dateien. Die Dateinamen stets mit den Buchstaben „pic“ (für Picture), denen dann eine vierstellige Zahl folgt (so benennen ja in der Regel auch Digitalkameras Eure Bilder). In den Dateieindungen unterscheiden sich die Dateien dann: es gibt PDF-Dateien, JPG-Dateien, TIF-Dateien und RAW-Dateien (Endung CR2) und manche Dateien existieren in mehreren Formaten. Dummerweise sind manche Dateinamen mit großen Buchstaben geschrieben und manche mit kleinen.



- Mit Hilfe eines Shellscripts soll nun Ordnung geschaffen werden. Schreibt deshalb ein Shellscript, das die folgenden Aufgaben erledigt:
1. Zunächst lädt es sich die Datei selbstständig von unserer Homepage herunter und packt sie im aktuellen Verzeichnis aus. Die URL zum ZIP-Archiv gibt Ihr Eurem Skript als Argument mit auf den Weg.
 2. Wir möchten keine Dateien mit Großbuchstaben haben. Deshalb soll Euer Skript nun alle Dateien, die in Großbuchstaben vorliegen, so umbenennen, dass sie nur noch aus kleinen Buchstaben (und natürlichen Ziffern) bestehen. Aber Vorsicht, denn...
 3. ... es dürfen beim Umbenennen keine Dateien überschrieben werden. Wenn es also im Archiv eine Datei namens „pic0001.jpg“ und eine namens „pic0001.jpg“ gibt, dann darf die Umbenennung

nicht vorgenommen werden. Stattdessen soll das Skript dann eine entsprechende Warnung ausgeben.

4. Jetzt gehen wir ans Aufräumen: wir legen fest, dass eine Dateieindung immer genau drei Buchstaben hat, denen ein Punkt vorangeht (hier: „pdf“, „jpg“, „dng“, „cr2“). Lasst Euer Skript für jede Dateieindung, die vorkommt, ein Unterverzeichnis erstellen, in das die jeweiligen Dateien dann verschoben werden. Mit anderen Worten: es soll ein Verzeichnis „pdf“ geben, das nur PDF-Dateien enthält. Ebenso entsteht also ein reines DNG-Verzeichnis, ein reines JPG-Verzeichnis usw.

5. Damit Euer Skript auch „zukunftsicher“ ist

Beispiellauf des Skripts:

Das aktuelle Verzeichnis ist zunächst einmal leer:

```
heid@h3: ~/blatt04$ pwd
~/home/heid/blatt04
heid@h3: ~/blatt04$ ls
heid@h3: ~/blatt04$
```

Das Skript wird gestartet

```
claudum@myhess.sh http://www.mathematik.uni-ulm.de/sai/ws09/scriptspra-chen/uebungen/04/unordnung.zip
```

Es folgen nun jede Menge Ausgabe-Zeilen, die von wget und unzip automatisch erzeugt werden. Euer Skript soll aber „seinen Senf“ dazu tun und auf bereits vorhandene Dateien hinweisen:

```
could not rename PIC0002.DNG to pic0002.jpg: file already exists!
Could not rename PIC0005.JPG to pic0005.jpg: file already exists!
[...] und noch viele weitere dieser Art...
```

Jetzt sieht das Verzeichnis wie folgt aus:

```
heid@h3: ~/blatt04$ pwd
~/home/heid/blatt04
heid@h3: ~/blatt04$ ls
media unordnung.zip
heid@h3: ~/blatt04$
```

Wie gewünscht liegt nun alles im Verzeichnis media (Unterverzeichnisse / doppelte Dateien):

```
heid@h3: ~/blatt04$ cd media
heid@h3: ~/blatt04/media$ ls
cr2 pdf PIC0006.DNG PIC0017.TIF PIC0026.JPG PIC0036.PDF tiff
dng PIC0002.DNG PIC0014.DNG PIC0021.TIF PIC0030.TIF PIC0039.PDF
jpg PIC0005.JPG PIC0017.CR2 PIC0022.JPG PIC0032.CR2 PIC0048.JPG
```

und z.B. demnächst auch mit MP3-Dateien umzugehen weiß, sollen die Verzeichnisnamen nicht durch Euch im Skript erwähnt, sondern vom Skript selbstständig ermittelt werden.

Der Kasten unten zeigt Euch, wie die Ausgabe des Skripts für die Beispieldatei „unordnung.txt“ und das Ergebnis am Ende aussieht:

P.S.: Das Skript lässt sich in unter 30 Zeilen schreiben!

P.P.S.: Lasst Euer Skript nun mal mit der Datei „fleechaos.zip“ (das ebenfalls auf der Homepage bereitsteht) laufen. Wenn alles klappt, solltet Ihr nun von der Flexibilität Eures Skripts überzeugt sein!