

I ♥ BASH

Johan Adriaans  
johan.adriaans@gmail.com

# Wat wordt er behandeld

- Waarom?
- Hello world?
- Rechtensysteem
- Shell expansion
- Control structures
- History
- Input/output redirection
- Powertools

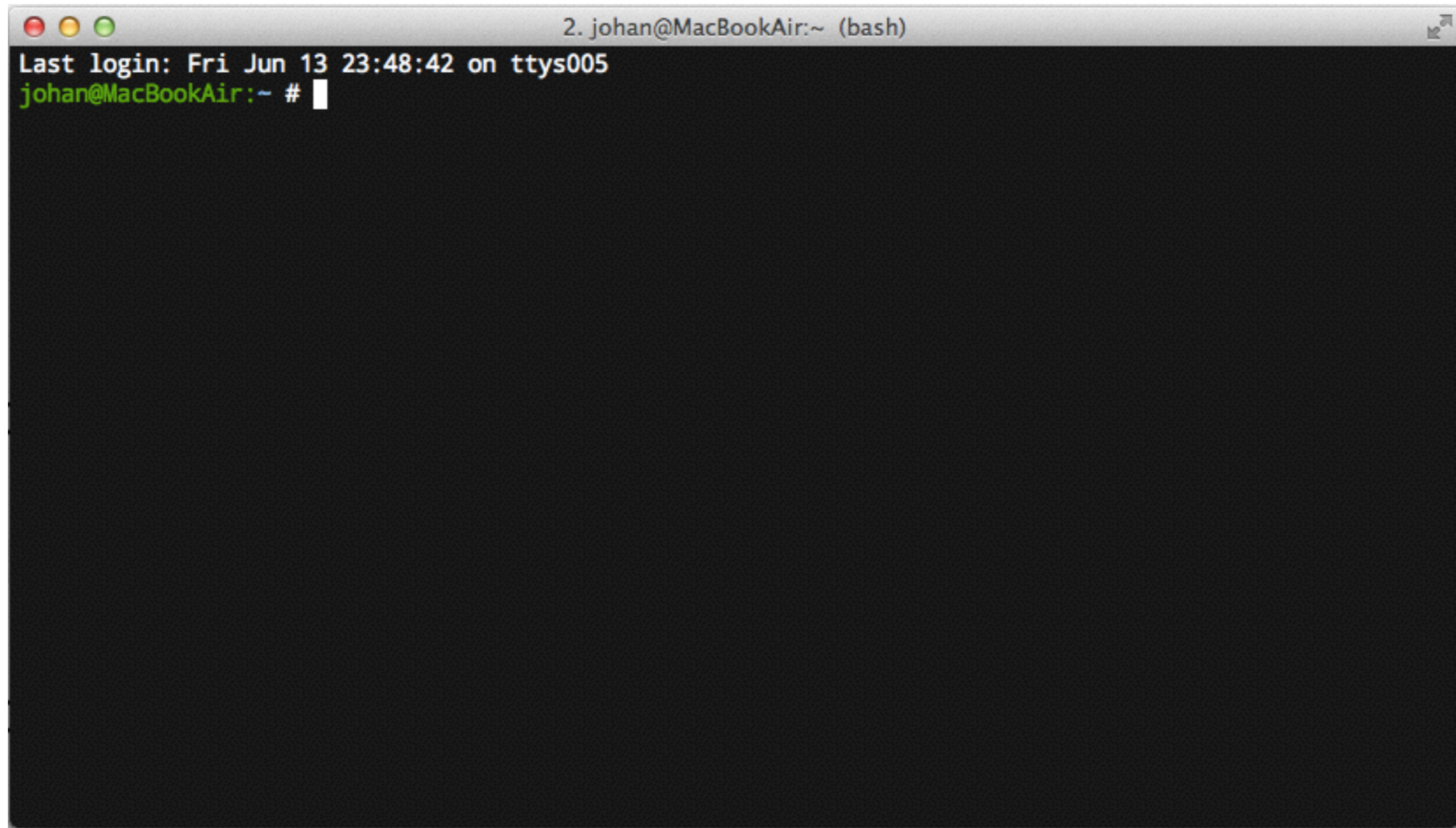
# Waarom BASH?

