UNIX
les bases

V. Berry

Université Montpellier 2

http://www.lirmm.fr/~vberry

Le module

Découpage :

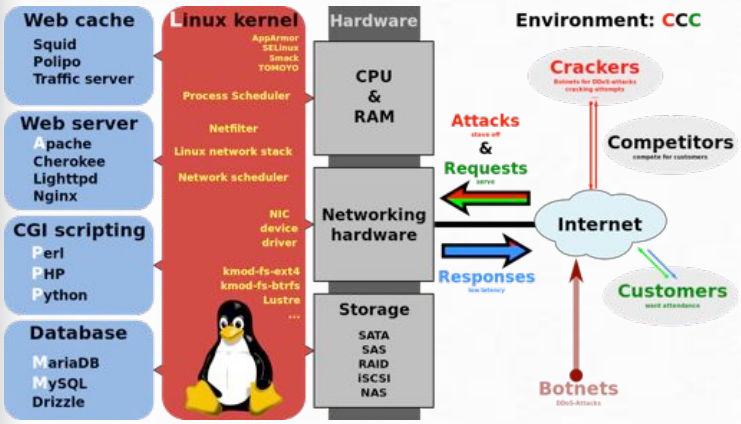
- Cours 1
- TP1 : I.G. - Navigateurs - Terminal - Cmdes de base
- TP2 : Commandes (suite) -

groupes allégés

- Cours 2
- TP 3 : Chemins - Archives - Droits
- TP 4 : Processus - Mémoire - Commandes réseau

groupes complets

Pourquoi Unix ?



Serveurs Web

Salles TP Polytech

Calculateurs,
Data centers

Appareils
embarqués

Au menu

- Notion d'architecture d'un ordinateur
- Notion de Systèmes d'Exploitation («SE»)
- Manipulation du Système de Gestion de Fichiers («SGF»)
- Le terminal de commandes
- Les processus
- Les éditeurs nano & emacs
- Commandes avancées & redirections
- Manipulation d'archives
- Variables d'environnement et fichiers de configuration
- Scripts
- Notions de réseau

Cours 1

Cours 2

Introduction

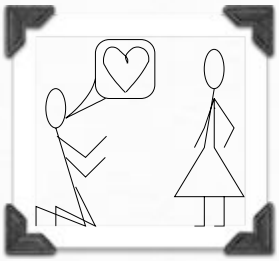
Objectif de l'ordinateur : manipuler de l'information (d'où «informatique»).

J. Popp

L'information peut

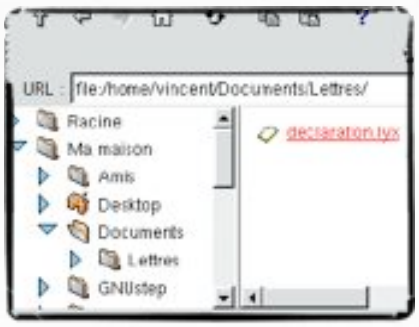
- avoir différentes natures : programme, texte, son, vidéo, ...
- subir différents traitements
  - acquisition (périphériques d'E/S, réseau, logiciels),
  - enregistrement (fichiers).
  - modification (logiciels spécifiques)

L'information a différents aspects suivant son niveau de représentation / l'acteur qui la considère

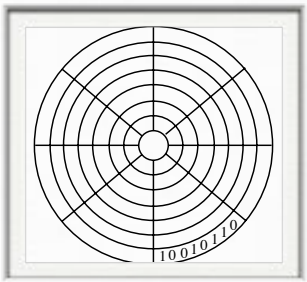


Sémantique : sens que l'information a pour les **humains**, forme (senti)mentale.

Structure : cette information est exprimée dans un document structuré en **mots, phrases, paragraphes** grâce à un **traitement de textes**.



