

# Introduktion til Terminal

Lars Nielsen  
lnc13.lars@gmail.com

1. februar 2014



# Forord

Denne bog er tiltænkt nybegynder (n00bz) inden for \*NIX<sup>1</sup> verden. Som godt kunne tænke sig at få en introduktion i hvad der ligger under den poleret brugergrænseflade.

## Om forfatteren

Lars er født i 1989 så et lævn fra det forige årtusind. Lars er uddannet datamatiker fra Aarhus Erhvervs Akademi sommeren 2013. Men han start med at læse datalogi ved Aarhus Universitet i 2009, men det var nu lidt for teoretisk, så han skiftet til den mere praktisk tilgang og læser nu software ved Aalborg Universtet under School of information and computer technology. Desuden har Lars arbejdet ved flere virksomheder der udarbejder webbaseret løsninger og nogle få virksomheder der arbejder primært med back-end løsninger. Kort sagt Lars er nørd, og bruger det mest af sin tid med softwareudvikling og primært ligger hans interesse inden for back-end, system udvikling og systemhåndtering.

## Kontakt:

Hvis du har kommentar, forslag eller andre ting til denne bog, ville det være at fortrække at smide en kommentar på bogens github side, som findes her [github.com/loopTools/intro\\_til\\_terminal](https://github.com/loopTools/intro_til_terminal). Hvor der findes en issue side, hvor sådanne ting bliver håndteret. Har du derimod et spørgsmål til Lars selv kan du kontakt ham på følgende mail adresse: [lnc13.lars@gmail.com](mailto:lnc13.lars@gmail.com)

## Tak til:

Tak til [www.linux.dk](http://www.linux.dk) og holdet bag, for at give mig lov til at bruge siden til at start min serie af guides, som er blevet samlet til den her bog.

Tak til Peter Lyberth og Kim Rostgaard Christensen, som begge står bag [linux.dk](http://linux.dk) og som af og til har læst nogle af online guidesne igennem, for stavefejl.

Tak til folkne bag L<sup>A</sup>T<sub>E</sub>X, som bogen er opbygget i.

---

<sup>1</sup>\*NIX er UNIX eller Linux baseret systemer

# Indhold

<b>1</b>	<b>Introduktion</b>	<b>1</b>
1.1	Hvad bliver ikke dækket . . . . .	1
1.2	Kommando opbygning . . . . .	1
1.3	MAN - Din ven i mørket . . . . .	2
<b>2</b>	<b>Navigering i filsystemet</b>	<b>4</b>
2.1	PWD - hvor er vi? . . . . .	4
2.2	LS - Hvad er der her? . . . . .	5
2.3	CD - Jeg vil væk . . . . .	5
2.4	MKDIR - Ny mappe . . . . .	6
2.5	CP - Kopi . . . . .	6
2.6	MV - Flyt dig . . . . .	6
2.7	RM - Væk med dig . . . . .	7
<b>3</b>	<b>Så kikker vi på diske</b>	<b>8</b>
3.1	df - fri disk . . . . .	8
3.1.1	-h - Menneskligt læsligt . . . . .	8
<b>4</b>	<b>Historie</b>	<b>9</b>
<b>5</b>	<b>Processer</b>	<b>11</b>
5.1	TOP hvad kører . . . . .	11
5.2	KILL og PKILL . . . . .	12
<b>6</b>	<b>Små gode ting</b>	<b>13</b>
6.1	Mange kommandoer på engang . . . . .	13
6.2	ALIAS . . . . .	13
6.3	CURL . . . . .	14
<b>A</b>	<b>Ordliste</b>	<b>15</b>

# Kapitel 1

## Introduktion

Siden den personlige computers oprindelse, har der været to primære input enheder til computer den ene er pegeværktøjet (musen, pointwheel trackball) og tastaturet. De fleste bruger i dag en kombination af de to, til at navigere og bruge deres computere. Så er der folk som os (nørder) som fortrækker at flytte hænderne mindst muligt fra tastaturet, medmindre vi rækker hånden ud af kaffen. Vi elsker at kunne bruge vores computere og systemer uden at skulle klikke rundt. Vi har et hemmeligt våben til dette, dette våben er bedre kendt som Terminal eller kommandolinje.

Dette våben er et lævn fra den gang computer ikke havde en mus og kun tastaturet som input enhed.

