



Git Bash crash kursus

Start small, move fast and aim high





Der er nogle vigtige tegn som du måske ikke ved hvor er på dit tastatur.

- | (pipe)
- < og >
- \$
- ` (backtick)
- ~ (tilde)

Hvor er du nu?

- pwd
 - Læg mærke til stien
- cd
 - ~ (relativ sti)
 - /c/Users/dig (absolut sti)
- df (hvor meget plads har jeg)

Øvelser

- Find en option til df så pladsen angives i GB
- cd til det øverste niveau



What is there?

Find **processer** med ps-kommandoen (bliver mere interessant på en riktig linux)

```
$ ps
  PID  PPID  PGID  WINPID  TTY      UID  STIME  COMMAND
 1980     1  1980    1088 ?      197108 14:49:12 /usr/bin/mintty
 2145  1981  2145     620 pty0  197108 15:41:14 /usr/bin/ps
 1981  1980  1981    2316 pty0  197108 14:49:12 /usr/bin/bash
```

Find **filer** med ls-kommandoen

```
Administrator@DESKTOP-7QA27BH MINGW64 ~/temp
$ ls -l
total 1
-rw-r--r-- 1 Administrator 197121 0 Aug 24 15:46 test.sh
-rwxr-xr-x 1 Administrator 197121 31 Aug 24 15:47 test2.sh*
```

Øvelser

- Cd til Downloads og udfør flg:
 - curl -k "<https://raw.githubusercontent.com/cphstud/20m9596V2-uge5/master/cars.csv>" --output cars.csv -s
 - curl -k "<https://raw.githubusercontent.com/cphstud/20m9596V2-uge5/master/test.sh>" --output test.sh -s
 - curl -k "<https://raw.githubusercontent.com/cphstud/20m9596V2-uge5/master/test2.sh>" --output test2.sh -s
- Hvor stor er filen test.sh?
- Sammenlign rettighederne med test2.sh. Hvad er forskellen?



What is there
part II?

Find **indhold** af filer med grep-kommandoen. Kombinerer man det med | (pipe) bliver det et stærkt værktøj

```
$ grep Audi cars.csv | grep 2dr
9;Audi;A4 1.8T convertible 2dr;4;170;23;30;3638;105;2005;95923
15;Audi;A4 3.0 convertible 2dr;6;220;20;27;3814;105;2011;59989
16;Audi;A4 3.0 quattro convertible 2dr;6;220;18;25;4013;105;2012;71989
22;Audi;TT 1.8 convertible 2dr (coupe);4;180;20;28;3131;95;1978;36022
23;Audi;TT 1.8 quattro 2dr (convertible);4;225;20;28;2921;96;1979;48027
24;Audi;TT 3.2 coupe 2dr (convertible);6;250;21;29;3351;96;1980;60034
```

Man kan også lede efter bestemte filer vha **find** og så kombinere med grep. Læg mærke til –v (mønster jeg ikke vil ha' med):

```
$ find . -type f | grep hosts | grep -v 64 | grep -v exe | grep -v dll
./System32/drivers/etc/hosts
./System32/drivers/etc/hosts.sam
```

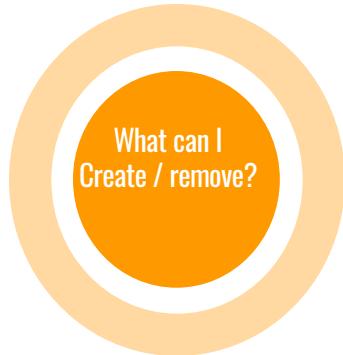
Eller vise **indhold** af filer med cat eller (ved store filer) head eller tail

```
Administrator@DESKTOP-7QA27BH MINGW64 ~/test/filer/store
$ head -2 cars.csv
obs;Make;Model;Cylinders;Horsepower;MPG_City;MPG_Highway;Weight;Wheelbase;Year;Milage
1;Acura;MDX;6;265;17;23;4451;106;1997;19

Administrator@DESKTOP-7QA27BH MINGW64 ~/test/filer/store
$ tail -2 cars.csv
427;Volvo;V40;4;170;22;29;2822;101;1983;37237
428;Volvo;XC70;5;208;20;27;3823;109;1984;49647
```