Logique : ce document est **stocké** dans un **fichier**, repéré dans le **S.E.** par un nom (declaration.lyx) dans un **répertoire** / **dossier**

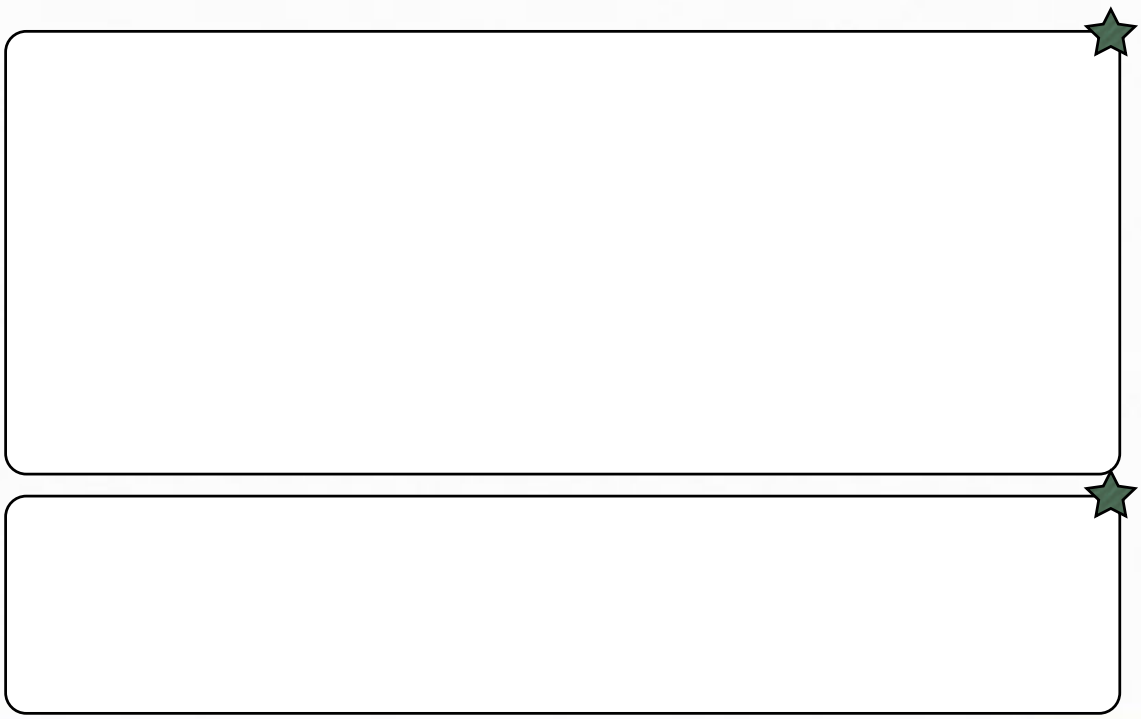


Physique : le fichier est stocké en **binaire** dans des **secteurs** (parfois dispersés) sur des pistes magnétiques du **disque**

Architecture d'un ordinateur

Soulevons le capot

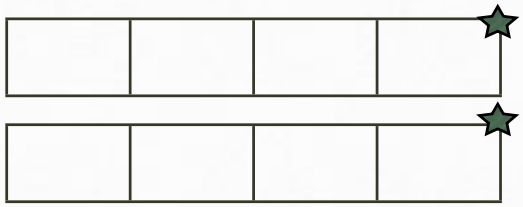
Quels composants contient un ordinateur ?



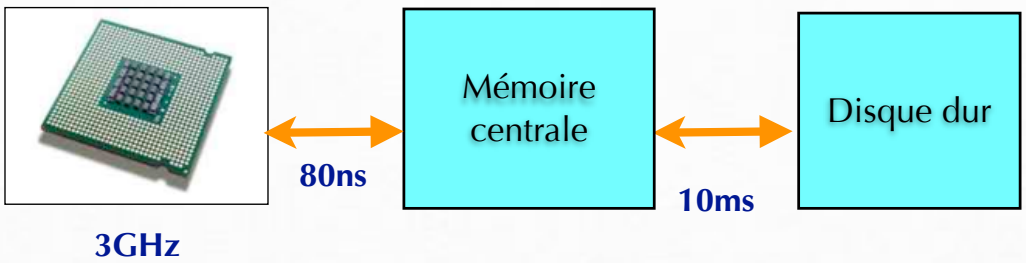
Quelques vitesses de transmission de l'information