- BASH wordt overal standaard bijgeleverd (Behalve bij MS Windows)
- Een van de weinige dingen waar Android en iOS het over eens zijn
- Bash openen is “onder de motorkap” van je computer kruipen
- Snel door het filesystem navigeren (Vooral verborgen-/systeem-mappen)
- Supersnel kleine taken automatiseren

# Supersnel kleine taken automatiseren?

- Top 20 ipadressen uit een apache log  
`cat access_log | cut -d " " -f 2 | sort | uniq -c | sort -h | tail -n 20`
- Verwijder alle torrents met een ratio van 2 uit transmission  
`transmission-remote -t `transmission-remote -l | awk '{print $8 " " $1}' | grep '^2' | awk '{print $2}' | tr "\n" "," | sed 's/,,$//` -remove`
- Kopieer een mysql database van server naar server  
`mysqldump -h remote.server.com database | mysql database`
- Maak een mysql database backup en mail deze als attachment  
`mysqldump database | gzip -c > backup.gz && echo "Mail body" | mutt -s "Backup demo" -backup.gz -- johan@izi-services.nl && rm backup.gz`
- Zoek en verwijder een regel tekst uit meerdere bestanden in 1x  
`locate SEO/Events.php | xargs sed -i '/http:\\/\\/search.yahooapis.com\\/SiteExplorer/d'`

# Hello world?



```
2. johan@MacBookAir:~ (bash)
Last login: Fri Jun 13 23:48:42 on ttys005
johan@MacBookAir:~ #
```

- BASH is een shell, net als windows
- Bash (Bourne-again shell) is de free-software variant van de Bourne shell (1977) en is in 1989 'geboren'
- Maakt gebruik van de POSIX standaard
- De Bourne Shell is ooit bedacht als programmeertaal en heeft dus ook control structures en variabelen
- BASH gebruikt 'readline' als input library, deze lib zorgt o.a. voor de tab-completion en history functionaliteiten

# Snelle opfrisser

- `ls -al` (list) toont de inhoud van de huidige directory
- `cd /pad/naar/folder` (change directory)
- `pwd` (print working directory)
- `cp` (copy)
- `rm` (remove)
- `mv` (move / rename)
- `cat` (concatenate) Bekijk de inhoud van een bestand
- `man` (manual)
- `ncal -wy` (Weeknummers van dit jaar)

# TAB

+

pijltjestoetsen (of ctrl-p en ctrl-n)



# Rechtensysteem

```
drwxr-xr-x  5 johan  staff    170 May 18 00:06 iMacros
-rw-r--r--  1 johan  staff    221 Jul 31 2013 ipv6.text
drwx----- 4 johan  staff    136 Mar 15 22:37 mail
-rw-r--r--  1 johan  staff   3730 Sep  4 2013 sdr-fmems.sdm
```

---

Owner	Group	Other
rwX	rwX	rwX

---

```
chmod a+r
chmod u+w
chmod g-x
chmod o-w
chmod 755
```

# Superkorte introductie tot grondgetallen

## **Basisregel**

**getal \* (grondgetal ^ positie van rechts)**

Decimaal: 5692

$$5*(10^3) + 6*(10^2) + 9*(10^1) + 2*(10^0)$$
$$5*1000 + 6*100 + 9*10 + 2*1$$

Hexadecimaal: 57E8

$$5*(16^3) + 7*(16^2) + 15*(16^1) + 8*(16^0)$$
$$5*4096 + 7*256 + 15*16 + 8*1$$

Binair: 1101

$$1*(2^3) + 1*(2^2) + 0*(2^1) + 1*(2^0)$$

# Shell Expansion

- `*` = Alle bestanden en folders in de huidige folder
- `*.ext` = Alle bestanden die deze expressie matchen
- `{a..z}{0..9}` = a1 a2 a3 ..... z7 z8 z9
- `/pad/naar/{file1,file2}` == `/pad/naar/file{1,2}`
- `~` `~-` `~+` `~user`
- `$(command)` == ``command``
- `echo $((2 + 2))`
- `VARIABLE=test; echo $VARIABLE; echo ${VARIABLE:=foo}`

```
machine:~ user# echo *
Applications Desktop Documents Dotfiles Downloads...
```

```
machine:~ user# echo Do*
Documents Dotfiles Downloads
```

```
machine:~ user# echo {a..z}
a b c d e f g h i j k l m n o p q r s t u v w x y z
```

```
machine:~ user# cp bestand{,.bak}
wordt: cp bestand bestand.bak
```

```
machine:~ user# sudo ls /home/*/Maildir/new
machine:~ user# sudo ls /home/user{1,2,3}/Maildir/new
```

```
machine:~ user# mkdir {2012..2014}-{1..12} && ls
2012-1 2012-2 2012-3 2012-4 2012-5 2012-6 2012-7 2012-8...
```

```
ls -la `whereis ls`
-rwxr-xr-x 1 root wheel 34736 Oct 29 2013 /bin/ls
```

