

Bases UNIX

Bastien DI PIERRO

Université Claude Bernard Lyon 1

25 Novembre 2014

DON'T PANIC
IT'S NOT WINDOWS



1 Systèmes d'exploitation

2 Bash

3 Droits et utilisateur

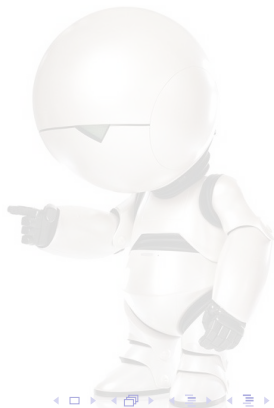
4 Connexion

5 Système de fichiers

6 Processus et mémoire

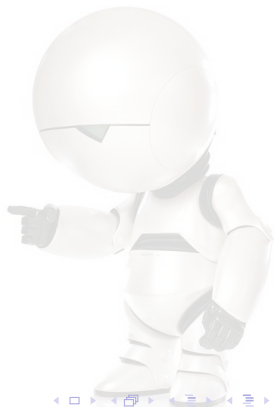
DON'T PANIC

IT'S NOT WINDOWS



- 1 Systèmes d'exploitation
 - qu'est-ce qu'un OS ?
 - Une commande simple
 - Différents systèmes d'exploitation (OS)
 - Historique
- 2 Bash
- 3 Droits et utilisateur
- 4 Connexion
- 5 Système de fichiers
- 6 Processus et mémoire

DON'T PANIC
IT'S NOT WINDOWS



Qu'est-ce qu'un OS ?

- Ordinateur : tas de circuits électroniques qui chauffent et parlent uniquement binaire.

DO IT'S NOT WINDOWS



Qu'est-ce qu'un OS ?

- Ordinateur : tas de circuits électroniques qui chauffent et parlent uniquement binaire.
- Humain : tas de cellules organiques qui parlent tout sauf binaire



DO _ _ _ _ _ C
IT'S NOT WINDOWS



Qu'est-ce qu'un OS ?

- Ordinateur : tas de circuits électroniques qui chauffent et parlent uniquement binaire.
- Humain : tas de cellules organiques qui parlent tout sauf binaire



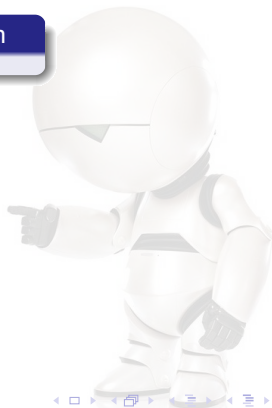
- Système d'exploitation (OS) : fait l'interfaçage entre le Hardware (machine) et l'humain au travers d'un ensemble de programmes.

Une commande simple

```
bastien@bastien-P35-DS3L:~/dossier_test$ ls  
cours.pdf  film.avi  image.jpg  musique.mp3  prgm.c  slide.tex  texte.txt  
bastien@bastien-P35-DS3L:~/dossier_test$
```

Système d'exploitation

DON'T PANIC
IT'S NOT WINDOWS



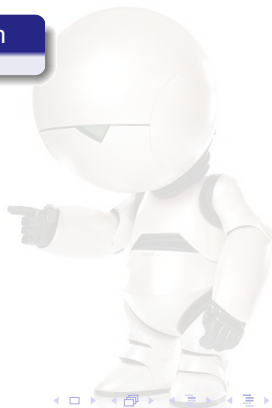
Une commande simple

```
bastien@bastien-P35-DS3L:~/dossier_test$ ls  
cours.pdf  film.avi  image.jpg  musique.mp3  prgm.c  slide.tex  texte.txt  
bastien@bastien-P35-DS3L:~/dossier_test$
```

Système d'exploitation



DON'T PANIC
IT'S NOT WINDOWS



Une commande simple

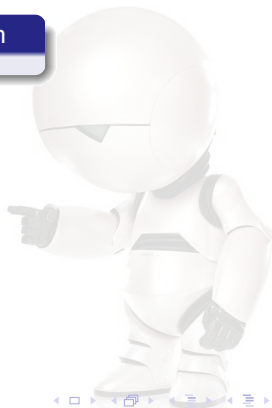
```
bastien@bastien-P35-DS3L:~/dossier_test$ ls  
cours.pdf film.avi image.jpg musique.mp3 prgm.c slide.tex texte.txt  
bastien@bastien-P35-DS3L:~/dossier_test$
```

Système d'exploitation



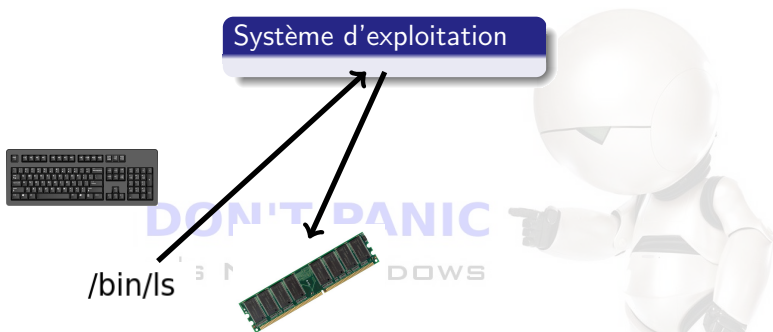
/bin/ls

DON'T PANIC
NOT WINDOWS



Une commande simple

```
bastien@bastien-P35-DS3L:~/dossier_test$ ls  
cours.pdf film.avi image.jpg musique.mp3 prgm.c slide.tex texte.txt  
bastien@bastien-P35-DS3L:~/dossier_test$
```



Une commande simple

```
bastien@bastien-P35-DS3L:~/dossier_test$ ls  
cours.pdf  film.avi  image.jpg  musique.mp3  prgm.c  slide.tex  texte.txt  
bastien@bastien-P35-DS3L:~/dossier_test$
```

Système d'exploitation



/bin/l's



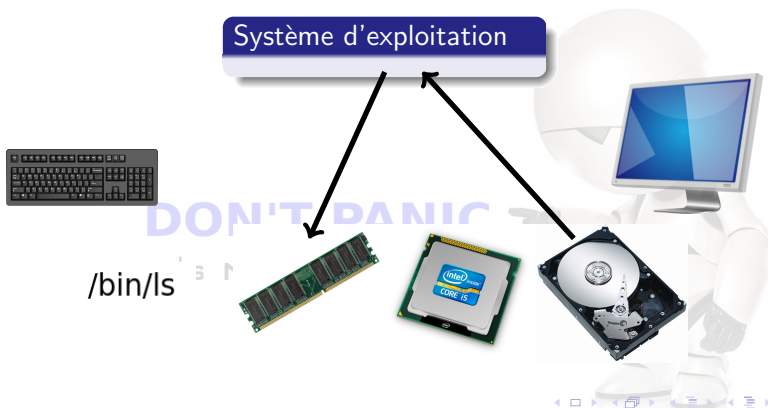
Une commande simple

```
bastien@bastien-P35-DS3L:~/dossier_test$ ls  
cours.pdf  film.avi  image.jpg  musique.mp3  prgm.c  slide.tex  texte.txt  
bastien@bastien-P35-DS3L:~/dossier_test$
```



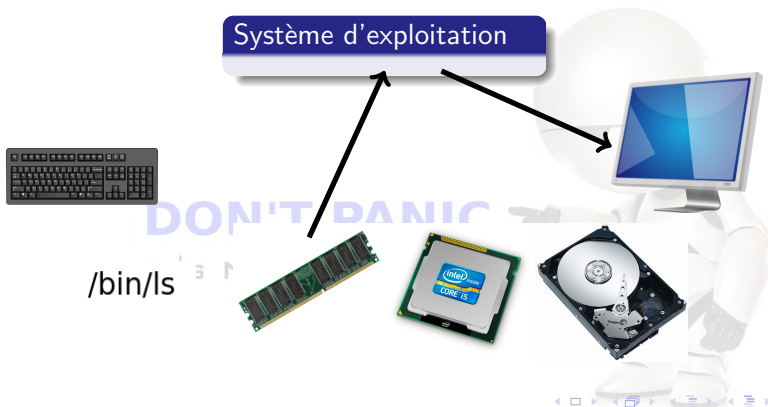
Une commande simple

```
bastien@bastien-P35-DS3L:~/dossier_test$ ls  
cours.pdf  film.avi  image.jpg  musique.mp3  prgm.c  slide.tex  texte.txt  
bastien@bastien-P35-DS3L:~/dossier_test$
```



Une commande simple

```
bastien@bastien-P35-DS3L:~/dossier_test$ ls  
cours.pdf  film.avi  image.jpg  musique.mp3  prgm.c  slide.tex  texte.txt  
bastien@bastien-P35-DS3L:~/dossier_test$
```

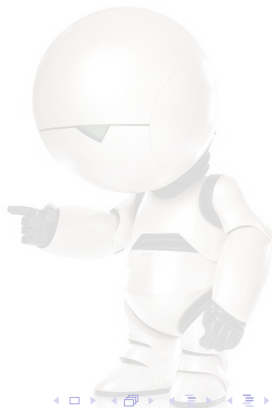


Différents systèmes d'exploitation (OS)

- Windows (Microsoft)



DON'T PANIC
IT'S NOT WINDOWS

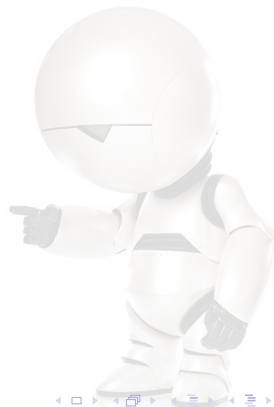


Différents systèmes d'exploitation (OS)

- Windows (Microsoft)
- Mac OS (Apple)



DON'T PANIC
NOT WINDOWS



Différents systèmes d'exploitation (OS)

- Windows (Microsoft)
- Mac OS (Apple)
- UNIX (Uniplexed Information and Computing Service)



DO NOT WINDOWS



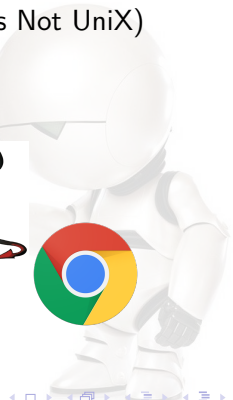
Différents systèmes d'exploitation (OS)

- Windows (Microsoft)
- Mac OS (Apple)
- UNIX (Uniplexed Information and Computing Service)
- GNU LINUX (GNU's Not Unix, Linux Is Not UniX)



Différents systèmes d'exploitation (OS)

- Windows (Microsoft)
- Mac OS (Apple)
- UNIX (Uniplexed Information and Computing Service)
- GNU LINUX (GNU's Not Unix, Linux Is Not UniX)
- Chrome OS (Google)



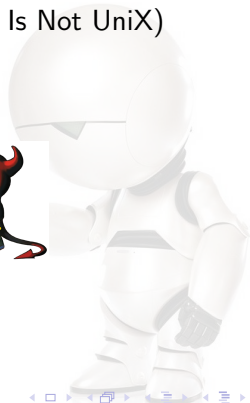
Différents systèmes d'exploitation (OS)

- Windows (Microsoft)
- Mac OS (Apple)
- UNIX (Uniplexed Information and Computing Service)
- GNU LINUX (GNU's Not Unix, Linux Is Not UniX)
- Chrome OS (Google)
- Android (Google)



Différents systèmes d'exploitation (OS)

- UNIX (Uniplexed Information and Computing Service)
- GNU LINUX (GNU's Not Unix, Linux Is Not UniX)