Période

Fréquence correspondante

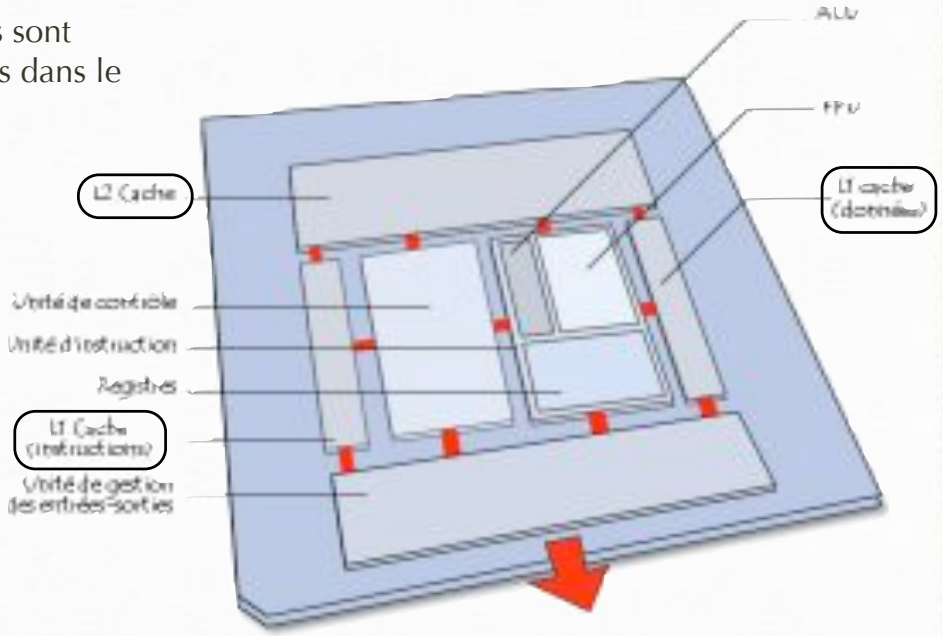


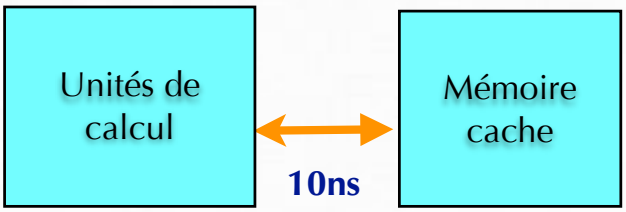
3 GHz : c'est la fréquence d'horloge du processeur
il est capable de faire quelque chose en 1/ 3 000 000 000s
une instruction utile se fera plutôt en 0,01µs
(1 /100 000 000 s)



Quelques vitesses de transmission de l'information

Les mémoire caches sont maintenant intégrées dans le processeur





Quelques vitesses de transmission de l'information

- USB 2.0 : 480 Mbit/s
- USB 3.0 : 5 Gbit/s
- USB 3.1 : 10 Gbit/s (en pratique 7 Gbit/s)
- Ethernet : 1 Gbit/s («Gigabit Ethernet»)
- WIFI 802.11n : jusqu'à 900 Mbit/s
- RAM : 10 à 25 G**o**/s (DDR4)

Système d'exploitation

Un utilisateur ne peut pas interagir directement avec le matériel !

Il lui faut un programme de base qui

- permet aux programmes d'utiliser les périphériques, le réseau, ...
- tourne en permanence dès le démarrage de l'ordinateur
- offre un classement logique des informations stockées
- peut dialoguer avec les humains (interface)
- permet l'exécution et le développement de programmes

Ce programme de base s'appelle un **système d'exploitation**

(*Operating System*)