Øvelser

- Prøv at se indholdet med cat cars.csv.
- Undersøg hvor mange biler der er med to døre (2dr). Hint: Kombinér grep med wc



Lav **mapper** med mkdir-kommandoen. En vigtig option er –p så den kan lave hele strukturen

```
Administrator@DESKTOP-7QA27BH MINGW64 ~
$ mkdir -p test/filer/store

Administrator@DESKTOP-7QA27BH MINGW64 ~
$ find ./test
./test
./test/filer
./test/filer/store
```

Lav **filer** med touch-kommandoen.

```
$ touch test.sh

Administrator@DESKTOP-7QA27BH MINGW64 ~/test/filer/store
$ ls -l
total 0
-rw-r--r-- 1 Administrator 197121 0 Aug 28 07:54 test.sh
```

Lav **filer med indhold** med cat-kommandoen og redirection (slut med <ENTER> og derpå <ctrl+d>).

```
Administrator@DESKTOP-7QA27BH MINGW64 ~/test/filer/store
$ cat > test2.sh
#!/bin/bash
echo "hello world"
```

Øvelser

- Lav en folder – temp – som ligger i dit home-dir
- Lav en fil – test3.sh – i temp-folderen som printer dit navn
- Hent cmd.sh fra <https://github.com/cphstud/20m9596V2-uge5> og læg det i temp-folderen
- Eksekver filen således: ./cmd.sh 10 og se hvad der sker



Fjern **indhold** med rm-kommandoen. En vigtig option er –rf så den kan fjerne hele strukturen. Man kan bruge wildcards hvis man vil fjerne meget. Men pas på med rm –rf * der fjerner alt.

```
Administrator@DESKTOP-7QA27BH MINGW64 ~/temp
$ ls
cmd.sh* kurt0/ kurt1/ kurt2/ kurt3/ kurt4/
Administrator@DESKTOP-7QA27BH MINGW64 ~/temp
$ rm -rf kurt*

Administrator@DESKTOP-7QA27BH MINGW64 ~/temp
$ ls
cmd.sh*
```

Flytte eller kopiere **filer og mapper** med mv eller cp. En vigtig option til cp er –r så den kan kopiere hele strukturen.

```
Administrator@DESKTOP-7QA27BH MINGW64 ~/temp
$ mv kurt0 kurt007

Administrator@DESKTOP-7QA27BH MINGW64 ~/temp
$ cp -r kurt1 kurt1007

Administrator@DESKTOP-7QA27BH MINGW64 ~/temp
$ cp cmd.sh cm2.sh

Administrator@DESKTOP-7QA27BH MINGW64 ~/temp
$ ls
cm2.sh* kurt007/ kurt10/ kurt11/ kurt2/ kurt4/ kurt6/ kurt8/
cmd.sh* kurt1/ kurt1007/ kurt12/ kurt3/ kurt5/ kurt7/ kurt9/
```

Øvelser

- Sørg for at fjerne alle filer i din temp-folder. Bortset fra cmd.sh
- Få cmd.sh til at lave 200-kurtfoldere. Læg en stor fil ind et sted så du overskriver én af dem, der ligger der allerede.
- Byt med din sidemakker. Se hvem der først finder den store fil.



What did I do?

Find alle dine **kommandoer** vha history-kommandoen. Læg mærke til at output fra history pipes over i head-kommandoen. Ellers bliver outputtet for langt

```
Administrator@DESKTOP-7QA27BH MINGW64 ~/stat (master)
$ history | head
 1  ls
 2  pwd
 3  mkdir temp
 4  cd temp/
 5  ls
...
```

Du kan gemme **kommandoer** i en fil vha redirection.

```
Administrator@DESKTOP-7QA27BH MINGW64 ~/stat (master)
$ history > minekommandoer

Administrator@DESKTOP-7QA27BH MINGW64 ~/stat (master)
$ ls -l minekommandoer
-rw-r--r-- 1 Administrator 197121 9255 Aug 28 19:29 minekommandoer
```