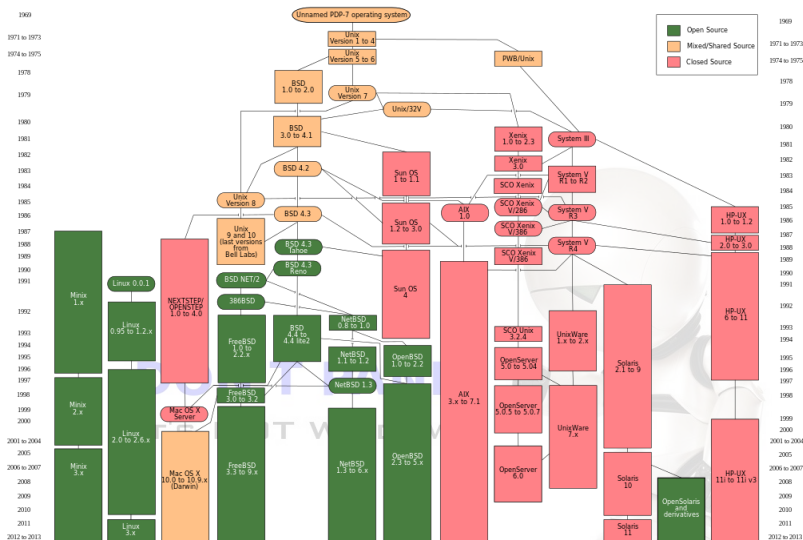


Historique

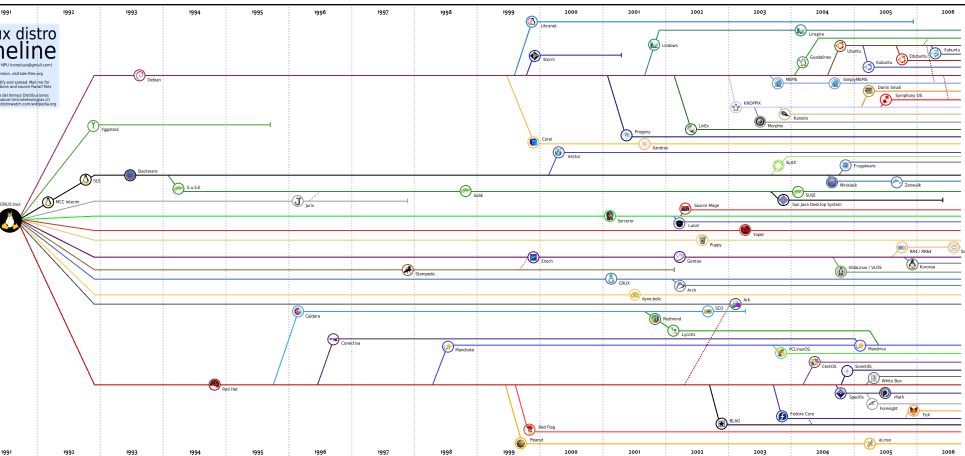
- Unix, créé en 1969-70 par Ken Thompson, Dennis Ritchie, Brian Kernigham (créateurs du langage C) ...
⇒ Conçu pour le calcul scientifique
- Apple (Steeve Jobs) : LisaOS 1983, MacOS 1984 (Noyau UNIX)
⇒ Rendre l'ordinateur accessible au grand public
- Microsoft : Windows 1.0 (1985, Chase Bishop et Bill Gates) (Idée "volée" à Apple)
- Linux, créé en 1991 par Linux Torvalds
⇒ Créer un système entièrement compatible avec UNIX et Gratuit !
- Google Android, 2008 (Basé sur un Noyau Linux)

UNIX : Père de tous les systèmes d'exploitations !

Historique



Historique



1 Systèmes d'exploitation

2 Bash

- Qu'est-ce qu'un Shell ?
- Différents types de shell
- A quoi ça ressemble ?
- Les commandes du bash

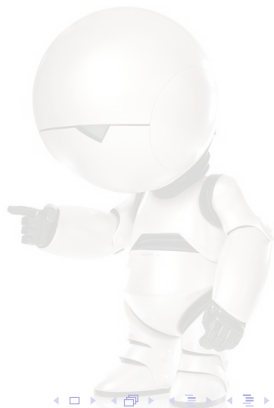
3 Droits et utilisateur

4 Connexion

5 Système de fichiers

6 Processus et mémoire

DON'T PANIC
IT'S NOT WINDOWS



Qu'est-ce qu'un Shell ?

- Shell (sh) = Interpreteur de lignes de commandes
- Interface de communication Machine/Utilisateur en mode texte
(Initialement la seule interface Homme/Machine)
- Exécute des actions élémentaires en fonction des commandes tapées au clavier.
- Peut être considéré comme un langage de programmation :
possibilité d'utiliser des variables, des fonctions, des conditions
...

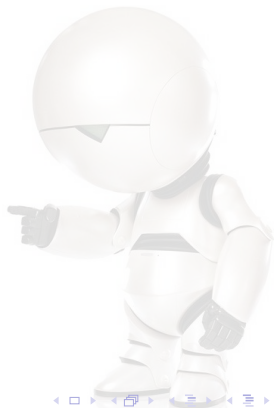
Puissance du shell

- Possibilité de gérer et administrer une machine uniquement avec le shell
- Plus rapide et plus efficace que les interfaces graphiques
⇒ Vous ne pourrez plus vous en passer !

Différents types de shell

- Le shell sh : le premier Shell sur les systèmes UNIX (Stephen Bourne)

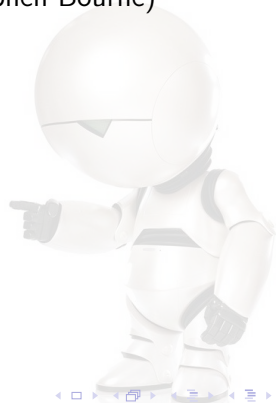
DON'T PANIC
IT'S NOT WINDOWS



Différents types de shell

- Le shell sh : le premier Shell sur les systèmes UNIX (Stephen Bourne)
- Le bash : Bourne-Again SHell (Stephen Bourne)

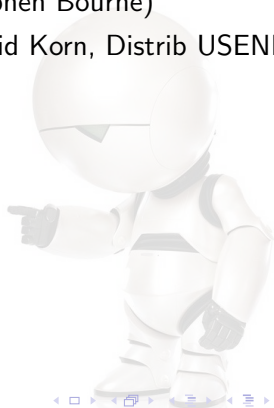
DON'T PANIC
IT'S NOT WINDOWS



Différents types de shell

- Le shell sh : le premier Shell sur les systèmes UNIX (Stephen Bourne)
- Le bash : Bourne-Again SHell (Stephen Bourne)
- Le ksh (ou tksh) : Korn SHell (David Korn, Distrib USENIX)

DON'T PANIC
IT'S NOT WINDOWS



Différents types de shell

- Le shell sh : le premier Shell sur les systèmes UNIX (Stephen Bourne)
- Le bash : Bourne-Again SHell (Stephen Bourne)
- Le ksh (ou tksh) : Korn SHell (David Korn, Distrib USENIX)
- Le csh (ou tcsh) : C SHell (Bill Joy, Distrib. BSD)

DON'T PANIC
IT'S NOT WINDOWS



Différents types de shell

- Le shell sh : le premier Shell sur les systèmes UNIX (Stephen Bourne)
- Le bash : Bourne-Again SHell (Stephen Bourne)
- Le ksh (ou tksh) : Korn SHell (David Korn, Distrib USENIX)
- Le csh (ou tcsh) : C SHell (Bill Joy, Distrib. BSD)

Choix : bash

- Plus complet que sh
- Le plus utilisé
- Shell par défaut sur la plupart des distributions

A quoi ça ressemble ?

```
bastien@bastien-P35-DS3L: ~/dossier_test
bastien@bastien-P35-DS3L:~/dossier_test$ ls
cours.pdf  film.avi  image.jpg  musique.mp3  prgm.c  slide.tex  texte.txt
bastien@bastien-P35-DS3L:~/dossier_test$
```

Annotations:

- commande (points to `ls`)
- Attente d'une commande (prompt) (points to the prompt `bastien@bastien-P35-DS3L:~/dossier_test$`)
- Dossier courant (points to `~/dossier_test`)
- Nom de la machine (points to `bastien`)
- Nom de l'utilisateur (points to `P35-DS3L`)
- Résultat de la commande (points to the list of files)

Les commandes de bases

- **ls** : liste les fichiers et dossiers dans le répertoire courant.
(Option : **ls -a** : inclus les éléments cachés; **ls -l** : liste détaillée avec vos droits, le propriétaire du fichier, le groupe du fichier, la date de création ; **ls -t** : affiche par ordre croissant de date de modification ...)

```
bastien@bastien-P35-DS3L:~/dossier_test$ ls -lrt
total 4
-rw-rw-r-- 1 bastien bastien  0 nov.  17 15:33 image.jpg
-rw-rw-r-- 1 bastien bastien  0 nov.  17 15:33 musique.mp3
-rw-rw-r-- 1 bastien bastien  0 nov.  17 15:33 film.avi
-rw-rw-r-- 1 bastien bastien  0 nov.  17 15:33 texte.txt
-rw-rw-r-- 1 bastien bastien  0 nov.  17 15:34 prgm.c
-rw-rw-r-- 1 bastien bastien  0 nov.  17 15:34 slide.tex
-rw-rw-r-- 1 bastien bastien  0 nov.  17 15:34 cours.pdf
-rw-rw-r-- 1 bastien bastien 20 nov.  18 11:23 texte
bastien@bastien-P35-DS3L:~/dossier_test$
```

IT'S NOT WINDOWS

Les commandes de bases

- `ls` : liste les fichiers et dossiers dans le répertoire courant.
(Option : `ls -a` : inclus les éléments cachés; `ls -l` : liste détaillée avec vos droits, le propriétaire du fichier, le groupe du fichier, la date de création ; `ls -t` : affiche par ordre croissant de date de modification ...)
- `pwd` : indique dans quel répertoire on se trouve

```
bastien@bastien-P35-DS3L:~/dossier_test$ pwd  
/home/bastien/dossier_test  
bastien@bastien-P35-DS3L:~/dossier_test$ █
```

DON'T PANIC
IT'S NOT WINDOWS



Les commandes de bases

- `ls` : liste les fichiers et dossiers dans le répertoire courant.
(Option : `ls -a` : inclus les éléments cachés; `ls -l` : liste détaillée avec vos droits, le propriétaire du fichier, le groupe du fichier, la date de création ; `ls -t` : affiche par ordre croissant de date de modification ...)
- `pwd` : indique dans quel répertoire on se trouve
- `cd` (change directory) : se déplace dans les différents répertoires

```
bastien@bastien-P35-DS3L:~$ cd dossier_test/  
bastien@bastien-P35-DS3L:~/dossier_test$
```

DON'T PANIC
IT'S NOT WINDOWS



Les commandes de bases

- `ls` : liste les fichiers et dossiers dans le répertoire courant.
(Option : `ls -a` : inclus les éléments cachés; `ls -l` : liste détaillée avec vos droits, le propriétaire du fichier, le groupe du fichier, la date de création ; `ls -t` : affiche par ordre croissant de date de modification ...)
- `pwd` : indique dans quel répertoire on se trouve
- `cd` (change directory) : se déplace dans les différents répertoires
- `su` : change d'utilisateur
(Exemple : `su gaston` : on devient l'utilisateur gaston
Attention : sans argument : on devient l'utilisateur root)

Les commandes de bases

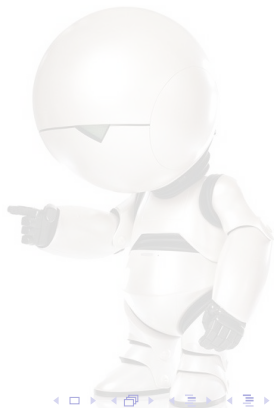
- **ls** : liste les fichiers et dossiers dans le répertoire courant.
(Option : **ls -a** : inclus les éléments cachés; **ls -l** : liste détaillée avec vos droits, le propriétaire du fichier, le groupe du fichier, la date de création ; **ls -t** : affiche par ordre croissant de date de modification ...)
- **pwd** : indique dans quel répertoire on se trouve
- **cd** (change directory) : se déplace dans les différents répertoires
- **su** : change d'utilisateur
(Exemple : **su gaston** : on devient l'utilisateur gaston
Attention : sans argument : on devient l'utilisateur root)
- **man** : affiche la documentation de n'importe quelle commande ! (Exemple : **man ls**)
⇒ LA commande à ne jamais oublier !

La gestion des fichiers

- **mkdir** : créé un dossier.

```
bastien@bastien-P35-DS3L:~/dossier_test$ ls
cours.pdf  film.avi  image.jpg  musique.mp3  prgm.c  slide.tex  texte  texte.txt
bastien@bastien-P35-DS3L:~/dossier_test$ mkdir toto
bastien@bastien-P35-DS3L:~/dossier_test$ ls
cours.pdf  film.avi  image.jpg  musique.mp3  prgm.c  slide.tex  texte  texte.txt  toto
bastien@bastien-P35-DS3L:~/dossier_test$
```

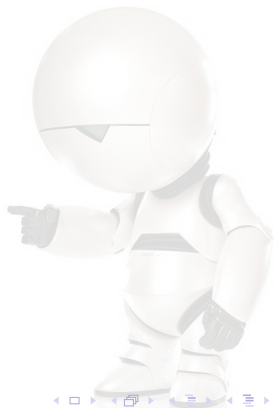
DON'T PANIC
IT'S NOT WINDOWS



La gestion des fichiers

- mkdir : créé un dossier.
- rmdir : supprime un dossier (s'il est vide), rm pour un fichier.

