Korte uitleg: De enkelvoudige hiërarchische bestandsorganisatie (directory tree, absolute and relative pathes)

Unix/Linux gebruikt voor het organiseren van de gegevensbestanden op mass-storage (schijf in al zijn vormen) een boomstructuur. Zo'n boomstructuur start altijd bij de wortel. En dat is in het Engels dus de root, maar dat heeft niets te maken met de gebruiker/user met de naam `root`! Er zijn meer operating systemen die zo'n boomstructuur gebruiken om bestanden te organiseren, maar let op: Unix/Linux heeft slechts één zo'n structuur in een lopend systeem. Alle verschillende stukjes mass-storage (file systems) die aan het systeem bekend zijn worden ergens in die boomstructuur opgehangen (ge'mount'). Dit in tegenstelling tot bijv. MS-DOS, waar ieder stukje los van de andere stukjes zijn eigen boomstructuur en dus zijn eigen root (bij MS-DOS aangegeven door een hoofdletter met een `:` erachter) heeft. Unix/Linux doet hier dus meer wat een operating system moet doen: de hardware voor de gebruiker verbergen. De gebruiker gebruikt bestanden ergens in de boom en weet niet op (welk stukje van) welke schijf dat staat.

De vertakkingen in de boom zijn ook bestanden met een speciale structuur genaamd 'directory' (in de bureaublad metafoor van de GUI gebruiker worden zij ook wel 'map' of 'folder' genoemd). Iedere directory bevat de gegevens over de bestanden in die directory (en dat kunnen dus ook weer directories zijn). Gevolg is dat een bestandsnaam uniek moet zijn binnen een directory, maar dezelfde naam kan in andere directories dus weer voorkomen. Om een bestand zonder enige twijfel aan te duiden moet je dus zeggen: ga vanuit de root directory naar directory `usr` en daarin naar directory `lib` en daarin naar directory `X11` en daarin naar directory `displaymanagers` en daar vindt je het bestand `xdm`. Daar is een kortere en exactere shrijfwijze voor `/usr/lib/X11/displaymanagers/xdm`. Uiteraard valt op dat de `/` wordt gebruikt om steeds een stapje hoger te gaan in de boomstructuur. De `/` is dan ook één van de weinige tekens die niet in een bestandsnaam mag voorkomen. Let er ook op dat de root directory eigenlijk geen naam heeft, maar dat de `/` erachter (en dus helemaal aan het begin van bovenstaande aanduiding) aangeeft dat dit pad/path (want dat is het) vanuit de root is opgebouwd. Een zogenaamd absoluut pad/absolute path.

De uitdrukking "absolute path" betekent waarschijnlijk dat er ook een "relative path" bestaat. En dat is ook zo. De vraag is natuurlijk: relatief ten opzichte van wat? Hiervoor is het begrip "working directory" ingevoerd. Je kunt dat ook zien als: de plaats waar je je bevindt in de directory tree. Nou bevind je je dus niet echt daar, maar deze plaats is opgeslagen in een environment variable `PWD`:

```
henk@boven:/usr/lib/X11/displaymanagers> echo $PWD
/usr/lib/X11/displaymanagers
henk@boven:/usr/lib/X11/displaymanagers>
```

Je ziet hier niet alleen de inhoud van `PWD`, maar ook dat `bash` (bij verstek) de inhoud ervan in je prompt zet om je te laten zien "waar je bent". Er is nog een environment variable in dit verband:

```
henk@boven:/usr/lib/X11/displaymanagers> echo $OLDPWD
/usr/lib/X11/etc
henk@boven:/usr/lib/X11/displaymanagers>
```

Die bevat de WD die je had voor je de huidige zette. Makkelijk om weer een stapje terug te gaan.

Natuurlijk kun je beide variablelen "met de hand" op waardes zetten. Maar dat is niet erg handig, afgezien van het feit dat er dan tikfouten in kunnen worden gezet zonder waarschuwing. Daarom heeft `bash` het ingebouwde commando `cd` (change directory). Wat doet `cd` (uiteraard kijk je in de `man` pagina van `bash` voor een beschrijving)?

- het controleert of de nieuwe WD bestaat:

  ```
  henk@boven:/usr/lib/X11/displaymanagers> cd aap
  bash: cd: aap: Bestand of map bestaat niet
  henk@boven:/usr/lib/X11/displaymanagers>
  ```

- het copieëert de huidige waarde van `PWD` naar `OLDPWD`;
- het zet `PWD` op de nieuwe waarde.

En er zijn speciale gevallen:

- ```
  cd
  ```

  dit is gelijk aan

  ```
  cd $HOME
  ```

- ```
  cd -
  ```

  dit is gelijk aan

  ```
  cd $OLDPWD
  ```

Samenvattend: de "working directory" (de inhoud van variablele `PWD`) is de plaats in de directory tree van waaruit een relative path wordt berekend.

Een relative path is een path dat niet begint met een `/`.

Aanbeveling: Als je meer dingen (en zeker ingewikkelde dingen) gaat doen in een directory, "ga" dan eerst naar die directory. Je schermt jezelf in zekere mate af van wat op andere takken van de boom zit. Vooral het steeds intikken

van lange paden kan tot fatale tikfouten leiden.

Met een relative path kan ik dus verder in de boom klimmen vanuit een bepaald punt. Kan ik ook afdalen?Ja, dat kan. Kijk naar het volgende:

```
henk@boven:~/test/bestanden> l
totaal 332
drwxr-xr-x 2 henk wij   4096 17 mei 15:48 ./
drwxr-xr-x 4 henk wij   4096 28 mei 12:16 ../
-rw-r--r-- 1 henk wij      0 17 mei 15:47 1aap
-rw-r--r-- 1 henk wij      0 17 mei 15:47 2noot
henk@boven:~/test/bestanden>
```

We zien hier twee directories met de namen `.` en `..` . Die heten echt zo (zie *Korte uitleg: Harde en zachte links*). De directory `.` in een directory is die directory zelf. De directory `..` in een directory is de directory waar deze directory in staat, een stapje afgedaald in de boom dus.

```
henk@boven:~/test/bestanden> cd ..
henk@boven:~/test>
```

Zoals je aan de prompt kun zien ben ik een stapje afgedaald in de directory tree.

En dat werkt natuurlijk ook met andere commando's:

```
henk@boven:~/test> cd -
/home/henk/test/bestanden
henk@boven:~/test/bestanden> ls -la ..
totaal 28
drwxr-xr-x  4 henk wij 4096 28 mei 12:16 .
drwxr-xr-x 87 henk wij 4096 15 jun 15:12 ..
drwxr-xr-x  2 henk wij 4096 17 mei 15:48 bestanden
drwxr-xr-x  2 henk wij 4096 18 mei 12:37 fff
-rw-r--r--  1 henk wij    0 26 mrt 10:31 file
-rw-r--r--  1 henk wij  566 13 mrt 11:08 hhh
-rwxr--r--  1 henk wij   49 13 mrt 10:39 script
-rw-r--r--  1 henk wij    3 31 mrt 19:29 spsp
henk@boven:~/test/bestanden>
```

En dat is dus de inhoud van `~/test`.