Shell expansion vindt plaats vóór  
het commando wordt uitgevoerd

```
machine:~ user# mkdir tmp && cd tmp
machine:tmp user# >cp
machine:tmp user# >foobar
machine:tmp user# * test
machine:tmp user# ls
cp foobar test
```

## Let op bij quotes!

```
machine:~ user# echo ls ${foo:=*}
ls Applications Desktop Documents Dotfiles Downloads...
```

```
machine:~ user# echo "ls ${foo:=*}"
ls *
```

```
machine:~ user# echo 'ls ${foo:=*}'
ls ${foo:=*}
```

```
machine:~ user# echo 'ls ${foo:=\\*}'
ls ${foo:=\\*}
```

# Control structures

- `cat nonexistentfile && echo 123`
- `cat nonexistentfile || echo 123`
- `if [ -f /etc/passwd ]; then echo 123; fi`
- `[ -f /etc/passwd ] && echo 123`
- `test -f /etc/passwd && echo 123`
- `for i in $(ls); do echo item: $i; done`
- `while [ ${C:=0} -lt 10 ]; do echo $C; let C=C+1; done`

# History

- het 'history' commando zelf
- !123 — !! — !-2
- !begintmet — !?bevat
- !begintmet:0 — !?bevat:1 — !!:^ — !!:\$ — !!:\* — !!:2-\$
- :p = print commando zonder het uit te voeren
- :h - t - r - e modifiers om een pad aan te passen
- !!:s/search/replace/ — !!:gs/search\_global/replace/
- ^search^replace
- fc = fix command - Opent editor met daarin laatste commando



Zoek een commando  
in de history lijst

**command**

**!321**  
*Commando nummer 321  
uit de history database*

**!!**  
*Het vorige commando*

**!vi**  
*Een commando dat  
begint met: 'vi'*

**!?vi**  
*Een commando dat  
'vi' bevat*

Selecteer 1 of meerdere  
'woorden' uit dit commando

**word  
designator**

**3**  
*3e woord*

**2-5**  
*woord 2 tot en met 5*

**\$**  
*laatste woord*

**^**  
*eerste woord*

**\***  
*alle woorden  
excl. commando*

**0**  
*Het programma zelf*

Pas een modifier  
toe op deze woorden

**modifier**

**p**  
*print output*

**(g)s**  
*s/search/replace/*

**h**  
*verwijder pad*

**t**  
*verwijder filename*

**e**  
*verwijder alles behalve  
de extensie*

**r**  
*verwijder de extensie*

```
machine:~ user# vi /pad/naar/een/file.txt
opent de vi text-editor
```

```
machine:~ user# ls -la !!:1
ls -al /pad/naar/een/file.txt
```

```
machine:~ user# ls -la !vi:1
ls -al /pad/naar/een/file.txt
```

```
machine:~ user# ^vi^ls -al
ls -al /pad/naar/een/file.txt
```

```
machine:~ user# cd !vi:1:h
output: cd /pad/naar/een
```

```
machine:~ user# cd !vi:1:h:s/een/foobar/
output: cd /pad/naar/foobar
```

```
machine:~ user# ctrl-r
(reverse-i-search)`: vi
```

# Input/output redirection

- Het koppelen en herleiden van datastromen
- STDIN, STDOUT, STDERR
- Eh.. file descriptors?
- Voorbeelden met plaatjes!

## STDIN (fd0)

```
machine:~ user# Alles wat je in de terminal typt is STDIN
```

## STDOUT (fd1)

```
machine:~ user# ls -al
```

```
total 8
```

```
drwxr-xr-x   4 johan  staff   136 Jun 22 13:08 .  
drwxr-xr-x+ 67 johan  staff  2278 Jun 22 15:20 ..  
-rw-r--r--   1 johan  staff     0 Jun 22 13:08 foo  
-rw-r--r--   1 johan  staff   220 May 25 14:54 bar
```

## STDERR (fd2)

```
machine:~ user# ls -al nietbestaandefolder
```

```
ls: nietbestaandefolder: No such file or directory
```

