Linux und Textdateien

Effizient arbeiten mit einfachen Mitteln

Patrick Bucher

11.04.2017

Windows- und Mac-User schreiben Texte meistens mit *Microsoft Word*. Als Dateiformat verwenden sie *Office Open XML* (OOXML1, `.docx`). Mit *Libre Office Writer*2 gibt es eine kostenlose Alternative zu *Microsoft Word*. Diese Software verwendet ein eigenes Dateiformat: *OpenDocument Text* (ODT3, `.odt`).

Beide Programme (*Word* und *Writer*) untersützen beide Formate (OOXML und ODT). Im Gegensatz zu *Word* funktioniert aber *Writer* auch auf Linux. Hartgesottene Linux-User verwenden aber lieber einfache Textdateien zum Speichern und Bearbeiten ihrer Texte. Warum?

Unterschiede zwischen Textdateien und *OOXML/ODT*

OOXML und ODT speichern Texte in einer komprimierten XML-Struktur ab. Einfache Textdateien haben keine solche Struktur, sondern bestehen nur aus einer Reihe von Zeichen. Dadurch ergeben sich einige Unterschiede in der Handhabung von OOXML/ODT-Dateien einerseits und Textdateien andererseits:

1. OOXML/ODT-Dateien lassen sich nur mit *Word* und *Writer* zuverlässig bearbeiten. Textdateien lassen sich mit einem beliebigen Texteditor bearbeiten.
2. OOXML/ODT-Dateien enthalten viele Zusatzinformationen für Formatierung, Struktur und zusätzliche Einstellungen. Textdateien haben diesen *Overhead* nicht, da sie nur die eigentlichen Nutzdaten enthalten.
3. OOXML/ODT-Dateien können nur so lange gelesen werden, wie die enstprechenden Programme dazu verfügbar sind. Textdateien, die nur aus einer Reihe von Zeichen bestehen, können immer gelesen werden.
4. OOXML/ODT-Dateien lassen sich nur im «WYSIWYG»-Modus («what you see is what you get») bearbeiten. Textdateien lassen sich auch als *Textstrom* bearbeiten.

Aus den ersten drei Punkten ergeben sich drei offensichtliche Vorteile für Textdateien. Doch welche Vorteile bieten *Textströme*?

Anwendungsbeispiel: Wörter in mehreren Artikeln zählen

Angenommen, wir haben eine Reihe von Artikeln; einmal im ODT-Format (`.odt`) und einmal im Textformat (`.txt`). Nun wollen wir herausfinden, welcher Artikel in Wörtern gemessen der längste ist. Mit unseren ODT-Dateien verfahren wir folgendermassen:

1. Wir öffnen den ersten Artikel mit *Writer*.
2. Wir gehen auf das Menü *Tools* und wählen den Eintrag *Word Count*.
3. Wir notieren uns den Dateinamen und die Anzahl Wörter dazu.
4. Wir schliessen den Artikel.
5. Wir wiederholen den Vorgang für den nächsten Artikel.

Dieses Vorgehen ist sehr aufwändig. Zudem muss der ganze Vorgang zu einem späteren Zeitpunkt wiederholt werden, falls die Artikel in Zwischenzeit umgeschrieben wurden.

Mit Textdateien funktioniert das einfacher. Man verwendet einfach folgende Befehlszeile, die untenstehende Ausgabe ergibt:

```
$ wc -w *.txt | sort -n -r

2220 eigenes-bier-brauen.txt
1739 berlinreise.txt
1231 neues-aquarium.txt
 893 im-stau.txt
```

Doch was hat das ganze zu bedeuten? Schauen wir uns die Befehlszeile genauer an:

Der `wc`-Befehl

- `wc`4 steht für «word count» und zählt die Wörter in einer Datei.
- Standardmässig gibt `wc` die Anzahl Zeilen, Wörter und Zeichen einer Datei aus. Mit dem Parameter `-w` (für «words») erhalten wir nur die Anzahl Wörter.
- Mit `*.txt` übergeben wir dem Programm sämtliche Textdateien im Arbeitsverzeichnis.

`wc -w *.txt` ist der erste Teil des Befehls. Führte man ihn aus, erhielte man folgende Ausgabe:

```
$ wc -w *.txt

1739 berlinreise.txt
2220 eigenes-bier-brauen.txt
 893 im-stau.txt
1231 neues-aquarium.txt
```

Wir sehen die Anzahl Wörter in der ersten Spalte, doch die Zeilen sind nicht sortiert.

Die Ausgabe an `sort` weiterleiten

Über das Muster `*.txt` erhält `wc` die Dateien in alphabetischer Reihenfolge, und gibt sie auch in alphabetischer Reihenfolge wieder aus. Darum kommt jetzt `sort`[5] zum Zug.

- Zwischen den Befehlen steht das Zeichen `|`. Das ist eine sogenannte *Pipe*, zu Deutsch etwa «Röhre». Eine Pipe nimmt die Ausgabe eines Programmes entgegen und leitet sie als Eingabe zum nächsten Programm weiter. *Der Text «fliesst» also wie ein Strom durch die Röhre.* (Textstrom)
- Das nächste Programm ist `sort`, das Textzeilen in alphabetisch aufsteigender Reihenfolge sortiert.
- Da wir keine alphabetische, sondern eine numerische Sortierung benötigen («100» wäre gemäss alphabetischer Sortierung kleiner als «9»), geben wir den Parameter `-n` («numeric») an.
- Zudem soll die Reihenfolge nicht aufsteigend (die kleinste Zahl am Anfang) sondern absteigend (die grösste Zahl am Anfang) sein, was wir mit dem Parameter `-r` («reverse») machen.

Wir haben gesehen, dass sich ein alltägliches Problem mithilfe von Textdateien und einfachen Programmen (`wc` und `sort`) effizienter lösen lässt als mit einer Textverarbeitung.

Quellen

1. Office Open XML
2. LibreOffice Writer
3. Open Document Format
4. Der `wc`-Befehl
5. Der `sort`-Befehl