Nice to know Stuff

Find a spezific folder by name and delete it recursivly. Search down from current directory:

```
# find . -type d -name "@eaDir" -print0 | xargs -0 rm -rf
```

```

michael@backup-server:/mnt/backup-disk/web$ find . -type d -name "@eaDir"
./enjoy/img/@eaDir
./enjoy/img/base/backgrounds/@eaDir
./tools/images/@eaDir
./download/languages/@eaDir
./download/templates/default/@eaDir
./download/templates/@eaDir
./download/templates/simple_image_gallery/@eaDir
./download/@eaDir
./download/index_icons/michael/@eaDir
./download/index_icons/winvista/@eaDir
./download/index_icons/kde/@eaDir
./download/index_icons/apache/@eaDir
./download/index_icons/winxp/@eaDir
./download/index_icons/osx/@eaDir
./download/files/@eaDir
./download/files/programs/@eaDir
./download/files/programs/MorphVOX Pro/@eaDir
..
```

Find a spezific file by name and delete it. Search down from current directory:

```
# find . -type f -name "Thumbs.db" -print0 | xargs -0 rm -f
```

```

michael@backup-server:/mnt/backup-disk/web$ find . -type f -name "Thumbs.db"
./enjoy/img/Thumbs.db
./enjoy/img/base/backgrounds/Thumbs.db
./tools/images/Thumbs.db
./download/index_icons/michael/Thumbs.db
./download/index_icons/winvista/Thumbs.db
./download/index_icons/kde/Thumbs.db
./download/index_icons/apache/Thumbs.db
./download/index_icons/winxp/Thumbs.db
./download/index_icons/osx/Thumbs.db
./_index_content/Thumbs.db
./books/index_icons/michael/Thumbs.db

```

```
./books/index_icons/winvista/Thumbs.db
./books/index_icons/kde/Thumbs.db
./books/index_icons/apache/Thumbs.db
..
```

Account Type: “Standard” or “Administrator”.

On the command line, run the **command id** or **groups** and see whether you are in the sudo group. On Ubuntu, normally, administrators are in the sudo group.

You may also have administrative access if you've been directly added to the list of sudoers — this is more likely if the administrator is familiar with Linux or Unix in general and didn't use the default Ubuntu method. Try running sudo echo ok and enter your password; if this prints ok, you're an administrator.

“grep” the available space from “df” output:

```
# df | grep -oP '/sda1.* \K\d+(?=\s+\d+%)'
```

```

michael@backup-server:~$ df | grep -oP '/sda1.* \K\d+(?=\s+\d+%)'
994425716

michael@backup-server:~$ df
Filesystem                            1K-blocks        Used   Available Use%
Mounted on
/dev/root                              15039728     1152500    13243704   9%
/
devtmpfs                                 470184           0      470184   0%
/dev
tmpfs                                    474792           0      474792   0%
/dev/shm
tmpfs                                    474792        6308      468484   2%
/run
tmpfs                                      5120           4        5116   1%
/run/lock
tmpfs                                    474792           0      474792   0%
/sys/fs/cgroup
/dev/mmcblk0p1                            41853       21327       20526  51%
/boot
tmpfs                                     94956           0       94956   0%
/run/user/1001
/dev/sda1                            1441091564   373256712   994561592  28%
/mnt/backup-disk
```

Explanation: Here, we match /sda3, then as many characters as possible until we find a stretch of numbers (\d+) which is followed by one or more spaces (\s+), then one or more numbers (\d+) and a %. The foo(?=bar) construct is a positive lookahead, it allows you to search for the string foo only if it is followed by the string bar. The \K is a PCRE trick that means “discard anything matched up to this point”. Combined with -o, it lets you use strings that precede your pattern to anchor your match but not print them.

Without -P, things are trickier. You would need multiple passes. For example:

```
df | grep -o '/sda3.*%' | grep -Eo '[0-9]+ *[0-9]+%' | grep -Eo '^[0-9]+'
```

Split text on whitespace in terminal output

To split text on whitespace you can use grep. There’s an infinite amount of ways to do this. This is one of them.

```
# echo 'string --with ###ALLKINDS### 0f ::outputs' | grep -oP '[^\s]+'
```

```

# echo 'string --with ###ALLKINDS### 0f ::outputs' | grep -oP '[^\s]+'
string
--with
###ALLKINDS###
0f
::outputs
```

---

curl only write to file if successful status 200

Make curl get the contents of a URL and write to file, but only write to file if the response is successful:

```
# curl -s -S -f -o blackgate-feed.json "$blackgate_rz"
```

- -s keeps curl quiet by hiding progress meter and error messages
- -S shows an error message if it fails (stderr)
- -f Fail silently (no output at all) on server errors, keeping stdout clean
- -o specifies an output file

Pipe video stream from raspberry pi to local computer with ffplay

I use this to get a live video stream from my Raspberry Pi with Camera attached

Execute this on the Pi, where TARGET_IP is my local computer where I will watch the stream, and PORT is an arbitrary port number.

```
# raspivid -t 999999 -o - | nc -u $TARGET_IP $PORT
```

Execute this on the local computer where you will watch the video stream

```
# nc -ul $PORT | ffplay -
```

- https://www.raspberrypi.org/blog/camera-board-available-for-sale/
- https://blog.philippklaus.de/2013/06/using-the-raspberry-pi-camera-board-on-arch-linux-arm/

Generating a pseudorandom password or string in Linux bash

Define a function in e.g. **~/.bashrc**

```

genpasswd() {
  tr -dc A-Za-z0-9 < /dev/urandom | head -c ${1:-36} | xargs
}
```

Where 36 is default length if no parameter is given

Usage:

```
# genpasswd
GVQ3ZHqrBRDzB1QwASA9uk6YsZPto2GWeRWR

# genpasswd 7
qvPWx7N
```

Last update: **2019/05/06 17:33**