Terminal eller Terminalen er et kommandolinje værktøj som findes i forskellig udgaver, men de fleste systemer bruger de samme kommandoer. De kommandoer som bliver brugt i denne bog kan bruges på Mac OS X, Linux og BSD, hvis en funktion ikke findes på alle systemer bliver dette naturligvis noteret.

### 1.1 Hvad bliver ikke dækket

Der er elementer af den kommando linje baseret verden, som ikke bliver dækket i denne bog og med god grund.

**Server opsætning** er et helt emne for sig selv, og er simpelt hen for stort til at blive dækket i en begynder bog på dette niveau og det varierer meget fra system til system.

**Tekstværktøjer** som en del af det at bruge kommandolinjen, er tekstværktøjer og der findes forskellig slags (Vi(m), Emacs, Nano, Pico, ...). Og da der er en kæmpe regionskrig i mellem specielt Vi(m) og Emacs har jeg valgt ikke at dække dem. I bogen bruges Vi eller Emacs. Jeg vil dog prøve at holde mig til Vi, da fleste systemer kommer med denne som standard.

**Hacking** er et mere avanceret emne og kræver lidt evner og bliver derfor ikke dækket i denne bog.

**User management** eller brugerhåndtering, er i bund og grund ikke så svært, men varierer for meget på tværs af systemer til at give et godt overordnet indblik.

### 1.2 Kommando opbygning

For at kunne benytte kommandolinjen eller terminalen, benyttes kommandoer. Disse kommandoer har en opbygning og for at benytte disse kommandoer er det ret vigtigt at forstå denne opbygning. En Kommando

består primært af tre dele, disse tre dele kaldes kommandoen, flag og argumenter.

## Kommandoen

Er navnet på den kommando som man ønsker at benytte sig af. navnet er altid den først del af et kommandokald. Eksempler på kommandonavne er; *cd*, *ls*, *pwd* og *mv*.

## Flag

Flag er parameter, som benyttes til at ændre eller manipulere det output en kommando returnerer disse kommer oftes efter kommandonavnet eksempelvis; *ls -a* som bliver beskrevet senere. Flag starter oftes med et dash (-) og derefter et enkelt bogstav. En kommando kan sagtens tage flere flag eller ingen, det ser vi eksempler på senere.

## Argumenter

Disse kommer oftes efter flag og er det "element" som en kommando bliver udført, forskellig kommando kan tage mere en et argument. Eksempel på kommando uden flag, men med et argument *ls /test\_2*

## 1.3 MAN - Din ven i mørket

Når man vil kende sin funktion lidt mere i dybten, bruger man en kommando som hedder *man*. *man* står for manual og giver et overblik over kommando, mulige flag og argumenter. se eksempel neden for

---

h271: intro_til_terminal	tools\$ man ls	
LS(1)	BSD General Commands Manual	LS(1)
NAME	ls -- list directory contents	
SYNOPSIS	ls [-ABCFGHLOPRSTUW@abcdefghijklmnopqrstuvwxyz1] [file ...]	
DESCRIPTION	<p>For each operand that names a file of a type other than directory, <i>ls</i> displays its name as well as any requested, associated information. For each operand that names a file of type directory, <i>ls</i> displays the names of files contained within that directory, as well as any requested, associated information.</p> <p>If no operands are given, the contents of the current directory are displayed. If more than one operand is given, non-directory operands are displayed first; directory and non-directory operands are sorted separately and in lexicographical order.</p> <p>The following options are available:</p>	

---

*man* siden kan navigeres med med piletasterne og man kommer udaf den ved at indtaste *q*. Hvis du vil vide lidt mere om *man*, så har *man* selv en man side og den tilgås ved at skrive *man man*

## Kapitel 2

# Navigering i filsystemet

En af de vigtigste ting at kunne i kommandolinjen, er at have evnerne til at navigere OS'ets filsystem. Derfor starter vi med at forklare de forskellige kommandoer til at finde rundt.

### 2.1 PWD - hvor er vi?

En ret relevant del af navigering er ofte at vide hvor man i det hele taget er. Det er her kommandoen *pwd* kommer ind i billedet. *pwd* står for **P**rint **W**orking **D**irectory, kommandoen returnerer den sti som man arbejder i lige nu.

---

```
h271: intro_til_terminal tools $ pwd
/Users/tools/Documents/intro_til_terminal
```

---

#### Og hvad så

Hvad kan man så bruge det til? Man kan altid bruge ens nuværende "position" til at navigere filsystemet. Men for at kunne bruge information skal man kunne forstå resultat, så lad os bryde stien ned.