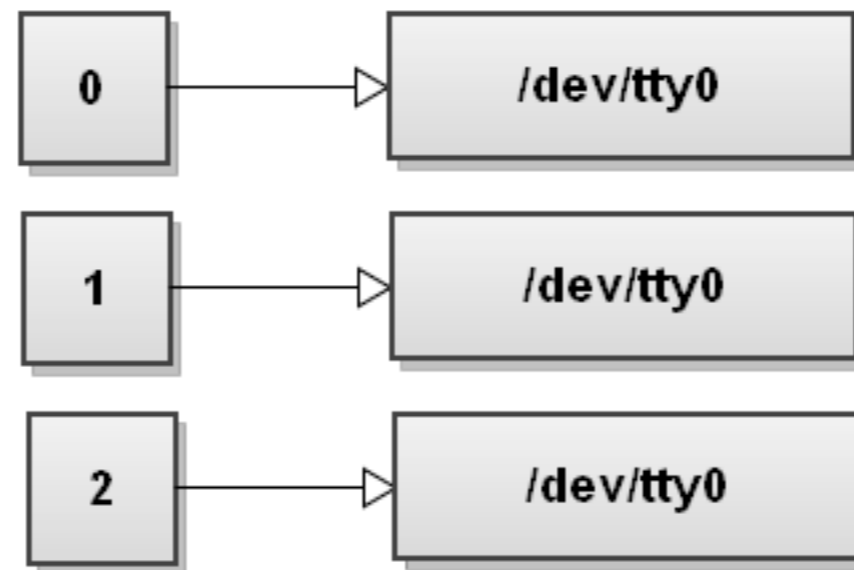
# File descriptors

- Iedere file bewerking gaat via de kernel
- De kernel controleert of alles in orde is en geeft een virtuele toegang tot dit bestand
- Alle geopende bestanden door alle processen worden bijgehouden door de kernel (Zoals vastgelegd in POSIX)
- Deze kun je inzien met het commando: `ls -l /proc/self/fd/`
- Het ID van deze virtuele toegang heet een file descriptor
- (Dit geldt ook voor netwerkverkeer)

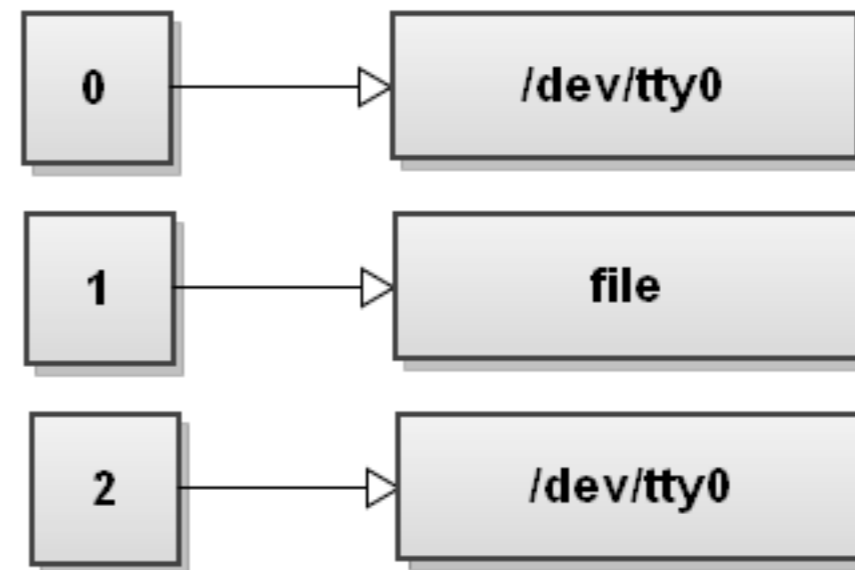
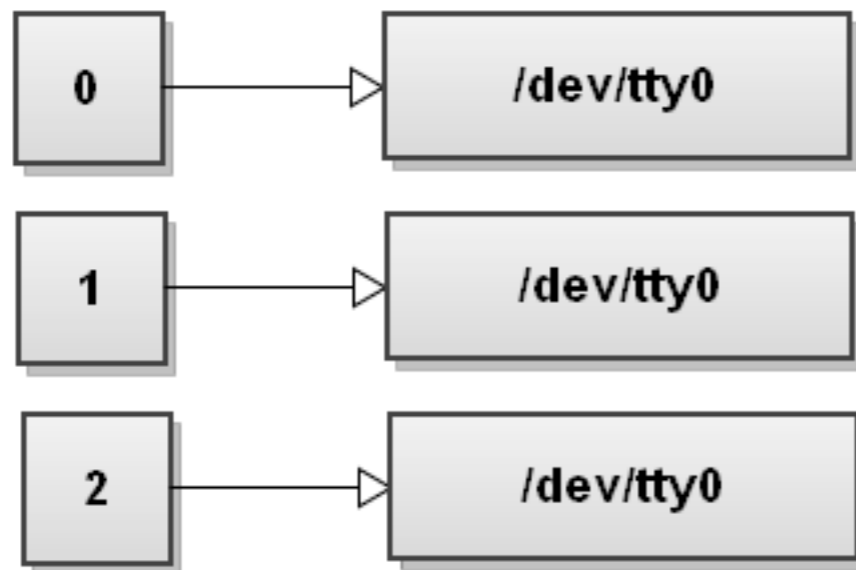
```
machine:~ user# lsof -p $$
```