Øvelser

- Lav en folder *stat* i dit homedir. Gem din historik i den folder med
- Undersøg hvordan du kan bruge date-kommandoen til at lave en fil der hedder: hist_08282021_1931 <dags dato og tidspunkt>
- Forsøg at få følgende til at virke: history > file-som-hedder-dags-dato-og-tidspunkt



I dit home-dir skal der være to filer - .bash_profile og .bashrc
og dine configurationer kommer i .bashrc

```
Administrator@DESKTOP-7QA27BH MINGW64 ~
$ cat .bash_profile
if [ -f ~/.bashrc ]; then
    source ~/.bashrc
fi
```

```
Administrator@DESKTOP-7QA27BH MINGW64 ~
$ cat .bashrc
alias lt="ls -ltra"
```

Du kan gøre dit terminalliv meget lettere ved at bruge
aliasser. De hører hjemme i .bashrc

Øvelser

- Lav et alias så du kan se din history med aliasset h
- Lav et alias så du kan greppe i din historik med flg: gr mkdir (hvor gr er alias for en kombination af history og grep)

Shell-programming er en kunst for sig. I git-videoen ser man en lille smule ..

```
Administrator@DESKTOP-7QA27BH MINGW64 ~
$ cat .bash_profile
if [ -f ~/.bashrc ]; then
    source ~/.bashrc
fi
```

Common constructs ..

```
#!/bin/bash
cnt=0
limit=$1
while [ $cnt -le $limit ]; do
    rd=`echo $RANDOM | tr '[0-9]' '[a-z]`"
    fn=`echo $RANDOM | tr '[0-9]' '[a-z]`"
    echo "Create dir $rd"
    mkdir -p kurt${cnt}/${rd}/kurt${cnt}
    touch kurt${cnt}/${rd}/kurt${cnt}/${rd}_${fn}.jpg
    cnt=$(( cnt + 1 ))
done
```

Argument Variables	
\$0	program name
\$1	1 st argument
\$2	2 nd argument
...	...
\$9	9 th argument
\$*	all arguments
\$#	No. of arguments

Common Constructs	
while read f	read text
do	file line
echo "Line is \$f"	by line
done < file	
foo=`ls`	get output of command

Test Operators	
if["\$x" -lt "\$y"];	# do something
then	
fi	
Numeric Tests	
lt	less than
gt	greater than
eq	equal to
ne	not equal
ge	greater or equal
le	less or equal
File Tests	
nt	newer than
d	is a directory
f	is a file
r	readable
w	writeable
x	executable

Øvelser

- Lav et lille script der tager to argumenter – en streng og et tal - og skriver dit navn + strengen det antal gange tallet angiver

How to VIM

VIM-editoren er også en kunst for sig. Man skal minimum kunne slippe ud af den igen ..

```
MINGW64:/c/Users/Administrator
Administrator@DESKTOP-7QA27BH MINGW64 ~
$ vim .bashrc
```

```
MINGW64:/c/Users/Administrator
alias lt="ls -ltra"
~
~
~
~
~
~
.bashrc [unix] (19:45 28/08/2021)
:q!
```

Øvelser

- Åben .bashrc med vim og kopier en linje med kommandoen yy og p

- Opret en "Git"-folder i dit homedir
 - Hent github.com/cphstud/explore_california
 - Brug rm -rf til at fjerne .git-folderen
 - Brug find, grep og wc til at finde antallet af png-filer
 - Brug find, egrep og wc til at finde antallet af gif,png & jpg
 - Brug cp -r til at lave en backup af explore_california til f.eks explore_california_bu
 - Brug find og grep -v til at få en liste over al indhold bortset fra assets
 - Inde fra explore_california udfører du følgende kommando:
 - sed -i 's/866/999/g' index.html
 - Undersøg vha diff-kommandoen ændringer i index.html
 - diff <den ændrede fil> <den uændrede fil i din backup>
 - Overvej om du tør udføre følgende (eller kan du validere de enkelte steps):
 - sed -i 's/866/999/g' `find . -type f | egrep -v "assets"`