DON'T PANIC
IT'S NOT WINDOWS



La gestion des fichiers

- mkdir : créé un dossier.
- rmdir : supprime un dossier (s'il est vide), rm pour un fichier.
- cat : affiche le contenu d'un fichier

```
bastien@bastien-P35-DS3L:~/dossier_test$ cat prgm.c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char* argv[])
{
    printf("Hello world!\n");
    return 0;
}
bastien@bastien-P35-DS3L:~/dossier_test$
```

DON'T PANIC
IT'S NOT WINDOWS



La gestion des fichiers

- mkdir : créé un dossier.
- rmdir : supprime un dossier (s'il est vide), rm pour un fichier.
- cat : affiche le contenu d'un fichier
- head : affiche les premières lignes d'un fichier
- tail : affiche les dernières lignes d'un fichier

DON'T PANIC
IT'S NOT WINDOWS



La gestion des fichiers

- mkdir : créé un dossier.
- rmdir : supprime un dossier (s'il est vide), rm pour un fichier.
- cat : affiche le contenu d'un fichier
- head : affiche les premières lignes d'un fichier
- tail : affiche les dernières lignes d'un fichier
- cp : copie un fichier

Exemple : `cp prgm.c hello.c` : copie prgm.c dans hello.c

DON'T PANIC
IT'S NOT WINDOWS



La gestion des fichiers

- mkdir : créé un dossier.
- rmdir : supprime un dossier (s'il est vide), rm pour un fichier.
- cat : affiche le contenu d'un fichier
- head : affiche les premières lignes d'un fichier
- tail : affiche les dernières lignes d'un fichier
- cp : copie un fichier
- **Exemple : cp prgm.c hello.c** : copie prgm.c dans hello.c
- grep : recherche une chaîne de caractères dans un fichier.

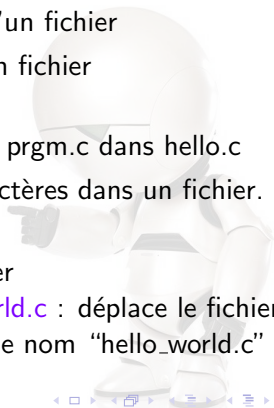
```
bastien@bastien-P35-DS3L:~/dossier_test$ grep printf prgm.c
printf("Hello world!\n");
bastien@bastien-P35-DS3L:~/dossier_test$
```

La gestion des fichiers

- `mkdir` : créé un dossier.
- `rmdir` : supprime un dossier (s'il est vide), `rm` pour un fichier.
- `cat` : affiche le contenu d'un fichier
- `head` : affiche les premières lignes d'un fichier
- `tail` : affiche les dernières lignes d'un fichier
- `cp` : copie un fichier
Exemple : `cp prgm.c hello.c` : copie `prgm.c` dans `hello.c`
- `grep` : recherche une chaîne de caractères dans un fichier.

- `mv` : déplace un fichier ou un dossier
Exemple : `mv hello.c toto/hello_world.c` : déplace le fichier `hello.c` dans le répertoire `toto` avec le nom "`hello_world.c`"

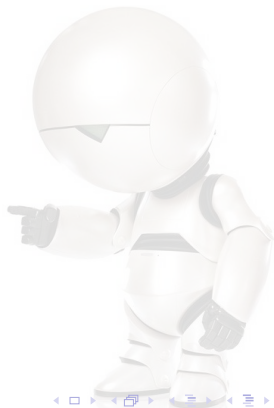
DON'T PANIC



Les caractères spéciaux

- / : séparateur de dossiers. Exemple :
`/home/bastien/dossier_test/`

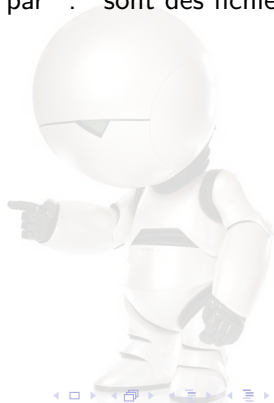
DON'T PANIC
IT'S NOT WINDOWS



Les caractères spéciaux

- / : séparateur de dossiers. Exemple :
/home/bastien/dossier_test/
- . : dossier courant, permet d'exécuter des programmes.
Exemple : ./prgm
(Attention, les fichiers commençant par "." sont des fichiers cachés)

DON'T PANIC
IT'S NOT WINDOWS



Les caractères spéciaux

- / : séparateur de dossiers. Exemple :
/home/bastien/dossier_test/
- . : dossier courant, permet d'exécuter des programmes.
Exemple : ./prgm
(Attention, les fichiers commençant par "." sont des fichiers cachés)
- .. : répertoire parent. Exemple : cd .. → va dans le dossier parent

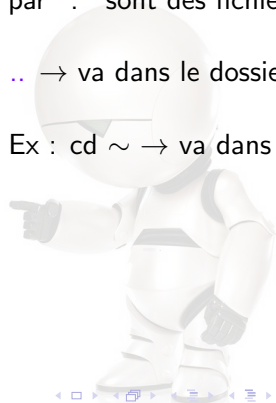
DON'T PANIC
IT'S NOT WINDOWS



Les caractères spéciaux

- / : séparateur de dossiers. Exemple :
/home/bastien/dossier_test/
- . : dossier courant, permet d'exécuter des programmes.
Exemple : ./prgm
(Attention, les fichiers commençant par "." sont des fichiers cachés)
- .. : répertoire parent. Exemple : cd .. → va dans le dossier parent
- ~ : dossier "home" de l'utilisateur. Ex : cd ~ → va dans le dossier /home/bastien/

DON'T PANIC
IT'S NOT WINDOWS



Les caractères spéciaux

- / : séparateur de dossiers. Exemple :
`/home/bastien/dossier_test/`
- . : dossier courant, permet d'exécuter des programmes.
Exemple : `./prgm`
(**Attention**, les fichiers commençant par "." sont des fichiers cachés)
- .. : répertoire parent. Exemple : `cd ..` → va dans le dossier parent
- ~ : dossier "home" de l'utilisateur. Ex : `cd ~` → va dans le dossier `/home/bastien/`
- * : représente n'importe quelle chaîne de caractère (ou rien)

```
bastien@bastien-P35-DS3L:~/dossier_test$ ls *
cours.pdf  film.avi  image.jpg  musique.mp3  prgm.c  slide.tex  texte  texte.txt
bastien@bastien-P35-DS3L:~/dossier_test$ ls *.c
prgm.c
bastien@bastien-P35-DS3L:~/dossier_test$ ls film.*
film.avi
bastien@bastien-P35-DS3L:~/dossier_test$
```

Les caractères spéciaux

- / : séparateur de dossiers. Exemple :
`/home/bastien/dossier_test/`
- . : dossier courant, permet d'exécuter des programmes.
Exemple : `./prgm`
(**Attention**, les fichiers commençant par "." sont des fichiers cachés)
- .. : répertoire parent. Exemple : `cd ..` → va dans le dossier parent
- ~ : dossier "home" de l'utilisateur. Ex : `cd ~` → va dans le dossier `/home/bastien/`
- * : représente n'importe quelle chaîne de caractère (ou rien)
- ? : représente n'importe quel caractère (un seul)

```
bastien@bastien-P35-DS3L:~/dossier_test$ ls prgm.*
prgm.c
bastien@bastien-P35-DS3L:~/dossier_test$ ls f???.*
film.avi
bastien@bastien-P35-DS3L:~/dossier_test$
```

Les caractères spéciaux

- / : séparateur de dossiers. Exemple :
/home/bastien/dossier_test/
- . : dossier courant, permet d'exécuter des programmes.
Exemple : ./prgm
(Attention, les fichiers commençant par "." sont des fichiers cachés)
- .. : répertoire parent. Exemple : cd .. → va dans le dossier parent
- ~ : dossier "home" de l'utilisateur. Ex : cd ~ → va dans le dossier /home/bastien/
- * : représente n'importe quelle chaîne de caractère (ou rien)
- ? : représente n'importe quel caractère (un seul)
- & : exécution d'un programme en arrière plan.

Les caractères spéciaux

- / : séparateur de dossiers. Exemple :
`/home/bastien/dossier_test/`
- . : dossier courant, permet d'exécuter des programmes.
Exemple : `./prgm`
(**Attention**, les fichiers commençant par "." sont des fichiers cachés)
- .. : répertoire parent. Exemple : `cd ..` → va dans le dossier parent
- ~ : dossier "home" de l'utilisateur. Ex : `cd ~` → va dans le dossier `/home/bastien/`
- * : représente n'importe quelle chaîne de caractère (ou rien)
- ? : représente n'importe quel caractère (un seul)
- & : exécution d'un programme en arrière plan.
- && : exécute deux commandes à la suite (seulement si la première ne renvoie pas d'erreur).

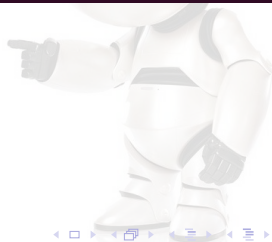
Exemple : `mkdir toto && cd toto`

Pipe et redirection

- `>` : redirige le résultat d'une commande dans un fichier (efface le contenu).

```
bastien@bastien-P35-DS3L:~$ ls dossier_test/* > tutu
bastien@bastien-P35-DS3L:~$ cat tutu
dossier_test/cours.pdf
dossier_test/film.avi
dossier_test/image.jpg
dossier_test/musique.mp3
dossier_test/prgm.c
dossier_test/slide.tex
dossier_test/texte
dossier_test/texte.txt
bastien@bastien-P35-DS3L:~$
```

DON'T PANIC
IT'S NOT WINDOWS

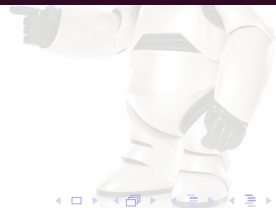


Pipe et redirection

- `>` : redirige le résultat d'une commande dans un fichier (efface le contenu).
- `>>` : redirige le résultat d'une commande à la fin d'un fichier.

```
bastien@bastien-P35-DS3L:~$ grep printf dossier_test/prgm.c >> tutu
bastien@bastien-P35-DS3L:~$ cat tutu
dossier_test/cours.pdf
dossier_test/film.avi
dossier_test/image.jpg
dossier_test/musique.mp3
dossier_test/prgm.c
dossier_test/slide.tex
dossier_test/texte
dossier_test/texte.txt
printf("Hello world!\n");
bastien@bastien-P35-DS3L:~$
```

DON'T PANIC
IT'S NOT WINDOWS



Pipe et redirection

- `>` : redirige le résultat d'une commande dans un fichier (efface le contenu).
- `>>` : redirige le résultat d'une commande à la fin d'un fichier.
- `<` : redirige le contenu d'un fichier dans un programme

Exemple : `./prgm < input.txt`

DON'T PANIC
IT'S NOT WINDOWS



Pipe et redirection

- `>` : redirige le résultat d'une commande dans un fichier (efface le contenu).
- `>>` : redirige le résultat d'une commande à la fin d'un fichier.
- `<` : redirige le contenu d'un fichier dans un programme
Exemple : `./prgm < input.txt`
- `|` (pipeline) : redirige le résultat d'une commande dans une autre commande

```
bastien@bastien-P35-DS3L:~/dossier_test$ ls | grep rg
prgm.c
bastien@bastien-P35-DS3L:~/dossier_test$ ls -l | head -3
total 8
-rw-rw-r-- 1 bastien bastien  0 nov.  17 15:34 cours.pdf
-rw-rw-r-- 1 bastien bastien  0 nov.  17 15:33 film.avi
bastien@bastien-P35-DS3L:~/dossier_test$
```