COMMAND	PID	USER	FD	TYPE	DEVICE	SIZE/OFF	NODE	NAME
bash	74018	johan	cwd	DIR	1,2	136	6985983	/Users/johan
bash	74018	johan	txt	REG	1,2	1228240	3175305	/bin/bash
bash	74018	johan	<u>0u</u>	CHR	16,6	0t988756	791	<u>/dev/tty0</u>
bash	74018	johan	<u>1u</u>	CHR	16,6	0t988756	791	<u>/dev/tty0</u>
bash	74018	johan	<u>2u</u>	CHR	16,6	0t988756	791	<u>/dev/tty0</u>

# Normale situatie



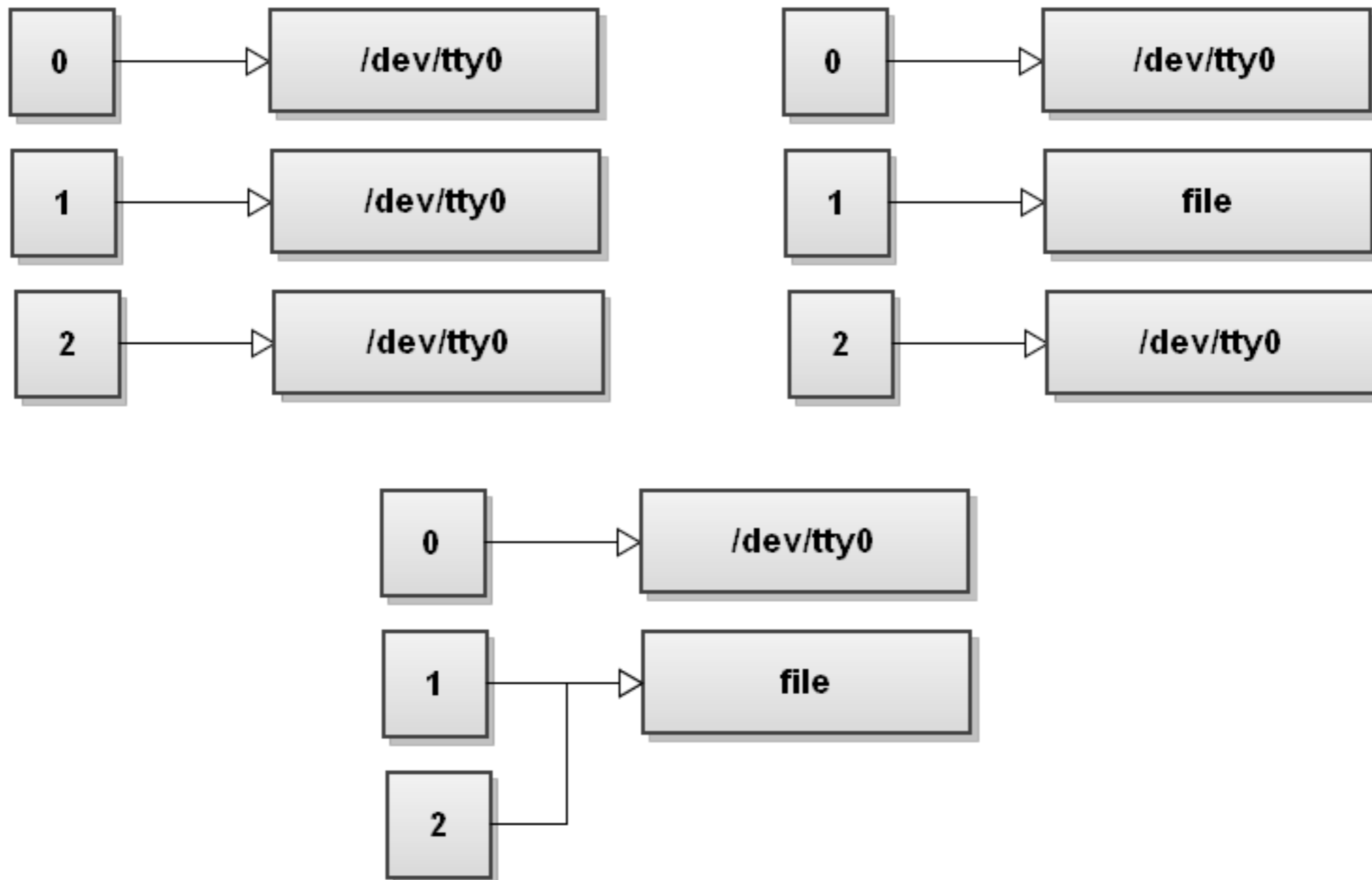
# \$ command >file





# \$ command &>file

\$ command >file 2>&1



```
machine:~ user# ls -la >file.txt 2>&1
```

Redirect alle output van "ls -al" naar file.txt

```
machine:~ user# ls -la &>/dev/null
```