- Gå til folderen hvor du downloaded cars.csv
 - Flg kommando giver antal biler sorteret på bilmærke
 - cat cars.csv | cut -d\"; -f2 | sort | uniq -c | sort -k1n
 - Modificer kommandoen så den i stedet sorterer på antal cylindre. Hvor mange biler har 6 cyl?
 - Brug grep til at finde ud af hvor mange Audi A6 der er i filen
 - Kan nogen forklare hvorfor grep "A6 3.0" cars.csv viser tre biler mens grep "A6 3.0" kun viser to?
- Tør du <https://gist.github.com/zachbrowne/8bc414c9f30192067831fafebd14255c> ?
- Hvis ja ..
 - Så ændre/tilføj alias til find så den (grep) ignorerer case.
 - Find ud af hvorfor whatismyip ikke virker og få den til at virke.
- Hvis nej
 - tilføj et alias "lt" som udfører "ls -ltra"
 - Tilføj et alias "psg" som udfører "ps -aef | grep -i "
 - Tilføj et alias for history
 - Tilføj et alias som grepper i din history

Ekstra øvelse: Lav 2 nøglepar

I git-bash (husk at 2.gang skal du kalde den noget andet)

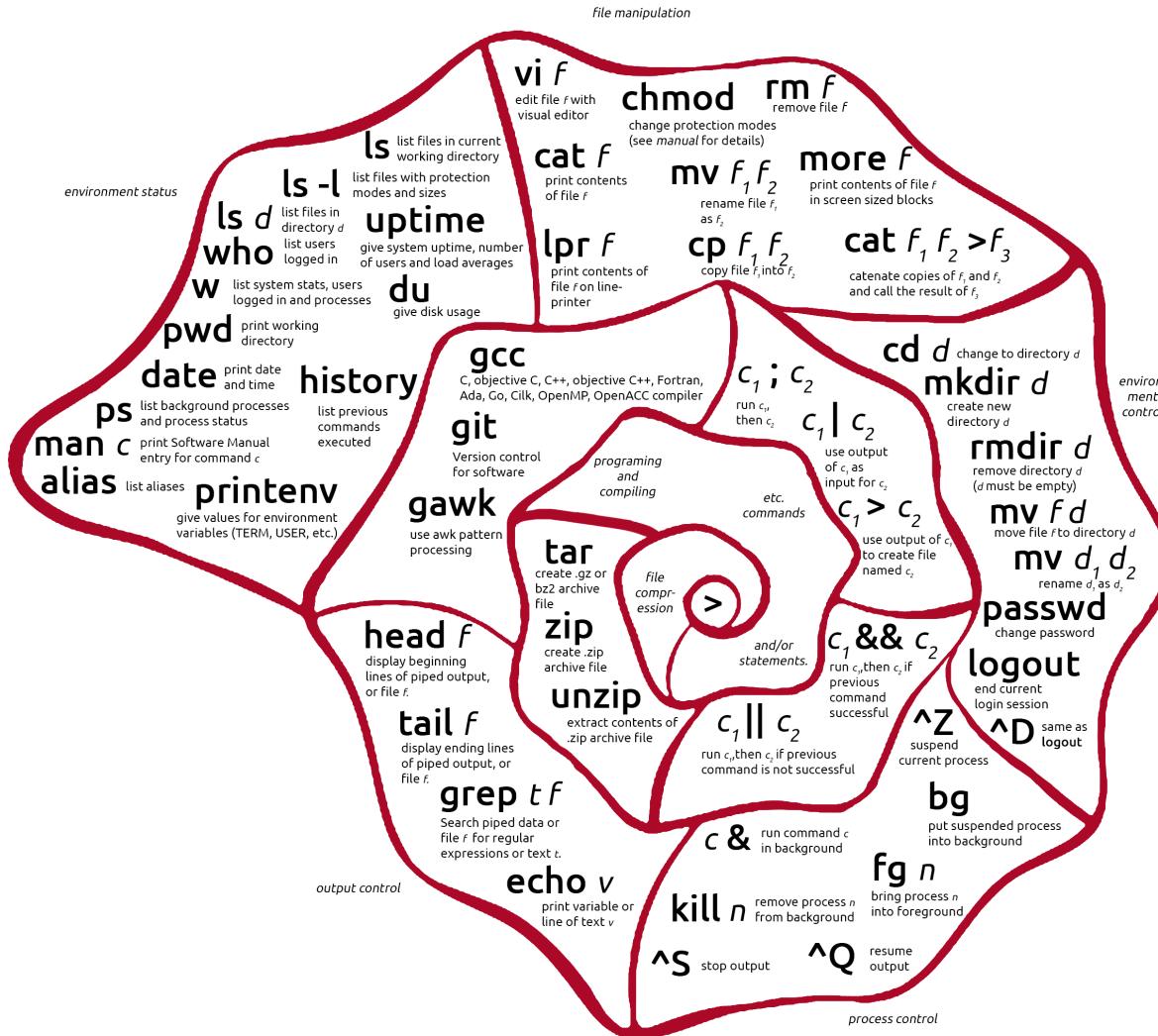
MINGW64:/c/Users/vagrant

```
vagrant@DESKTOP-F245E69 MINGW64 ~
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/vagrant/.ssh/id_rsa):
```