#### **intro\_til\_terminal**

Det sidste element af stien er den mappe, man arbejder i og interager med. Det er altså her andre kommandoer som interagerer med filer og visse andre kommandoer udføre deres arbejde hvis de bliver kaldt.

**/Users/tools/Documents/**

Er den overordnet filsti, som man er nødt til at komme igennem for at kunne komme til *intro\_til\_terminal*.



## 2.2 LS - Hvad er der her?

En anden vigtig del af at arbejde i filsystemet, er at vide hvilke filer og mapper, som er i den mappe man arbejder i. Det er til dette formål man bruger kommandoen *ls*. *ls* betyder list directory contents, altså list indholdet af en mappe. Hvis kommandoen kaldes ud flag, returneres en liste af alt, som ikke er skjult i mappen.

---

```
h271: intro_til_terminal tools$ ls
LICENSE      fil_sys_nav .tex intro.tex      main_book.tex
README.md    forord.aux   main_book.aux main_book.toc
compile.sh    forord.tex   main_book.log  ordliste.aux
fil_sys_nav .aux intro.aux     main_book.pdf  ordliste.tex
```

---

### Men jeg har skjulte filer

På UNIX baseret, systemer er alle filer og mapper som starter med *.* skjulte, eksemplet kunne hede *.skjult*. Den vil vi altså gerne kunne se, så vi bruger flaget *-a* som står for all eller alle, også får man en lidt anderledes liste returneret.

---

```
h271: intro_til_terminal tools$ ls -a
.      LICENSE      fil_sys_nav .tex intro.tex
..     README.md    forord.aux   main_book.aux
.git   compile.sh    forord.tex   main_book.log
.skjult fil_sys_nav .aux intro.aux     main_book.pdf
```

---

### List alt i en undermappe

Man kan også liste alt i mappe ved at give stien til mappe

---

```
h271:test tools$ ls
t_1  t_2  test_2
h271:test tools$ ls t_1
demo.txt
```

---

## 2.3 CD - Jeg vil væk

For at kunne navigere rundt i sit filsystem skal man naturligvis også kunne komme væk fra den mappe man er i. Dette gøres via kommandoen *cd*, som står **C**hange **D**irectory eller ændre mappe.

---

```
h271: intro_til_terminal tools$ pwd
/Users/tools/Documents/intro_til_terminal
h271: intro_til_terminal tools$ cd test/
h271:test tools$ pwd
/Users/tools/Documents/intro_til_terminal/test
```

---

Hvis man vil et "trin" op i filsystemet skriver man `cd ..`.

## 2.4 MKDIR - Ny mappe

Man kan få brug for oprette en mappe, dette gøres med kommandoen; *mkdir* som betyder **MaKe Directory** eller lav mappe. Kommandoen opretter en mappe i den nuværende arbejdesmappe.

---

```
h271:test tools$ pwd
/Users/tools/Documents/intro_til_terminal/test
h271:test tools$ ls
h271:test tools$ mkdir test_2
h271:test tools$ ls
test_2
```

---

## 2.5 CP - Kopi

Hvis man ønsker at kopiere en fil, gøres dette med kommandoen *cp* som står for **C**opy. Kommandoen kræver 2 argumenter; filen som ønskes kopieret og destinations mappen. Så hvis man har en mappe *t\_1* som indholder en fil *demo.txt* som man ønsker at flyttet til mappen *t\_2* gøres det sådan her:

---

```
h271:test tools$ ls t_1
demo.txt
h271:test tools$ ls t_2
h271:test tools$ cp t_1/demo.txt t_2
h271:test tools$ ls t_1
demo.txt
h271:test tools$ ls t_2
demo.txt
```

---

## 2.6 MV - Flyt dig

**M**ove eller flyt er en måde hvor på man kan flytte en fil eller en mappe fra et sted i systemet til et andet. Lige som *cp* tager *mv* to argumenter, elementer som man vil havde flyttet og destinations mappen.