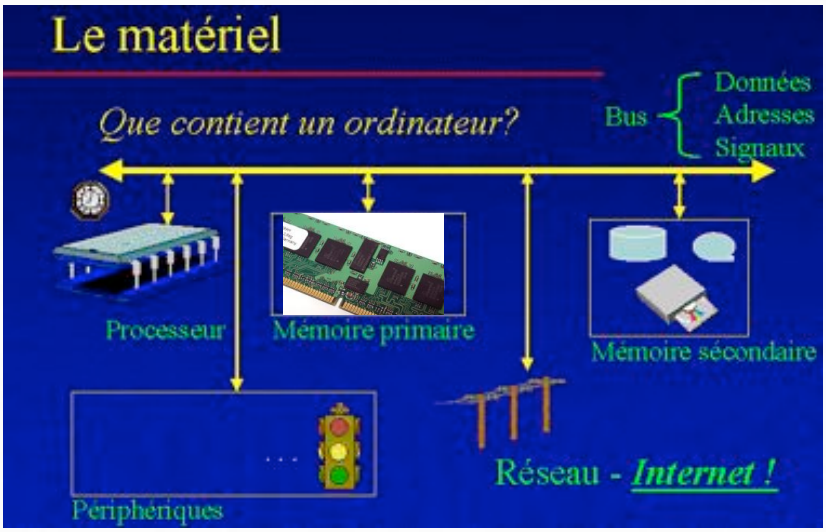
Exemples. : Windows, Linux, Unix, Mac OS X, Symbian OS, Androïd, (chacun ayant plusieurs versions)

Définition d'un SE

Il est impossible de définir simplement un S.E., celui-ci joue plusieurs rôles que nous allons essayer de cerner.

- chef d'orchestre
- gestionnaire
- interprète
- . . .

Chef d'orchestre



- Sous la baguette du SE que l'information transite entre composants matériels
- En cas de panne ou d'erreur matérielle, c'est lui qui doit trouver des solutions, faire que l'ordinateur réponde quand même à l'utilisateur
- C'est le SE qui est en charge de coordonner les composants de l'ordinateur

Gestionnaire

Un S.E. ne produit aucune information ou ressource.
Comme un *gouvernement*, il gère les ressources en arbitrant les accès concurrents, il définit les règles d'accès.

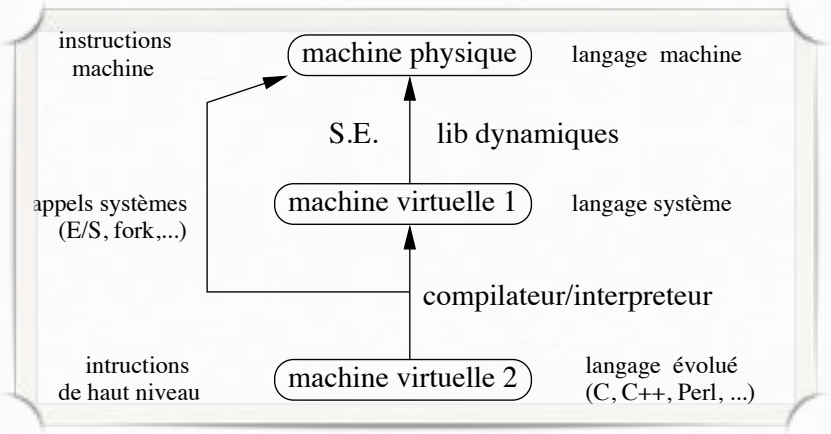
Plus précisément, il aura en charge la gestion

- des programmes en cours d'exécution : chargement en mémoire, accès au processeur, synchronisation
- de l'organisation de l'espace mémoire et de l'organisation des informations sur disque
- des utilisateurs : identification, protection des informations, zones de partages d'information