Informations système

- ps : affiche la liste des processus en cours

```

bastien@bastien-P35-D53L:~/dossier_test$ ps -A --forest
  PID TTY          TIME CMD
    2 ?            00:00:00 kthreadd
   875 ?            00:01:16 kerneloops
   886 ?            00:00:40 acpid
   894 ?            00:00:00 lightdm
  5218 tty7        06:24:47   \ Xorg
  5295 ?            00:00:00   \ lightdm
  5483 ?            00:00:04       \ init
  5555 ?            00:00:00           \ ssh-agent
  5559 ?            00:02:29           \ dbus-daemon
  5574 ?            00:00:05           \ window-stack-br
  3618 ?            00:13:20           \ gnome-terminal
  3627 ?            00:00:00           \   \ gnome-pty-helpe
 31716 pts/7        00:00:01           \   \ bash
    506 pts/7        00:00:57           |   \ vi
    367 pts/0        00:00:00           \   \ bash
  3784 pts/0        00:00:00           \   \ ps
  4842 ?            00:00:08           \ unity-scope-hom
11932 ?            00:00:00           \ kdeinit4
11935 ?            00:00:00           \   \ klauncher
11937 ?            00:00:03           \   \ kded4
11964 ?            00:00:06           \   \ kactivitymanage
11992 ?            00:00:00           \   \ kwalletd
25219 ?            00:52:54           \   \ chromium-browser
25227 ?            00:00:00           |   \   \ chrome-sandbox
25230 ?            00:00:00           |       \   \ chromium-browser
25234 ?            00:00:00           |           \   \ chromium-browser
25363 ?            00:03:11           |               \   \ chromium-browser
25394 ?            00:00:22           |                   \   \ chromium-browser
27524 ?            00:01:52           |                       \   \ chromium-browser
   3260 ?            00:00:50           |                           \   \ chromium-browser
25258 ?            00:00:26           |                               \   \ chromium-browser
25341 ?            00:00:00           |                                   \   \ chromium-browser
31869 ?            00:00:00           \ knottify4
   1786 ?            00:00:10           \ update-manager
   2435 ?            00:00:01           \ file-roller
   2499 ?            00:01:42           \ evince
  2506 ?            00:00:00           \ evince

```



Informations système

- ps : affiche la liste des processus en cours
- top : affiche la liste des processus en temps réel

```
top - 15:22:02 up 57 days, 7:51, 3 users, load average: 0,29, 0,37, 0,31
Tasks: 200 total, 2 running, 198 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0,7 us, 0,3 sy, 0,0 ni, 98,8 id, 0,2 wa, 0,0 hi, 0,0 si, 0,0 st
KiB Mem: 4046992 total, 3068572 used, 978420 free, 101024 buffers
KiB Swap: 4191228 total, 380584 used, 3810644 free, 1022060 cached Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
5861	bastien	20	0	1723860	229556	43708	S	0,7	5,7	515:13.21	compiz
3618	bastien	20	0	683996	28316	11520	S	0,3	0,7	13:25.38	gnome-terminal
5218	root	20	0	424948	166792	128404	S	0,3	4,1	385:18.36	Xorg
25219	bastien	20	0	1474312	255476	77056	S	0,3	6,3	52:56.35	chromium-browser
25394	bastien	20	0	1125596	163760	30920	S	0,3	4,0	8:23.42	chromium-browser
1	root	20	0	38144	5324	1428	S	0,0	0,1	0:11.95	init
2	root	20	0	0	0	0	S	0,0	0,0	0:00.18	kthreadd
3	root	20	0	0	0	0	S	0,0	0,0	0:07.97	ksoftirqd/0
5	root	0	-20	0	0	0	S	0,0	0,0	0:00.00	kworker/0:0H
7	root	20	0	0	0	0	R	0,0	0,0	5:48.29	rcu_sched
8	root	20	0	0	0	0	S	0,0	0,0	2:17.48	rcuos/0
9	root	20	0	0	0	0	S	0,0	0,0	2:45.08	rcuos/1

Informations système

- ps : affiche la liste des processus en cours
- top : affiche la liste des processus en temps réel
- df (Disk Free) : affiche l'espace disque disponible
- du (Disk Usage) : affiche l'espace disque utilisé

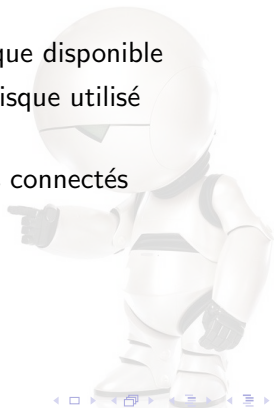
```
bastien@bastien-P35-DS3L:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1       143G   11G  125G   8% /
none            4,0K    0 4,0K   0% /sys/fs/cgroup
udev            2,0G   4,0K  2,0G   1% /dev
tmpfs           396M   1,1M  395M   1% /run
none            5,0M    0 5,0M   0% /run/lock
none            2,0G   39M  1,9G   2% /run/shm
none            100M  100K  100M   1% /run/user
bastien@bastien-P35-DS3L:~$ du -hsc /*
6,7M    ./bin
164M    ./Bureau
48K     ./code
4,0K    ./Documents
16K     ./dossier_test
148K    ./Images
4,0K    ./Musique
422M    ./Téléchargements
4,0K    ./tutu
1,1G    total
```

Informations système

- ps : affiche la liste des processus en cours
- top : affiche la liste des processus en temps réel
- df (Disk Free) : affiche l'espace disque disponible
- du (Disk Usage) : affiche l'espace disque utilisé
- who : affiche la liste des utilisateurs connectés

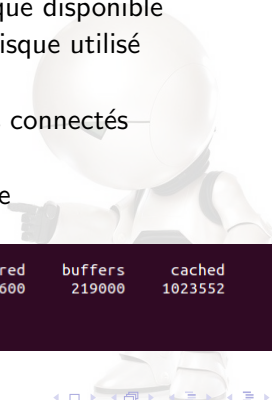
DON'T PANIC

IT'S NOT WINDOWS



Informations système

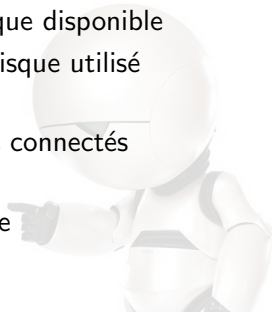
- ps : affiche la liste des processus en cours
- top : affiche la liste des processus en temps réel
- df (Disk Free) : affiche l'espace disque disponible
- du (Disk Usage) : affiche l'espace disque utilisé
- who : affiche la liste des utilisateurs connectés
- free : affiche la mémoire vive utilisée



```
bastien@bastien-P35-DS3L:~$ free
              total        used         free       shared    buffers     cached
Mem:      4046992     3419008     627984      164600      219000     1023552
-/+ buffers/cache:    2176456     1870536
Swap:      4191228     379424      3811804
bastien@bastien-P35-DS3L:~$
```

Informations système

- ps : affiche la liste des processus en cours
- top : affiche la liste des processus en temps réel
- df (Disk Free) : affiche l'espace disque disponible
- du (Disk Usage) : affiche l'espace disque utilisé
- who : affiche la liste des utilisateurs connectés
- free : affiche la mémoire vive utilisée
- uname : information sur le système



```
bastien@bastien-P35-DS3L:~$ uname -a
Linux bastien-P35-DS3L 3.13.0-24-generic #46-Ubuntu SMP Thu Apr 10 19:11:08 UTC 2014
x86_64 x86_64 x86_64 GNU/Linux
bastien@bastien-P35-DS3L:~$
```

Et encore plus ...

- clear : efface l'écran !
- echo : affiche une chaîne de caractères.
Exemple : echo "Hello World"
- history : affiche l'historique des commandes

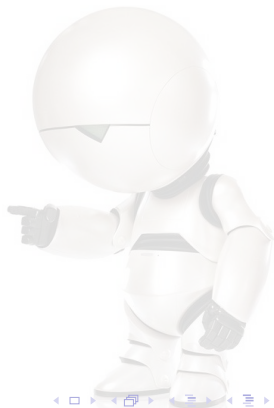
```
2117 ps -A --forest
2118 top
2119 df -h
2120 du -hsc
2121 du -hsc ./*
2122 who
2123 free
2124 uname -a
2125 clear
2126 echo "coucou"
2127 history
bastien@bastien-P35-DS3L:~$
```

Les scripts : la programmation dans le bash

- Les conditions

```
if [ x -eq 0 ] ; then
...
elif [ x -gt 0 ] ; then
...
else
...
fi
```

DON'T PANIC
IT'S NOT WINDOWS



Les scripts : la programmation dans le bash

- Les conditions

```
if [ x -eq 0 ] ; then
```

```
...
```

```
elif [ x -gt 0 ] ; then
```

```
...
```

```
else
```

```
...
```

```
fi
```

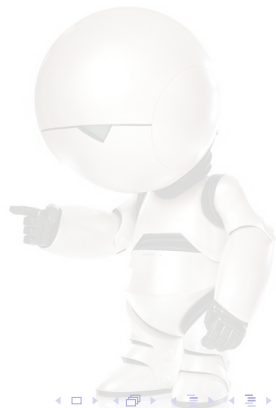
- Les boucles shell

```
for i in $(ls);
```

```
do
```

```
echo $i
```

```
done
```



DON'T PANIC

IT'S NOT WINDOWS

Les scripts : la programmation dans le bash

- Les conditions

```
if [ x -eq 0 ] ; then
```

```
...
```

```
elif [ x -gt 0 ] ; then
```

```
...
```

```
else
```

```
...
```

```
fi
```

- Les boucles shell

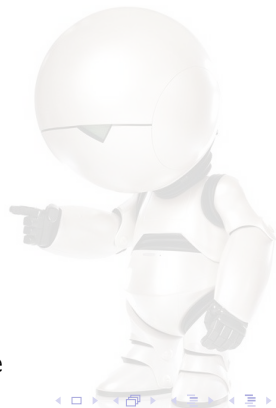
```
for i in $( ls );
```

```
do
```

```
echo $i
```

```
done
```

- \$: récupère la valeur d'une variable

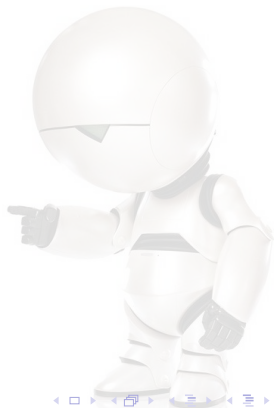


DON'T PANIC
IT'S NOT WINDOWS

Les scripts : la programmation dans le bash

- Les boucles C-like
for i in 'seq 1 10';
do
echo \$i
done

DON'T PANIC
IT'S NOT WINDOWS



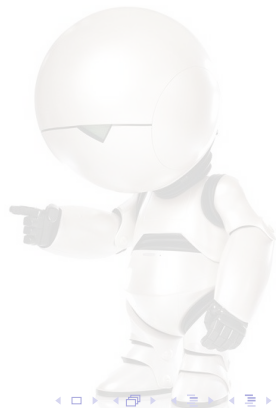
Les scripts : la programmation dans le bash

- Les boucles C-like

```
for i in `seq 1 10`;  
do  
echo $i  
done
```

- Les boucles avec conditions

```
COUNTER=0  
while [ $ COUNTER -lt 10 ];  
do  
echo The counter is $COUNTER  
let COUNTER=COUNTER+1  
done
```



DON'T PANIC
IT'S NOT WINDOWS

Les scripts : la programmation dans le bash

- Les boucles C-like

```
for i in `seq 1 10`;
do
echo $i
done
```

- Les boucles avec conditions

```
COUNTER=0
while [ $ COUNTER -lt 10 ];
do
echo The counter is $COUNTER
let COUNTER=COUNTER+1
done
```

- test :

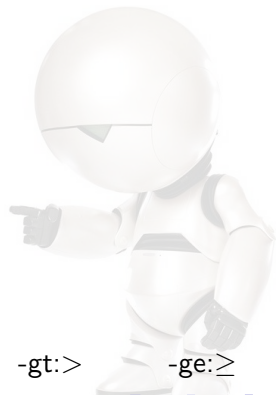
-eq==

-lt:<

-le:≤

-gt:>

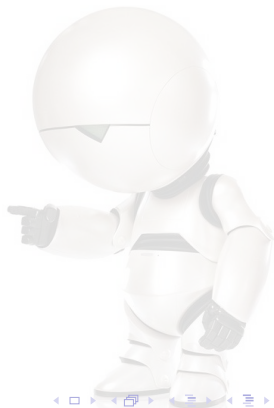
-ge:≥



DON'T PANIC
IT'S NOT WINDOWS

Exemple de script