---

```
h271:test tools$ ls t_1
demo.txt
h271:test tools$ ls t_2
h271:test tools$ mv t_1/demo.txt t_2/
h271:test tools$ ls t_1
h271:test tools$ ls t_2
demo.txt
```

---

## Omdøb

Man kan også omdøbe en fil med kommandoen *mv*, dette gøres ved at tage en fil som først parameter og bruger en fil som anden parameter, eksempelvis

---

```
h271:t_2 tools$ ls
demo.txt
h271:t_2 tools$ mv demo.txt demo_2.txt
:t_2 tools$ ls
demo_2.txt
```

---

## 2.7 RM - Væk med dig

**ReMove** eller fjern er en kommando som kan bruges til at slette filer.

---

```
h271:test tools$ ls t_2/
demo_2.txt
h271:test tools$ rm t_2/demo_2.txt
h271:test tools$ ls t_2/
```

---

### -r fjer alt under mig og mig

Hvis man ønsker at slette en mappe og alt der ligger i mappen, skal man sætte følgende flag på *rm -r*, hvor *-r* står for rekrusiv. Hvilket vil sige at den udføre kommandoen rekrusivt.

---

```
h271:test tools$ ls
t_1    t_2    test_2
h271:test tools$ ls test_2/
demo_2.txt
h271:test tools$ rm -r test_2/
h271:test tools$ ls
t_1    t_2
```

---

### RMDIR - Fjern mappe med indhold

Hvis man vil fjerne en mappe, som man ved er tom, kan man skrive *rmdir* og så mappens navn som argument.

---

```
h271:test tools$ ls
t_1    t_2
h271:test tools$ ls t_2
h271:test tools$ rmdir t_2/
h271:test tools$ ls
t_1
```

---

## Kapitel 3

# Så kikker vi på diske

I dette kapitel kikkeres der på, hvordan man får overblik over, hvor meget plads der er på ens interndisk og eksterndiske, pludseligt små sjove ting omkring diske.

### 3.1 df - fri disk

*df* free informere dig om hvor meget plads der bliver brugt på din disk og hvor meget der er frit på disken. For at benytte kommandoen indtastes simpelthen bare *df*

```
larss-mbp:intro_til_terminal tools$ df
Filesystem            512-blocks      Used Available Capacity    iused   ifree %used Mounted on
/dev/disk0s2          1463469952 1180339824 282618128      81% 147606476 35327266      81% /
devfs                  364          364          0    100%      631          0    100% /dev
map -hosts              0            0          0    100%          0          0    100% /net
map auto_home           0            0          0    100%          0          0    100% /home
```

#### 3.1.1 -h - Menneskligt læsligt

Som kan ses på outputtet i billedet her over, er outputtet ikke videre forståeligt, for ikke teknisk folk. Men *df* har et flag som er *-h*, som giver et bedre og mere læsbar output og kaldes således *df -h*.

```
Larss-MacBook-Pro:intro_til_terminal tools$ df -h
Filesystem      Size  Used Avail Capacity    iused   ifree %used Mounted on
/dev/disk0s2    698Gi  648Gi  49Gi    93% 170001446 12932296    93% /
devfs           189Ki  189Ki   0Bi   100%      652          0    100% /dev
map -hosts       0Bi    0Bi   0Bi   100%          0          0    100% /net
map auto_home    0Bi    0Bi   0Bi   100%          0          0    100% /home
localhost:/bYSLgDbJGvN4gBD4Tv92qS 698Gi  698Gi   0Bi   100%          0          0    100% /Volumes/MobileBackups
/dev/disk1s2     931Gi  890Gi  42Gi    96% 233224550 10881865    96% /Volumes/MyBook
//Lars%20Nielsen@tools_TC.afpovertcp_tcp.local/tools 1.8Ti  517Gi  1.3Ti   28% 135528221 352326125   28% /Volumes/tools
/dev/disk2s1     3.8Gi  3.7Gi  110Mi   98%          0          0    100% /Volumes/UDEN NAVN
```

## Kapitel 4

# Historie

Hvis man ofte bruger samme kommando med samme parametre, eller ikke lige kan huske hvordan en kommandos struktur den er, kan du bruge kommandoen *history*

---

```
h271: intro_til_terminal tools$ history
495 sudo rm /etc/postgres-reg.ini
496 brew list
497 brew update
498 brew install postgresql
499 sudo port install postgresql93
500 sudo port selfupdate
501 cd Documents/intro_til_terminal/
502 ls
503 open main_book.pdf
504 clear
505 history
h271: intro_til_terminal tools$
```

---

Dette er som det kan ses et meget lille udsnit af alle kommandoer, man kan scrolle igennem sin historie. Her kan man så se tidlige kommandoer man har kaldt.

### Kald mig igen

Som det kan ses i outputet ovenfor har være kommando fået tildelt et nummer, eksempelvis har *ls* fået nummer 502. det nummer er ret behjælpeligt hvis man ikke ønsker at indtastet hele kommandoen igen, så kan man i stedet for indtaste *!nummer* og så bliver kommandoen kørt igen, Men man skal være opmærksom på at tallet ændre sig, som kommandoerne bliver skubet ud af historien.