- Nu skal du "catte" indholdet af id_rsa.pub:

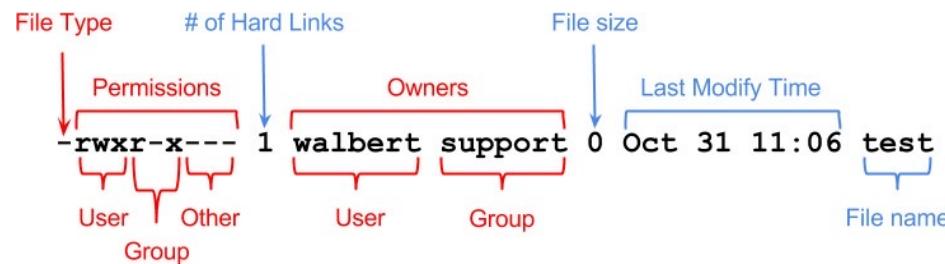
```
$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQgQDF8r0VqIr56PCJhAd5L+Ru5mBFTLcYLso7p7X3XgNnWe0BmWb8HxiNaK4XDPE+22f/pXCwdtWkr3JaagDTjHFyD941+
ap179tZWrjTzqo3r9Gr0DYLUv1zu01j/+KyAqhPB1wku0fHCix0bYb00o7pPWDIAwbNaUYcMtKgtiwEELssosTILTpFPu8itWaJV6WmQnf2ndvpdfAb+f7qDaNHQRbeN6EZ
VwmFHu4repPA/1A6Ssk1c3LgykiVRVw/v1LbGzikudv4BhAmMH7hQQKp1zRKKs2RLrUx4R/poLOL8QjJFjtqKsGzxYmb7zk3PaQidkni4aZZZVyu04ejJCeI3FZ3z7FO
TjcmsFLDKv7sLR770V3phNAbfWveX4+4A0NQLsH6r+DGftVawV+GYa5j1aC7bzZh1jccWVfgXNiPEF0ILJqMYgtXb1jbqR2z9LIXeduP+bqpR5YdZYldUxHg6tCKGDGtC7
CPsLIlnJfcLqD6ILGeCYg2txm7c4s0U= vagrant@DESKTOP-F245E69
```

Et lidt anderledes cheat-sheet



File permissions

If the command `ls -l` is given, a long list of file names is displayed. The first column in this list details the permissions applying to the file.



The `chmod` command changes the permission on a given file or directory.

`chmod` sets permissions in two ways.

- Using symbols
- Using octal values

Octal	Symbol	Permission
0	---	No Permissions
1	--x	Execute
2	-w-	Write
3	-wx	Write and Execute
4	r--	Read
5	r-x	Read and Execute
6	rw-	Read and Write
7	rwx	Read, Write, and Execute