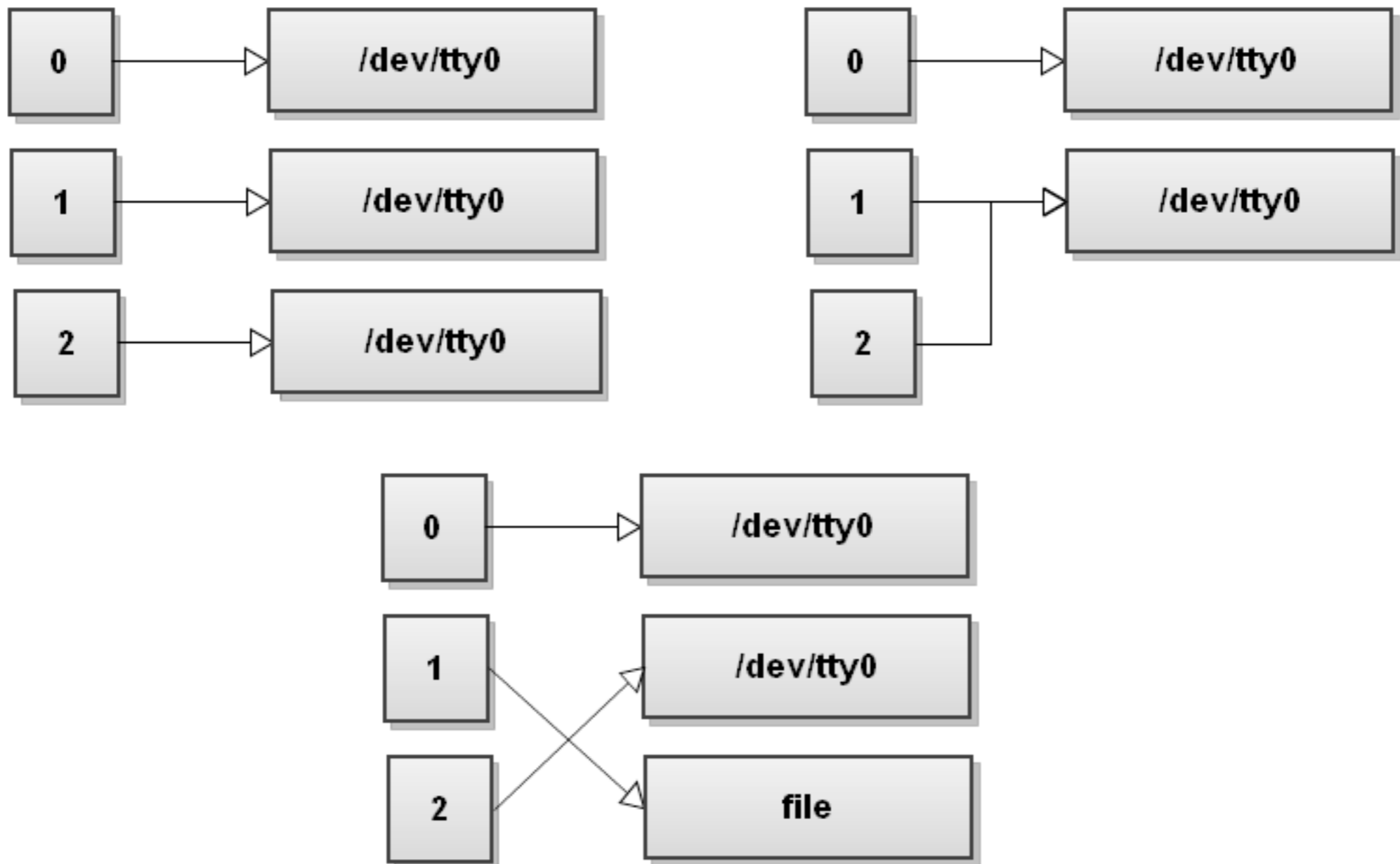
Redirect alle output naar /dev/null ('prullenbak' file)

```
machine:~ user# ls -la 2>/dev/null
```