---

```
h271: intro_til_terminal tools$ !502
ls
#compile.sh#  disk.tex      hist.aux      main_book.aux  mkdir
#hist.tex#    disk.tex~    hist.tex      main_book.log  ordliste.aux
LICENSE       fil_sys_nav .aux hist.tex~      main_book.out  ordliste.tex
README.md     fil_sys_nav .tex images        main_book.pdf  test
compile.sh    forord.aux   intro.aux     main_book.tex
```

## Grib mig i historien

Hvis man kender den kommando, man ønsker at kalde, men bare har glemte eksempeltvis parameterne der skal sendes med. Kan man benytte *grep* i kombination med *history*, ved at skrive følgende kommando *history | grep #navn.på\_kommando#*, | kaldes en pipe. Neden for ses et eksempel, hvor historien for *vi* gerne vil findes

---

```
h271: intro_til_terminal   tools$ history | grep vi
   50  vi compile.sh
  114  vi ~/. bash_profile
  206  vi App.rb
  393  vi superliga.c
  410  vi merge.rb
  413  vi merge.rb
  414  vi merge
  415  vi merge.rb
  419  vi merge.rb
  491  vi Makefile
```

---

# Kapitel 5

## Processer

Som de fleste ved hvis de kommer fra eksempelvis Windows eller Mac OS X, kan man se hvilke processer der kører via en jobliste på \*nix systemer kan dette også gøres via kommandolinje og man kan også dræbe de processer som er løbet løbsk.

### 5.1 TOP hvad kører

Top viser de mest belastede processer der kører og lidt information om forskellige ting. Kommandoen kaldes ved at skrive *top* og følgende output bliver vist

---

```
h271: intro.til_terminal tools$ top
Processes: 188 total, 3 running, 4 stuck, 181 sleeping, 832 threads   11:22:47
Load Avg: 1.55, 1.62, 1.67   CPU usage: 10.8% user, 14.47% sys, 75.43% idle
SharedLibs: 175M resident, 0B data, 35M linkedit.
MemRegions: 39816 total, 2761M resident, 130M private, 770M shared.
PhysMem: 5441M used (1216M wired), 2750M unused.
VM: 454G vszize, 1310M framework vszize, 0(0) swapins, 0(0) swapouts.
Networks: packets: 2348958/2325M in, 2145578/1783M out.
Disks: 2436784/21G read, 207259/6427M written.
```

---

PID	COMMAND	%CPU	TIME	#TH	#WQ	#PORT	#MREG	MEM	RPRVT	PURG
4362	mdworker	0.0	00:00.09	4	0	54	58	1976K 1084K	0B	
4152	top	11.7	00:34.09	1/1	0	23	35	2244K 2020K	0B	
4149	com.apple.au	0.0	00:00.03	2	1	48	53	1328K 648K	0B	
4148	com.apple.au	0.0	00:00.01	2	1	28	42	1004K 444K	0B	
4147	com.apple.qt	0.0	00:00.07	2	0	73	81	2956K 1452K	0B	
4146	com.apple.We	0.0	00:03.40	9	0	243	557	44M 40M 72K		
4129	com.apple.Co	0.0	00:00.01	2	1	30	41	956K 400K	0B	
4122	com.apple.Pr	0.0	00:00.01	2	1	35	42	980K 412K	0B	
4119	Preview	0.0	00:06.53	4	0	205	444	35M 38M 20M		
3926	Twitter	0.0	00:03.23	8	1	223	744	59M 47M 196K		
3920	com.apple.hi	0.0	00:00.01	2	0	31	38	916K 368K	0B	
3919	com.apple.qt	0.0	00:00.08	2	0	72	81	2932K 1428K	0B	
3918	com.apple.We	3.9	00:23.06	11	2	267	1249+	127M+ 110M+ 692K		

## 5.2 KILL og PKILL

*kill* og *kill* bruges til at dræbe processer, men på to forskellig måder.