```
#!/bin/bash
# la ligne precedente
# selectionne le shell voulu,
# et ceci est un commentaire !
FILE_LIST="ls *.html"
echo "Liste des fichiers '${FILE_LIST}'"
RESULT=""
for file in ${FILE_LIST}
do
FIRST_LINE='head -1 ${file}'
RESULT=${RESULT}${FIRST_LINE}
done
echo ${RESULT} | cat > entete.txt
echo "'$RESULT' Fin du script. "
```



1 Systèmes d'exploitation

2 Bash

3 Droits et utilisateur

- Droits d'un fichier
- Modification des droits
- Propriétaire et groupes

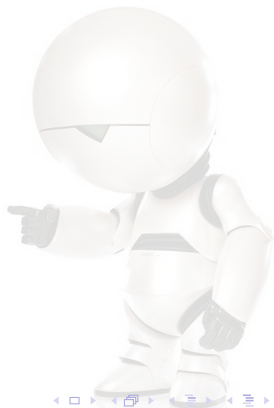
4 Connexion

5 Système de fichiers

6 Processus et mémoire

DON'T PANIC

IT'S NOT WINDOWS



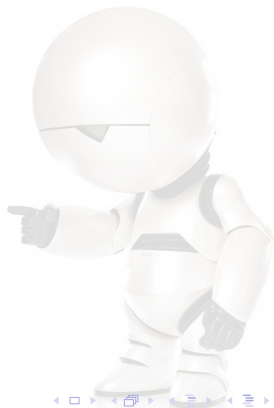
Droits d'un fichier ou d'un dossier

\$ls -l

-rwxr-xr- - bastien bastien 359 nov. 4 09:54 prgm.c

drwxr-xr-x bastien bastien 4096 sept. 2 18:11 dossier_test

DON'T PANIC
IT'S NOT WINDOWS



Droits d'un fichier ou d'un dossier

\$ls -l

-rwxr-xr- - bastien bastien 359 nov. 4 09:54 prgm.c

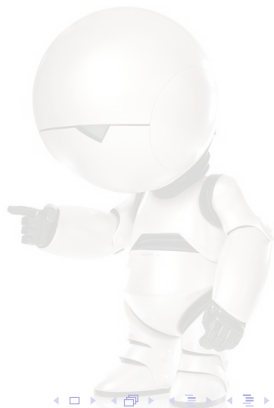
drwxr-xr-x bastien bastien 4096 sept. 2 18:11 dossier_test

r : droit en lecture (ou non -)

w : droit en écriture (ou non -)

x : droit en exécution (ou non -)

DON'T PANIC
IT'S NOT WINDOWS



Droits d'un fichier ou d'un dossier

\$ls -l

-rwxr-xr- - bastien bastien 359 nov. 4 09:54 prgm.c

drwxr-xr-x bastien bastien 4096 sept. 2 18:11 dossier_test

r : droit en lecture (ou non -)

w : droit en écriture (ou non -)

x : droit en exécution (ou non -)

1^{er} caractère : indique s'il s'agit d'un dossier (d) ou non (-)

3 caractères suivants : droits du propriétaire du fichier

3 caractères suivants : droits des membres du groupe du fichier

3 caractères suivants : droits des autres utilisateurs

1^{er} nom : propriétaire du fichier

2nd nom : groupe

Modification des droits

- La commande chmod

Syntaxe : chmod référence opérateur mode fichier

Référence :

- u : l'utilisateur propriétaire du fichier
- g : les membres du groupe du fichier
- o : les autres (ni u ni g)
- a : tout le monde

Opérateur :

- + : ajout de droits
- - : suppression de droits

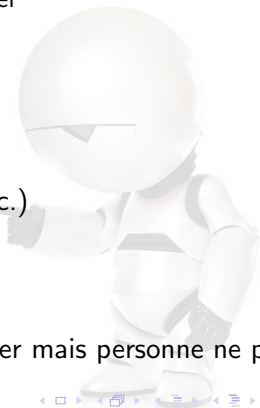
Mode : r (read), w (write) ou x (exec.)

- Exemple :

chmod g-w prgm.c

chmod a+r prgm.c

⇒ Tout le monde peut lire mon fichier mais personne ne peut le modifier.



Modification des droits

- Modification en mode octal

- 0 : rien
- 1 : exécution
- 2 : écriture
- 4 : lecture

⇒ L'addition donne les droits !

⇒ lecture + écriture = $4+2 = 6$

⇒ exécution + lecture = $1 + 4 = 5$

- Exemple :

`chmod 777 test.sh` : tout le monde peut lire, écrire et exécuter le fichier

`chmod 754 test.sh` : je peux tout faire, le groupe peut seulement lire et exécuter, les autres ne peuvent que lire le fichier

Propriétaire et groupes

- La commande `chown` : changement de propriétaire
Syntaxe : `chown nouveau_propriétaire fichier`
Exemple : `chown gontrand prgm`
Option `-R` : changement récursif pour un dossier et tous ses éléments
Attention : seul le propriétaire et root sont autorisés !
- La commande `chgrp` : changement de groupe
Même Syntaxe, même utilisation !
- Création d'un groupe : `groupadd nom du groupe`
- Ajouter un utilisateur à un groupe (ou plusieurs)
`usermod -aG groupe1, groupe2 user`
⇒ Permet de limiter la visibilité des données !

DON'T PANIC
IT'S NOT WINDOWS

1 Systèmes d'exploitation

2 Bash

3 Droits et utilisateur

4 Connexion

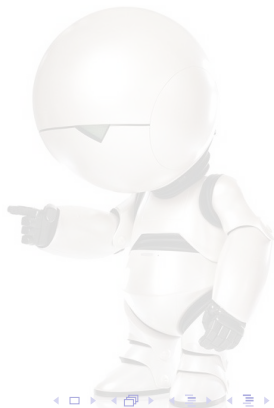
- OS multi-user
- Connexion à distance
- Les tunnels

5 Système de fichiers

6 Processus et mémoire

DON'T PANIC

IT'S NOT WINDOWS



OS multi-user



- Réduction du coût, du bruit, de l'encombrement, de la pollution
- Partage des ressources

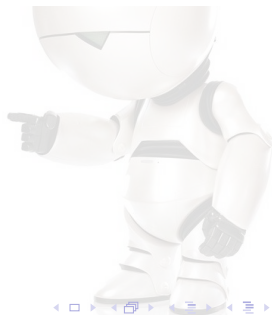


Connexion à distance : ssh

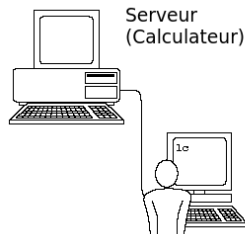


Serveur
(Calculateur)

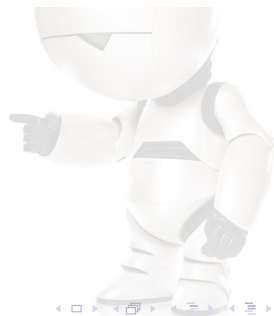
DON'T PANIC
IT'S NOT WINDOWS



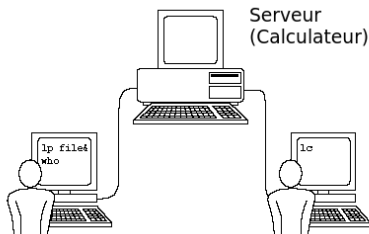
Connexion à distance : ssh



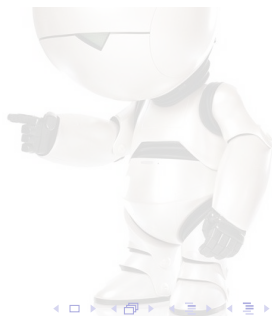
DON'T PANIC
IT'S NOT WINDOWS



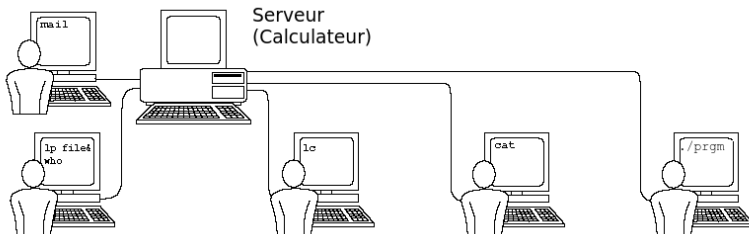
Connexion à distance : ssh



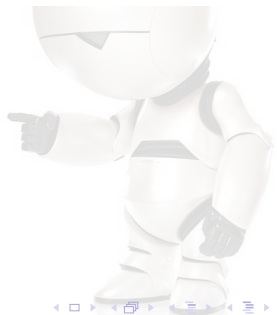
DON'T PANIC
IT'S NOT WINDOWS



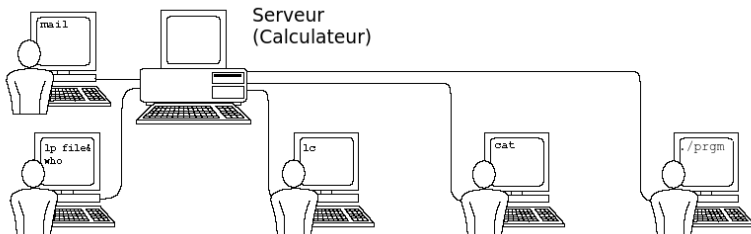
Connexion à distance : ssh



DON'T PANIC
IT'S NOT WINDOWS



Connexion à distance : ssh



- Connexion à distance

- `ssh dipierro@jade.cines.fr -p 1001` (Option `-p` : sélection du port, défaut 22)
- `ssh bastien@147.94.57.136 -X` (Option `-X` : permet les interfaces graphiques)

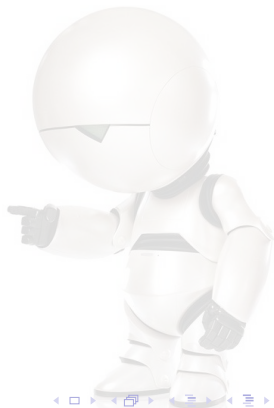
- Copie à distance

- `scp ernest@saturne:/home/ernest/Bureau/prgm.c ./code/`
- `scp bg.png ernest@saturne:/home/ernest/Images/`

Les tunnels

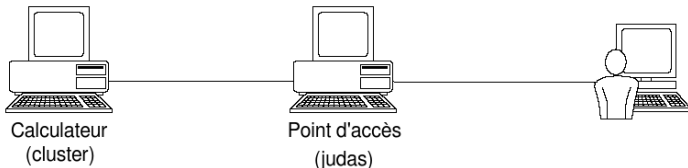
- Connexion au travers d'un (ou plusieurs) point d'accès

DON'T PANIC
IT'S NOT WINDOWS

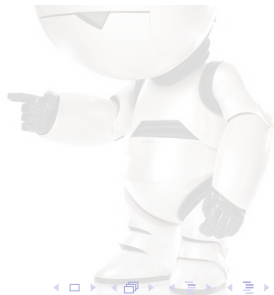


Les tunnels

- Connexion au travers d'un (ou plusieurs) point d'accès
- 1^{ère} solution : 2 connexions ssh

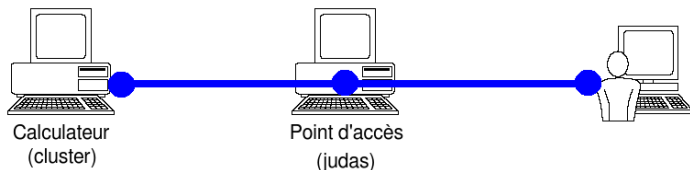


DON'T PANIC
IT'S NOT WINDOWS



Les tunnels

- Connexion au travers d'un (ou plusieurs) point d'accès
- 1^{ère} solution : 2 connexions ssh



- 2^{nde} solution : création d'un tunnel (redirection sur le port local 1234)

```
ssh -N -f dipierro@judas -L 1234:cluster:22
```

- -L : forward (redirection)
- -N : pas de commande à distance (redirection seulement)
- -f : s'exécute en arrière plan (reste actif)

Les tunnels

- Connexion au travers d'un (ou plusieurs) point d'accès
- 1^{ère} solution : 2 connexions ssh



- 2^{nde} solution : création d'un tunnel (redirection sur le port local 1234)

```
ssh -N -f dipierro@judas -L 1234:cluster:22
```

- -L : forward (redirection)
- -N : pas de commande à distance (redirection seulement)
- -f : s'exécute en arrière plan (reste actif)

- Connexion au travers du tunnel (idem scp avec -P)

