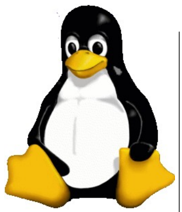
shell card



<...> replace with valid name
[...] lists optional parameter

man <command>
explanation for given command

apropos [-s <nr>] <string>
searches for commands

Filesystem

/ start of the filesystem
~ the home directory
. the directory in which the user is currently working
.. the parent directory

pwd
prints the current working directory

cd <directory>
change position in the filesystem to <directory>
without parameter back to the home directory

file <file>
infos about the content of a file

ls [-alitdr1] <name>
lists the content of a directory
-a all, including hidden files
-l list attributes of entries
-i inode numbers
-t sorted by time
-r reverse order
-d infos about the directory and not its content

mkdir [-pm] <directory>
create directory
-p create missing dir in a path
-m set access permission

find <start dir> [option <parameter>]
search the filesystem below <start dir>
-iname <file> searches for <file>
-type <type> searches for file types
-user <user> searches for files belonging to <user>

mv <old> <new>
move or rename <old> to <new>

cp <file> <new>
copy file <file> to <new>

ln [-sf] <file> <link>
create a second name for a file
hardlink different names for same inode
-s symbolic link
-f force

touch <file>
sets the access date and time to now

rm [-rf] <file>
delete <file>
-r recursive
-f force

df [-h] <directory>
information about filesystems
-h human readable

rsync -avz –progress source target
source or target must be local
source or target [user@][host:]<dir>

dd if=<in file> bs=<size> of=<out file>
bs blocksize

file content

less <file>
more <file>
lets you sroll trough the content of a file.

tail [-nf]
shows the last 10 lines of a file
-n shows the last n lines
-f shows the last 10 lines and waits for further content.

cat <file>
shows the content of a file

file <file>
analyses content of a file

grep [-ivln] <expression> <file>
search for <expression> in a file.
-i ignore capital/normal spelling
-v show everything except lines with <expression>
-l show filename
-n show line number

user

passwd [<user>]
change passwort
the superuser root can also change the password for other user.

id [<user>]
information about user

who [am i]
information about logged on users.

usefull commands

tar [-ctxvf] <file list>
create / extract an archive
-c create
-t show content
-x extract
-v verbose
-f <file>

top
display resource usage

iotop
shows the I/O usage

ls -l /proc/<PID>/fd/
shows which files are opened by <PID>

Permission

umask <permission>
permission which will **not** be set when creating file or directory

chmod [ugo=+-rwx] <file>
controls the basic access permission of <file>
= sets permission
+/- grants or retract permission
ugo user/group/other
rwx read/write/execute

process

ps [-efu]
process information
-e every process on the system
-f full information
-u <user> process information for a specific user

kill [-19] <pid>
kills a process
-1 the process gets an signal to stop
-9 the process will be killed

uname [-an]
information about operating system
-n hostname
-a infos about version, architecture, ...

pgrep [-lf] <pattern>
search for process id by <pattern>
-f search <string also in parameters
-l print command name

pkill -<signal> [-fl] [-U] <pattern>
-<signal> signal to be send to process
-U only this user ID/name

http://oelkers.de/tips/pocketCardShell.pdf

administration

useradd [-d] <user>

- create a new user <user>
- -d with home directory

passwd [<user>]

- change password
- the superuser may also change the password of other user.

groupadd <group>

- create a new user group

usermod [-acdgG] <group>

- modify the user attributes
- -a append groups (only with -G)
- -c change comment
- -d change home directory
- -g set primary group
- -G additional Groups

mount -t <fs-type> <special device> <directory>

- mounting filesystems

mkfs -t <fs-type> <special device>

- create filesystem on special device

cryptsetup [luksFormat | luksOpen]

- enrcpt/decrypt filesystem
- luksFormat <special device>
  - format and encrypt <special dev>
- luksOpen <special dev> <crypt-dev>
  - creates crypto-dev in /dev/mapper

network

ping [-c] <remote host>

- being alive check of <remote host>
- -c <count> number of pings to send

arp [-dsn]

- address resolution protocol information
- -d <remote host> delete entry from
- -s <remote host> <remote mac> insert entry
- -n no name resolution

ifconfig [<eth0> [<ip-address>]]

- interface information and manipulation

netstat [-an]

- print network information
- -a all
- -n do not resolve host and port names

lsof -i

- list open connection
- -i4 ipv4 connection
- -i :<port> connection for this port

tracroute <remote host>

- print route to remote host

tcpdump [-vv]

- packet sniffer
- -vv more information
- -i <interface> for <interface>

iptables [-LFAI]

- maintaine filrewall
- -L liste network roules
- -F flush firewall
- -A append roule
- -I insert roule

shell build in

positional variables

$0	name of script
$1-$9	positional parameters
${10}	positional paramters 10 onward
$#	nr of positional parameters
$*	all positional parameters
$?	returnwalue

[<expression>]

- test expression
- `==, !=` tstring comparsion
- `-eq,-ne,-le,-lt,-ge,-gt,` numeric comparsion
- `-f` exists file?
- `-d` exists directory?
- `-s` file contains content

if [<expression>]
then
    <commands>
[else]
    <commands>
fi

- executes commands dep. on expr.

case <variable> **in**
    <expression>) <commands> **;;**
    <expression>) <commands> **;;**
        ***)** <commands> **;;**
esac

- executes commands when variablen content equals <expression>.

while [<expression> **]**
do
    <commands>
done

- repeat while <expression> is true

for <variable name> **in** <parameter list>
do
    <commands>
done

- repeat for every parameter once.

function <name>
{
    <commands>
}

- defines a function

alias <name>=<command>

- defines a alias for a command

read [-p <string>] variable list

- requests input from user and places it word wise in the variables

regular expression

.	character
[...]	exactly one char in brackets
[^...]	exactly one char not in bracket
<re>*	repeatedly the same reg. expr.
^<re>	reg. expr. at the begin of line
<re>$	reg. expression at end of line
\(<re>\)	mark reg. expression
\n	substitute marked reg. expr.
&	substitute found string

i/o redirection

[0]< <file>	redirect stdin. read from file
[1]> <file>	redirect stdout. write to file
2> <file>	redirect stderr. write to file
>>, 2>>	append to file
|	concatenate stdout of one cmd to stdin of the next cmd
2>&1	send stderr to stdout
$(<command>)	command substitution

http://oelkers.de/tips/pocketCardShell.pdf