### KILL

Basere sig på en nummer som kaldes *PID* som er en process id, og som kan ses i outputtet for top. Alle processer har et *PID* og de er alle unik (det vil sige ikke to er ens). Det at der er unikke tillader at man kan bruge *kill* til at dræbe en process, lad os dræb Twitter processen som har PDI: 3926

---

```
h271: intro_til_terminal tools$ kill 3926
```

---

### PKILL - Vi dræber på navn

Man kan også dræbe en process via dens kommando navn, så hvis man lige som under *kill* vil dræbe Twitter kan det gøres således med *kill*

---

```
h271: intro_til_terminal tools$ kill twitter
```

---

Med *kill* skal man være lidt mere påpasselig, da der godt kan køre flere processer med samme kommando navn. Der for anbefales det at man som nybegynder benytter *kill*.



# Kapitel 6

## Små gode ting

I dette kapitel dækkes små ”hacks” til at kunne gøre arbejdet med kommandolinjen letter.

### 6.1 Mange kommandoer på engang

Når man med tiden opdager at flere kommando tit bliver ud ført sammen og i hvis række følge, bliver man træt af at skrive en kommando trykke enter og så skrive næste kommando, og det er her man bliver glad for *EE* som tillader en at link kommandoer. Eksempelvis, hvis man vil lave en mappe også gå direkte ind i den, altså kombinere *mkdir* og *cd*, og i dette eksempels tilfælde også *pwd* kan dette gøres således .

---

```
h271: intro_til_terminal  tools$ pwd && mkdir kombi_test && cd kombi_test && pwd
/Users/tools/Documents/intro_til_terminal
/Users/tools/Documents/intro_til_terminal/kombi_test
```

---

Som udføre alle kommandoerne i række følge.

### 6.2 ALIAS

Kommandoer kan være lange og svære og huske, derfor kan man benytte sig af *alias*. Alias er i sig selv ikke kommando men en funktionalitet, som er en del af ens kommandolinjeværktøj. For at benytte sig af alias, skal der editeres i en fil som kaldes *.bashrc*, denne fil kan editeres ved at benytte *VI*. Som eksempel, vil vi benytte kommandoen *cd* som eksempel, også vil vi gerne ændre nuværende mappe til skrivebordet, vi gør således:

---

```
h271: intro_til_terminal  tools$ vi ~/.bashrc
```

---

Når *VI* åbnes skal man trykke *i* for at indsætte de tegn der er behov for og på en ny linje nederst, i file skrives:

```
alias cddesktop='cd ~/Desktop'
```

For at gennem trykkes der på escape knappe (esc), og så følge *:wq. :wq* er en vi kommando, w = write, q = quit. Når man har gemt, skal man genlæse sin *.bashrc* fil, dette gøres således:

---

```
h271: intro_til_terminal tools$ . ~/.bashrc
```

---

## 6.3 CURL

I dette eksempel benyttes material fra [www.ruby-lang.org](http://www.ruby-lang.org)

CURL er en måde hvor på en fil kan hentes fra en server, det gøre ret simpelt med *curl #adresse til filen på serveren#*, men outputtet giver ikke altid mening. Derfor kan man gøre brug af flaget *-O*, som gør at filen kan gives et navn. kommandostrukturen ser således *curl #navn\_på\_fil# #adresse til filen på serveren#* og man skal være i den mappe, man ønsker filen skal ligge, eller give stien ved fil navnet. Se eksempel neden for

---

```
h271h271:curl tools$ ls
h271:curl tools$ curl -o ruby-2.1.0.tar.gz
http://cache.ruby-lang.org/pub/ruby/2.1/ruby-2.1.0.tar.gz
% Total    % Received % Xferd Average Speed   Time    Time     Time Current
                               Dload  Upload  Total   Spent    Left  Speed
 100 14.3M  100 14.3M    0     0 6368k      0  0:00:02  0:00:02 --:--:-- 8655k
h271:curl tools$ ls
ruby-2.1.0.tar.gz
```

---

# Bilag A

## Ordliste

Da ikke alle benytter de samme ord for alt, er der her en ordlist, som forklare ordne som bliver brugt.

**Mappe** et element som kan indeholde andre mapper og filer. Kaldes også for folder.

**\*NIX** er et gennerelt udtryk for systemer som basere sig på UNIX og GNU/Linux

**\*BSD** er en familie af UNIX baseret systemer som bygger på BSD eller Berkely Software Distrobution, familiemedelmmmer er blandt andre Mac OS X, FreeBSD og OpenBSD.

**OS X og Mac OS X** er det system som Apple Inc. benytter på deres bærebare og stationære computer. OS X er delvist baseret på NextSTEP og FreeBSD.

**Parameter** disse kaldes også argumenter, men i denne bog brugers argumenter om noget andet.

**Dash (-)** kaldes normalt bindestreg.