```
ssh dipierro@localhost -p 1234
```


1 Systèmes d'exploitation

2 Bash

3 Droits et utilisateur

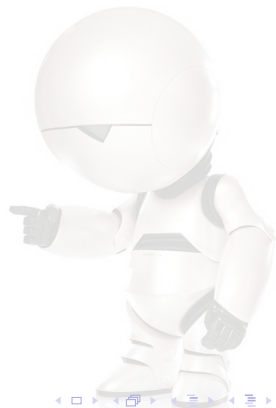
4 Connexion

5 Système de fichiers

- Partitions
- Système de fichiers Linux

6 Processus et mémoire

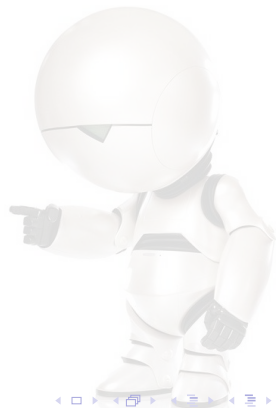
DON'T PANIC
IT'S NOT WINDOWS



Les partitions

- Idée : découper virtuellement un disque dur en plusieurs disques (partitions)

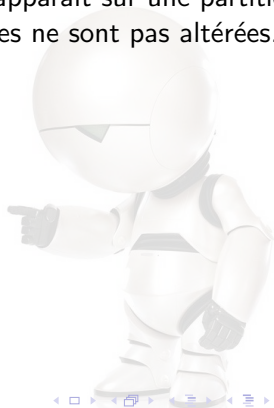
DON'T PANIC
IT'S NOT WINDOWS



Les partitions

- Idée : découper virtuellement un disque dur en plusieurs disques (partitions)
- But : sécurisation. Si un problème apparaît sur une partition, les données enregistrées sur les autres ne sont pas altérées.

DON'T PANIC
IT'S NOT WINDOWS



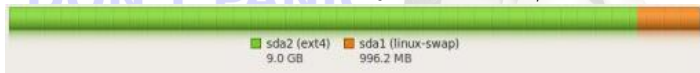
Les partitions

- Idée : découper virtuellement un disque dur en plusieurs disques (partitions)
- But : sécurisation. Si un problème apparaît sur une partition, les données enregistrées sur les autres ne sont pas altérées.
- Format des partitions :
 - FAT et FAT 32 (File Allocation Table) : lisible par tous les OS (**Attention** : fichiers limités à 4Go). Utilisé dans la plupart des clefs USB et DD externes.
 - NTFS (New Technology File System) : réservé à Microsoft.
 - ext2, ext3 ou ext4 : réservé aux systèmes UNIX/Linux.

IT'S NOT WINDOWS

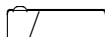
Les partitions

- Idée : découper virtuellement un disque dur en plusieurs disques (partitions)
- But : sécurisation. Si un problème apparaît sur une partition, les données enregistrées sur les autres ne sont pas altérées.
- Format des partitions :
 - FAT et FAT 32 (File Allocation Table) : lisible par tous les OS (**Attention** : fichiers limités à 4Go). Utilisé dans la plupart des clefs USB et DD externes.
 - NTFS (New Technology File System) : réservé à Microsoft.
 - ext2, ext3 ou ext4 : réservé aux systèmes UNIX/Linux.



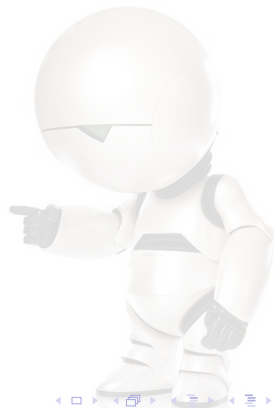
- Swap (ou échange) : partition cachée et réservée au système.
⇒ Extension de la mémoire physique (RAM) sur le disque dur.

Système de fichiers Linux

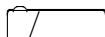


/=root

DON'T PANIC
IT'S NOT WINDOWS



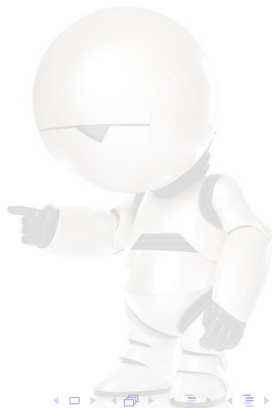
Système de fichiers Linux



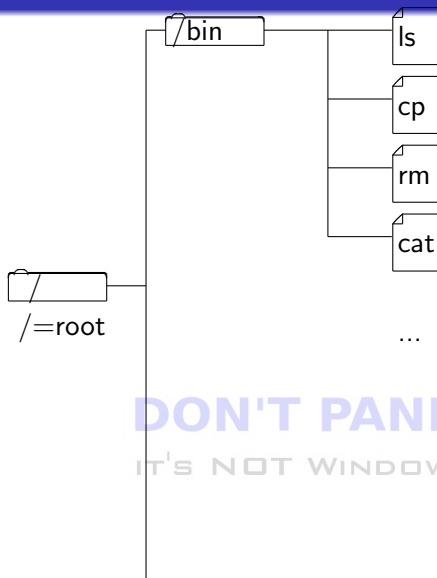
/=root

DON'T PANIC

IT'S NOT WINDOWS



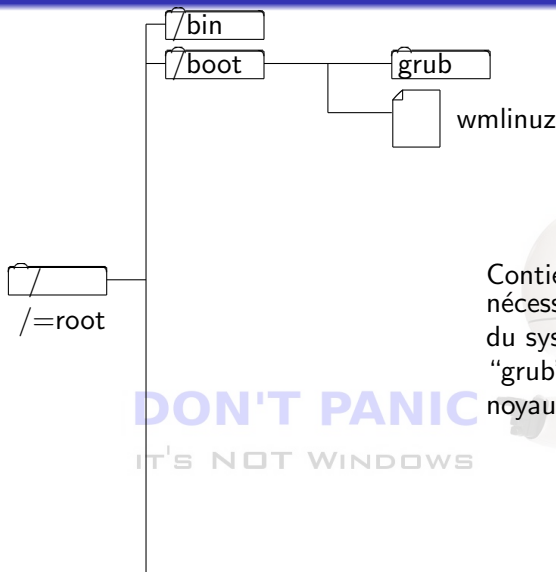
Système de fichiers Linux



Contient la plupart des programmes usuels partagés entre le système, les utilisateurs et l'administrateur

DON'T PANIC
IT'S NOT WINDOWS

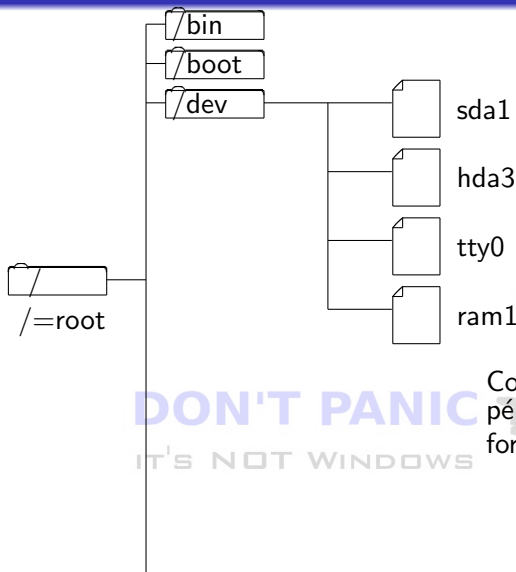
Système de fichiers Linux



Contient les fichiers nécessaires au démarrage du système, le boot-loader "grub" ainsi que le noyau "wmlinux"

DON'T PANIC
IT'S NOT WINDOWS

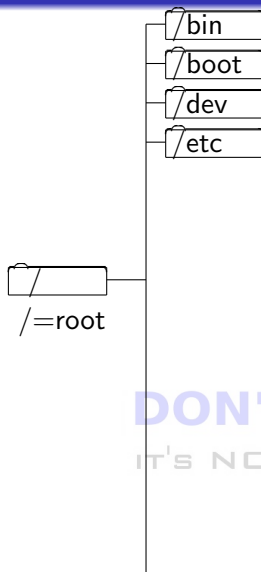
Système de fichiers Linux



DON'T PANIC
IT'S NOT WINDOWS

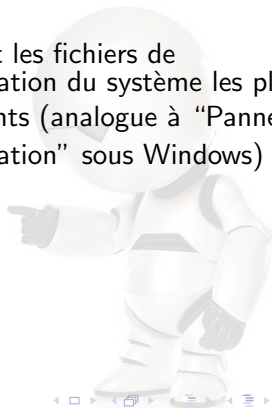
Contient l'ensemble des périphériques matériels sous forme de fichiers spéciaux

Système de fichiers Linux

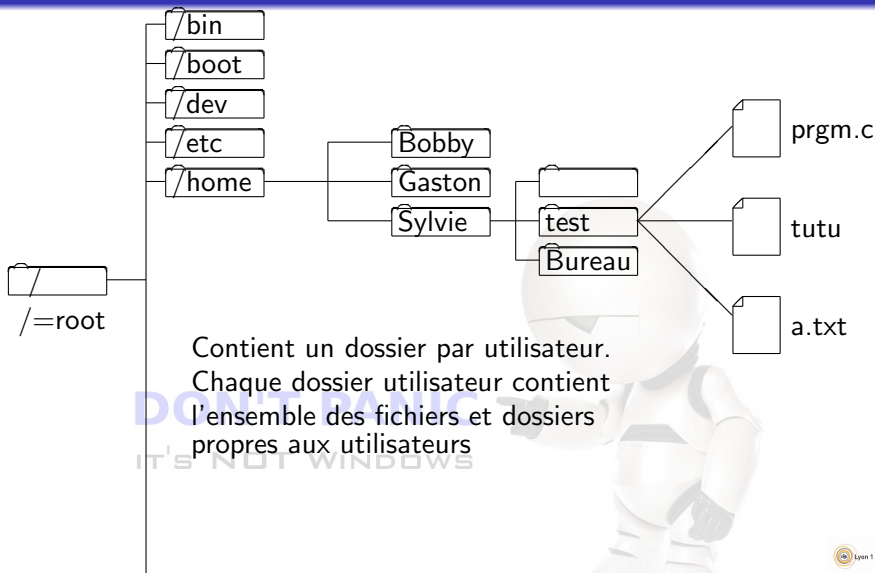


Contient les fichiers de configuration du système les plus importants (analogue à “Panneau de configuration” sous Windows)

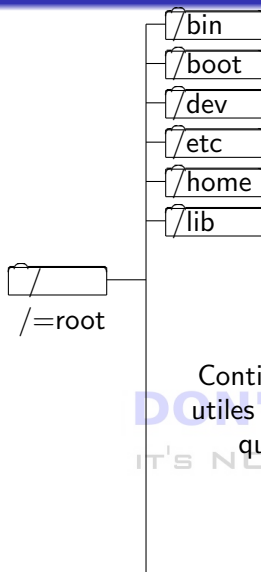
DON'T PANIC
IT'S NOT WINDOWS



Système de fichiers Linux

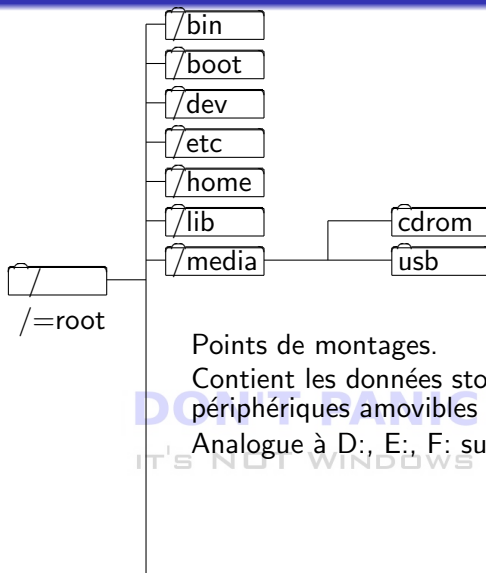


Système de fichiers Linux



Contient un ensemble de bibliothèques utiles aux programmes du système ainsi qu'aux programmes utilisateurs.

Système de fichiers Linux



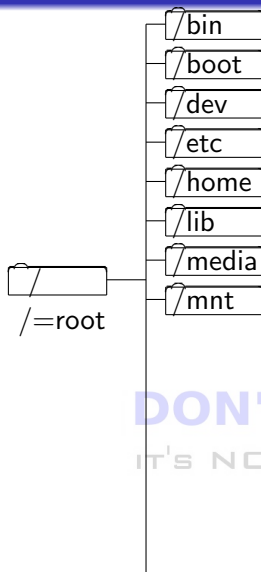
Points de montages.

Contient les données stockées sur périphériques amovibles (USB, CDROM,...).

Analogue à D:, E:, F: sur Windows



Système de fichiers Linux

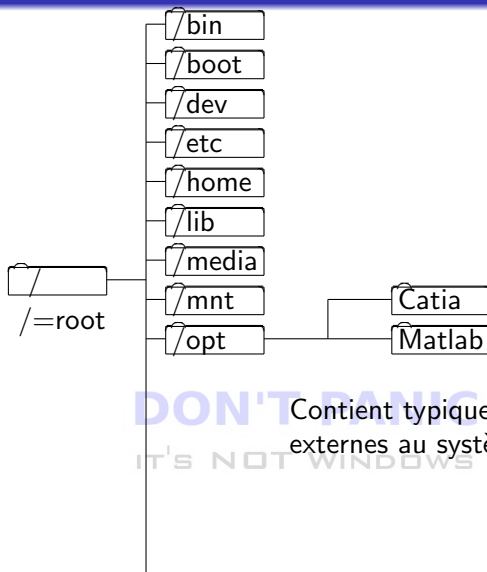


Même comportement que media.
Dépend de la distribution ...

DON'T PANIC
IT'S NOT WINDOWS



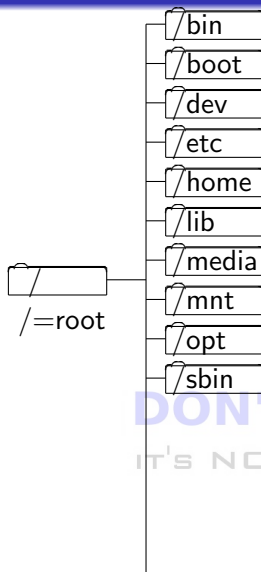
Système de fichiers Linux



DON'T PANIC
IT'S NOT WINDOWS

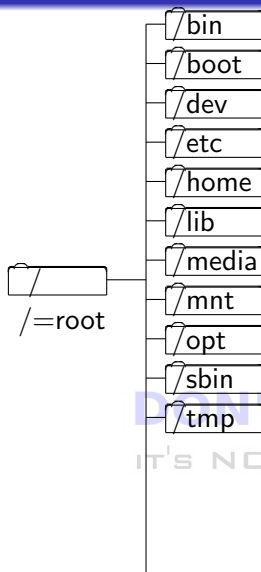
Contient typiquement les logiciels
externes au système.

Système de fichiers Linux



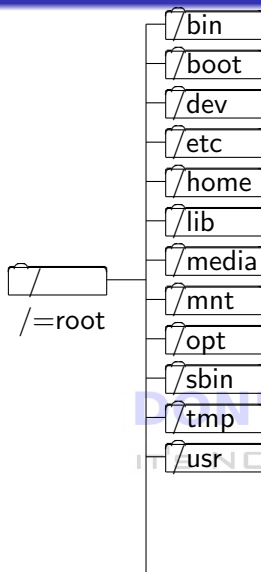
Contient les logiciels uniquement nécessaires au système et à l'administrateur.

Système de fichiers Linux

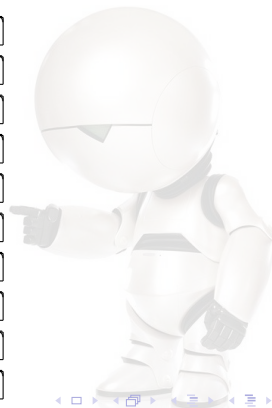
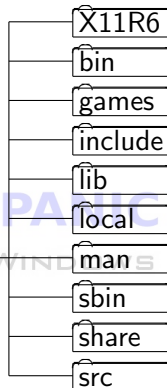


Permet le stockage temporaire de fichiers.
Accessible à tout le monde, mais est effacé à chaque redémarrage.

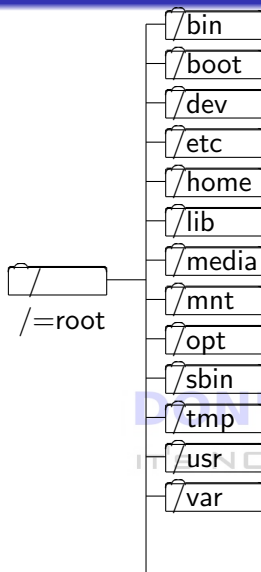
Système de fichiers Linux



Ensemble de programmes, bibliothèques, documentations (man) liés aux programmes utilisateurs.

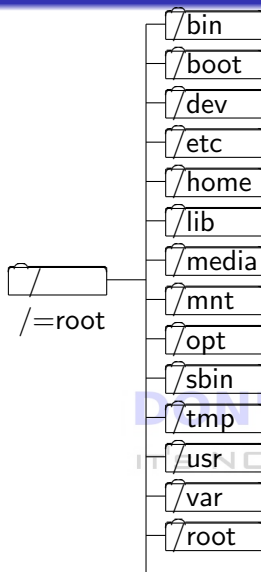


Système de fichiers Linux



Dossier de stockage de fichiers temporaires et fichiers de variables créés par les programmes utilisateurs (fichiers log, file de mail, fichiers en cours de téléchargement, en cours d'impression ...)

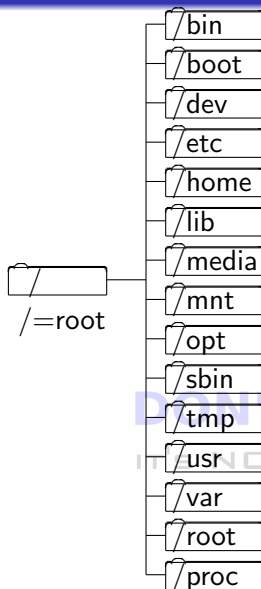
Système de fichiers Linux



Dossier personnel de root.
root = administrateur



Système de fichiers Linux



Système de fichiers virtuel contenant des informations sur les ressources du système. Contient un dossier par processus actif contenant des infos sur le processus, ainsi que des fichiers relatifs à l'état du système.



1 Systèmes d'exploitation

2 Bash

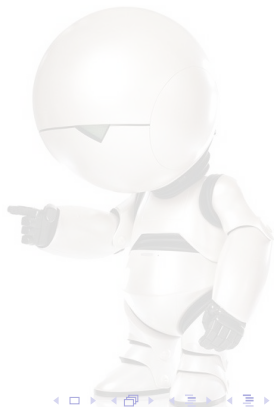
3 Droits et utilisateur

4 Connexion

5 Système de fichiers

6 Processus et mémoire

- Notion de processus
- Propriétés d'un processus
- Cycle de vie d'un processus
- Traitement des données

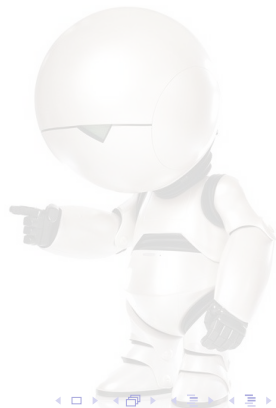


DON'T PANIC
IT'S NOT WINDOWS

Notion de processus

- Notion centrale du système d'exploitation

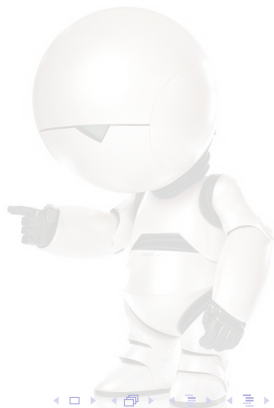
DON'T PANIC
IT'S NOT WINDOWS



Notion de processus

- Notion centrale du système d'exploitation
- Définition : un processus est un programme exécutable (un morceau de code source) qui s'exécute en mémoire.

DON'T PANIC
IT'S NOT WINDOWS



Notion de processus

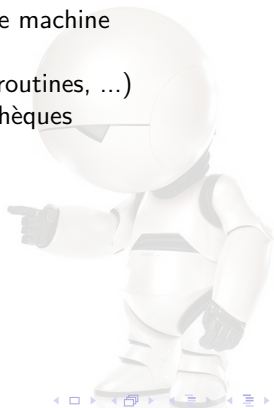
- Notion centrale du système d'exploitation
- Définition : un processus est un programme exécutable (un morceau de code source) qui s'exécute en mémoire.

Programme exécutable :

- Séquence d'instructions en langage machine
- Calculs, tests ...
- Appel aux procédures (fonctions, routines, ...)
- Appel au système ou à des bibliothèques

Exemple : firefox

DON'T PANIC
IT'S NOT WINDOWS



Notion de processus

- Notion centrale du système d'exploitation
- Définition : un processus est un programme exécutable (un morceau de code source) qui s'exécute en mémoire.

Programme exécutable :

- Séquence d'instructions en langage machine
- Calculs, tests ...
- Appel aux procédures (fonctions, routines, ...)
- Appel au système ou à des bibliothèques

Exemple : firefox

- Membre d'une hiérarchie parent/enfant

DON'T PANIC
IT'S NOT WINDOWS

Notion de processus

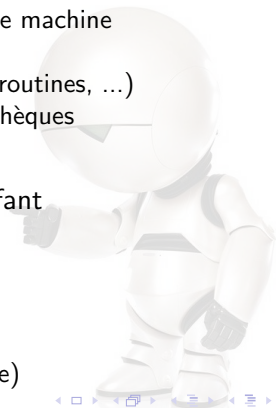
- Notion centrale du système d'exploitation
- Définition : un processus est un programme exécutable (un morceau de code source) qui s'exécute en mémoire.

Programme exécutable :

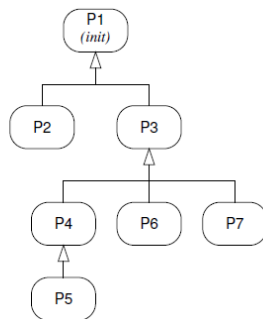
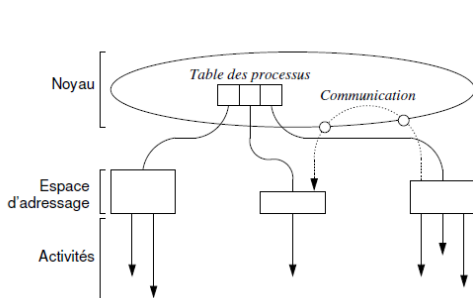
- Séquence d'instructions en langage machine
- Calculs, tests ...
- Appel aux procédures (fonctions, routines, ...)
- Appel au système ou à des bibliothèques

Exemple : firefox

- Membre d'une hiérarchie parent/enfant
- Programme en cours d'exécution
 - Activité (calculs, tests ...)
 - Données (zones mémoires)
 - Propriétés
 - Communication (dans la hiérarchie)



Notion de processus



DON'T PANIC
IT'S NOT WINDOWS

- Chaque processus est indépendant ! $\Rightarrow$ Progression en "parallèle" !



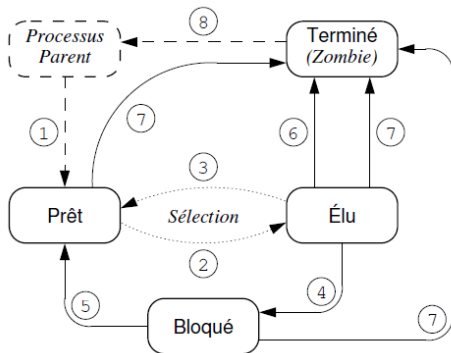
Propriétés d'un processus

- Un identifiant unique (PID)
- Une priorité
- L'identité de l'utilisateur (et ses droits)
- Son état (Sleeping, Running, Paused, Zombie)