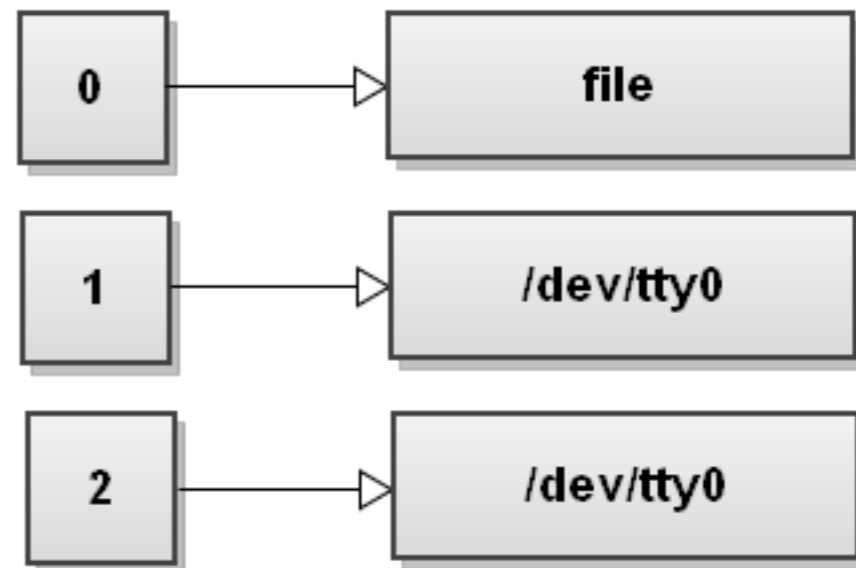
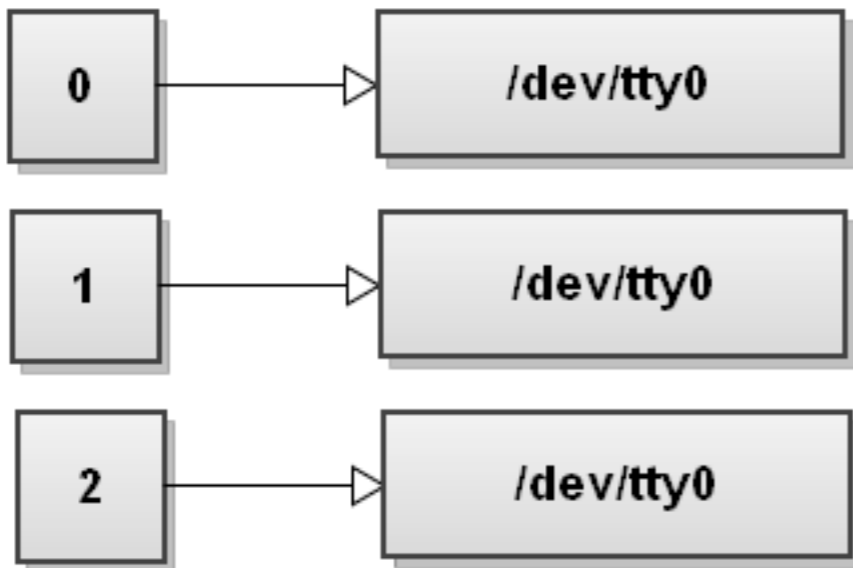
Redirect alleen de errors naar /dev/null

# volgorde belangrijk?

```
$ command 2>&1 >file
```



# \$ command <file



```
machine:~ user# mysql database <file.sql
```

Koppelt de file descriptor van file.sql aan STDIN van het commando

```
machine:~ user# mysql database <<<"SELECT * FROM TABLE"
```