```
top - 15:22:02 up 57 days, 7:51, 3 users, load average: 0,29, 0,37, 0,31
Tasks: 200 total, 2 running, 198 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0,7 us, 0,3 sy, 0,0 ni, 98,8 id, 0,2 wa, 0,0 hi, 0,0 si, 0,0 st
KiB Mem: 4046992 total, 3068572 used, 978420 free, 101024 buffers
KiB Swap: 4191228 total, 380584 used, 3810644 free. 1022060 cached Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
5861	bastien	20	0	1723860	229556	43708	S	0,7	5,7	515:13.21	compiz
3618	bastien	20	0	683996	28316	11520	S	0,3	0,7	13:25.38	gnome-terminal
5218	root	20	0	424948	166792	128404	S	0,3	4,1	385:18.36	Xorg
25219	bastien	20	0	1474312	255476	77056	S	0,3	6,3	52:56.35	chromium-browser
25394	bastien	20	0	1125596	163760	30920	S	0,3	4,0	8:23.42	chromium-browser
1	root	20	0	38144	5324	1428	S	0,0	0,1	0:11.95	init
2	root	20	0	0	0	0	S	0,0	0,0	0:00.18	kthreadd
3	root	20	0	0	0	0	S	0,0	0,0	0:07.97	ksoftirqd/0
5	root	0	-20	0	0	0	S	0,0	0,0	0:00.00	kworker/0:0H
7	root	20	0	0	0	0	R	0,0	0,0	5:48.29	rcu_sched
8	root	20	0	0	0	0	S	0,0	0,0	2:17.48	rcuos/0
9	root	20	0	0	0	0	S	0,0	0,0	2:45.08	rcuos/1

Cycle de vie d'un processus



1: Création

2: Utilisation du processeur

3: Un autre utilise le processeur

4: Suspension ou attente de ressources

5: Relance ou ressource disponible

6: Terminaison normale

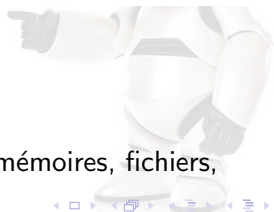
7: Destruction

8: Attente par le parent

DON'T PANIC

IT'S NOT WINDOWS

- Prêt : en attente du CPU
- Élu : en cours d'exécution
- Bloqué : en attente de ressources (mémoires, fichiers, périphériques ...)



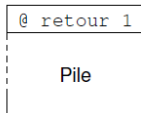
Traitement des données : le Stack

- Principe : empilement des données
- Ordre : premier arrivé, dernier sorti

Trajet du pointeur programme

```
{  
  // ...  
  x=f1();  
  // @ retour 1  
  // ...  
}
```

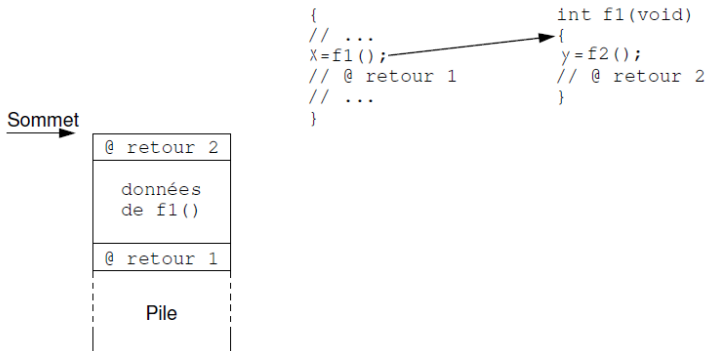
Sommet



Traitement des données : le Stack

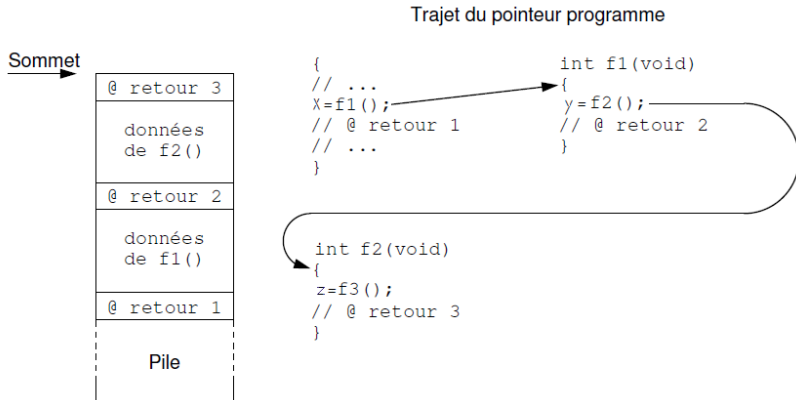
- Principe : empilement des données
- Ordre : premier arrivé, dernier sorti

Trajet du pointeur programme



Traitement des données : le Stack

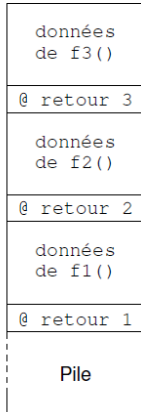
- Principe : empilement des données
- Ordre : premier arrivé, dernier sorti



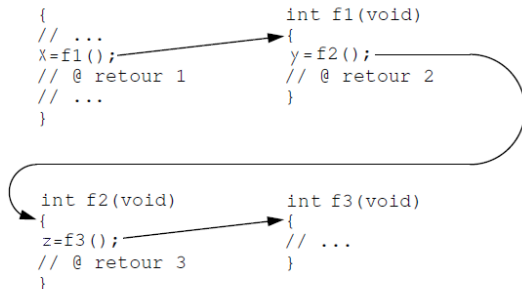
Traitement des données : le Stack

- Principe : empilement des données
- Ordre : premier arrivé, dernier sorti

Sommet
→

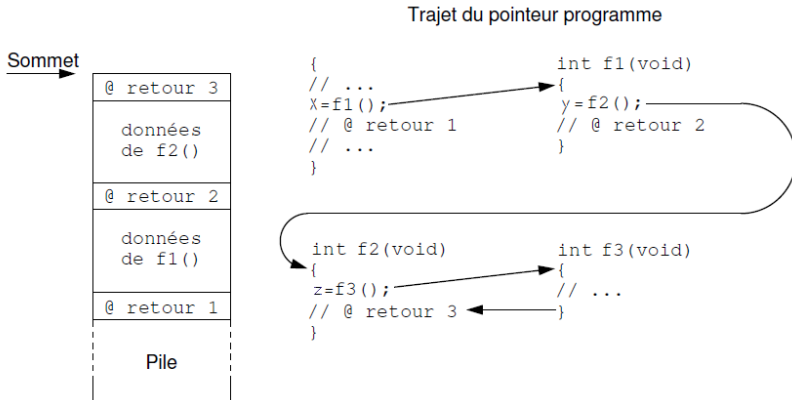


Trajet du pointeur programme



Traitement des données : le Stack

- Principe : empilement des données
- Ordre : premier arrivé, dernier sorti

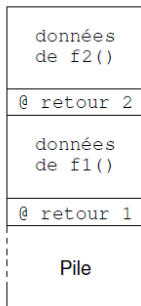


Traitement des données : le Stack

- Principe : empilement des données
- Ordre : premier arrivé, dernier sorti

Trajet du pointeur programme

Sommet



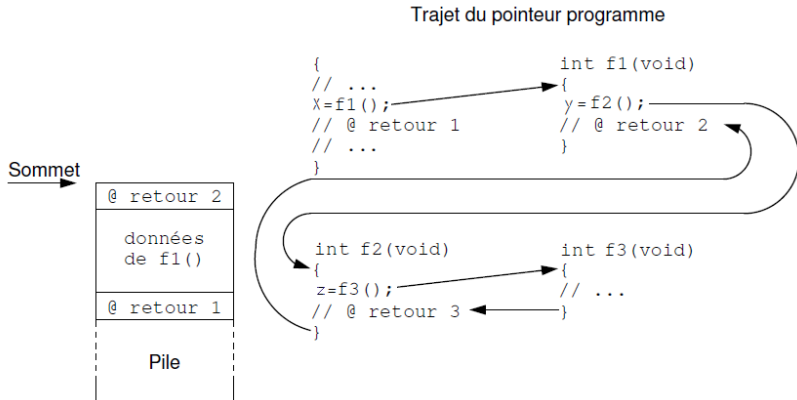
```
{  
  // ...  
  x=f1();  
  // @ retour 1  
  // ...  
}  
  
int f1(void)  
{  
  y=f2();  
  // @ retour 2  
}
```

Diagram illustrating the program pointer trajectory:

- The call to `x=f1();` in the first function block points to the start of the `int f1(void)` function.
- The return from `int f1(void)` points back to the instruction following `x=f1();` in the first function block.
- The call to `z=f3();` in the `int f2(void)` function points to the start of the `int f3(void)` function.
- The return from `int f3(void)` points back to the instruction following `z=f3();` in the `int f2(void)` function.
- The return from `int f2(void)` points back to the instruction following the call to `int f2(void)` in the first function block.

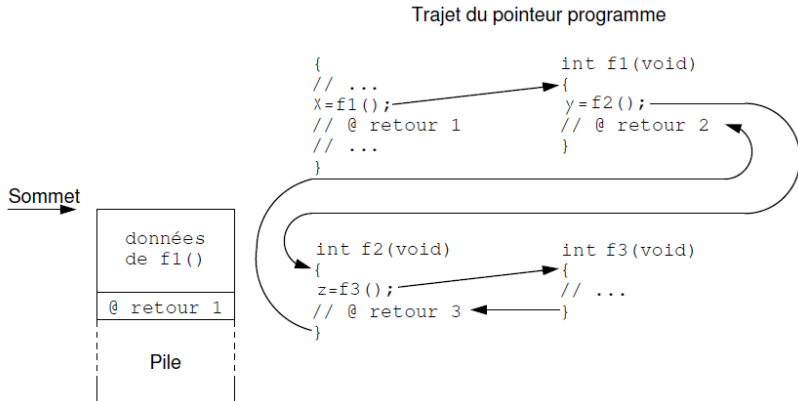
Traitement des données : le Stack

- Principe : empilement des données
- Ordre : premier arrivé, dernier sorti



Traitement des données : le Stack

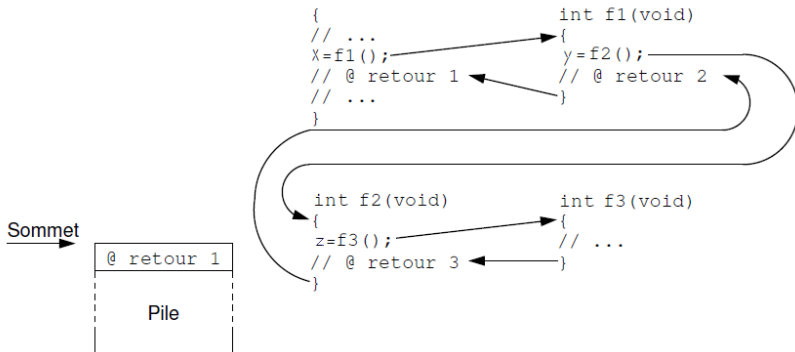
- Principe : empilement des données
- Ordre : premier arrivé, dernier sorti



Traitement des données : le Stack

- Principe : empilement des données
- Ordre : premier arrivé, dernier sorti

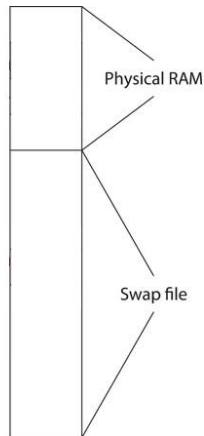
Trajet du pointeur programme



Traitement des données : la mémoire virtuelle

- Deux programmes (processus) A et B
⇒ Chacun ne doit pas accéder à l'espace mémoire de l'autre !

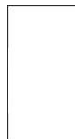
DON'T PANIC
IT'S NOT A WINDOW



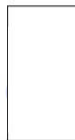
Traitement des données : la mémoire virtuelle

- Deux programmes (processus) A et B
⇒ Chacun ne doit pas accéder à l'espace mémoire de l'autre !
- Le système d'exploitation crée une mémoire virtuelle par processus

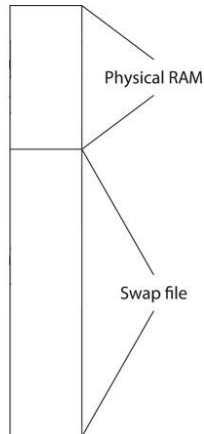
DON'T PANIC
IT'S NOT A WINDOW



Program A

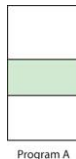


Program B

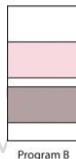


Traitement des données : la mémoire virtuelle

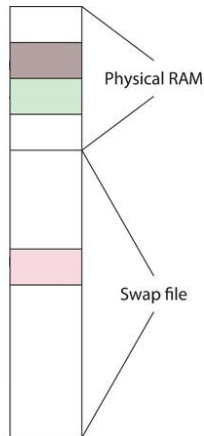
- Deux programmes (processus) A et B
⇒ Chacun ne doit pas accéder à l'espace mémoire de l'autre !



- Le système d'exploitation crée une mémoire virtuelle par processus



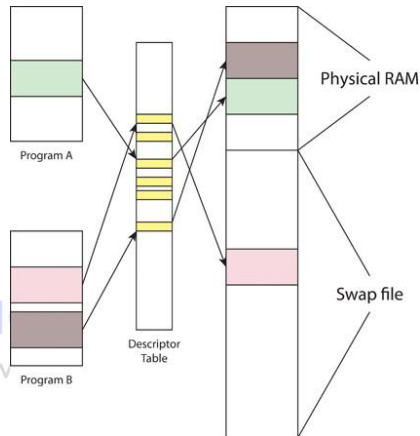
- Comment est-ce géré dans la RAM ?



DON'T PANIC
IT'S NOT WINDOWS

Traitement des données : la mémoire virtuelle

- Deux programmes (processus) A et B
⇒ Chacun ne doit pas accéder à l'espace mémoire de l'autre !
- Le système d'exploitation crée une mémoire virtuelle par processus
- Comment est-ce géré dans la RAM ?
- Utilisation d'une Table Globale de Descripteurs (GDT)



Conclusion

Unix/Linux

- Système d'exploitation créé pour le calcul scientifique
- Utilisé sur la plupart des grands calculateurs
- Possibilité de connecter plusieurs utilisateurs en même temps
- Gestion efficace des droits et du système de fichiers

Bash

- Interpréteur de commande puissant et efficace
- Possibilité de programmer