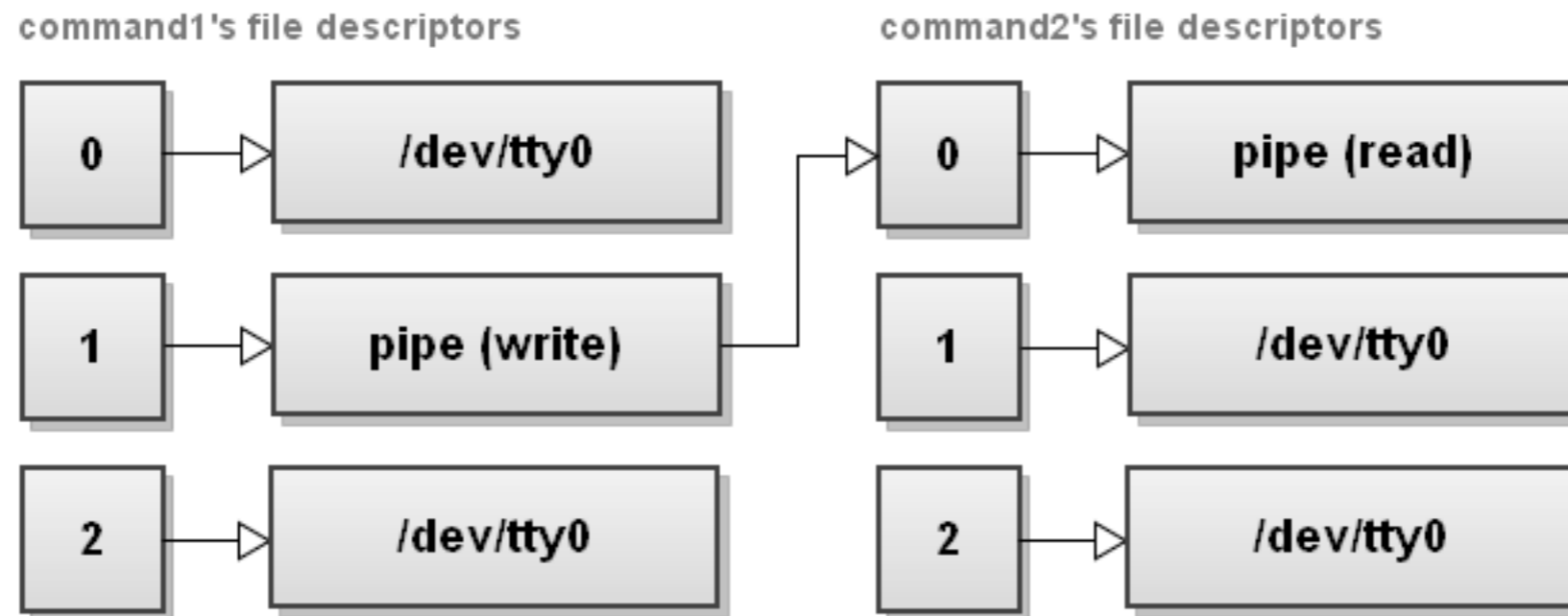
Redirect een stuk tekst naar de STDIN van een commando

```
machine:~ user# mysql database < <(sql_generator)
```

Koppel STDOUT van het commando sql\_generator aan de STDIN van mysql

# \$ command1 | command2

\$ command1 > >(command2)



```
machine:~ user# grep user /var/log/auth.log | less
```

Maak de output leesbaarder met het programma less

```
machine:~ user# cat foo.ini | awk '{print $1}' | sort | uniq
```

Print output van foo.ini, toon eerste kolom, sorteer en ontdubbel

```
machine:~ user# tree ~ | mutt -s "ls output" email@adres.nl
```

Mailt de output van het tree commando naar het opgegeven adres

```
machine:~ user# find . -name "*.sql"
./files/export_v1.sql
./files/export_v2.sql
./files/export_v2_structure.sql
./files/export_test.sql
./backup/export_v2.sql
```

```
machine:~ user# find . -name "*.sql" | grep "_v2"
./files/export_v2.sql
./files/export_v2_structure.sql
./backup/export_v2.sql
```

```
machine:~ user# find . -name "*.sql" | grep "_v2" | xargs tar -zcf
test.tgz
```

```
machine:~ user# tar -t test.tgz | mutt -s "SQL" -a test.tgz email@
domain.com
```

```
machine:~ user# rm test.tgz
```



# Met redirection kun je dus

- Snel korte en krachtige programma's maken
- Snel automatiseren zonder programmeren
- Makkelijk zelf programma's schrijven die je binnen dergelijke structuren kunt gebruiken.

# Powertools

- grep - Toon alleen de regels die matchen met de zoekopdracht
- uniq - Ontdubbelen van regels
- sort - Sorteert de output
- find - Zoek bestanden
- head en tail - Pak de eerste of de laatste regels van een stream
- xargs - Gebruik STDIN als parameters achter het opgegeven commando
- rev - Draai iedere regel om, handig om snel op de laatste kolom of letter te sorteren
- cut - Deel een regel op in kolommen en selecteer welke worden getoond
- awk - Eigen taaltje om tekst te beïnvloeden, gebruik ik vaak in plaats van cut
- sed - Stream editor, wordt veel gebruikt om aanpassingen te maken in meerdere bestanden tegelijkertijd

Einde