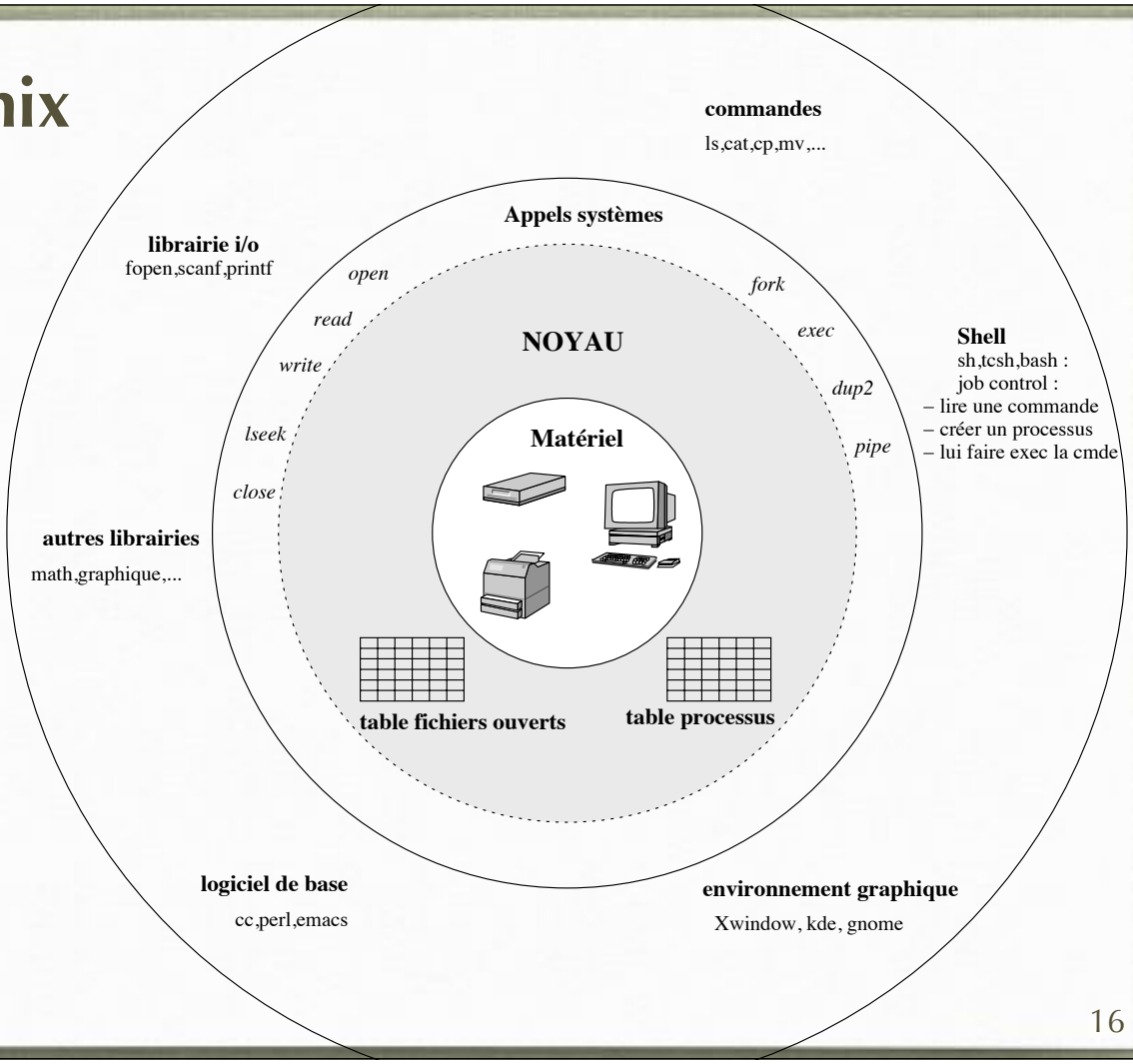
Interprète

Le S.E. a pour rôle de nous affranchir de la complexité du matériel, il sert de traducteur en proposant un/des langage/s de plus haut niveau.

Les applications de niveau supérieur communiquent au S.E. leurs intentions par le biais d'appels systèmes



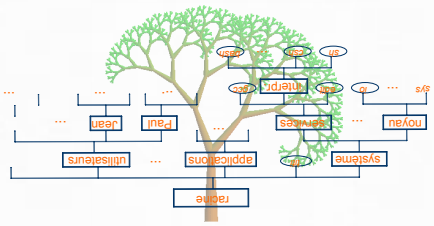
Unix

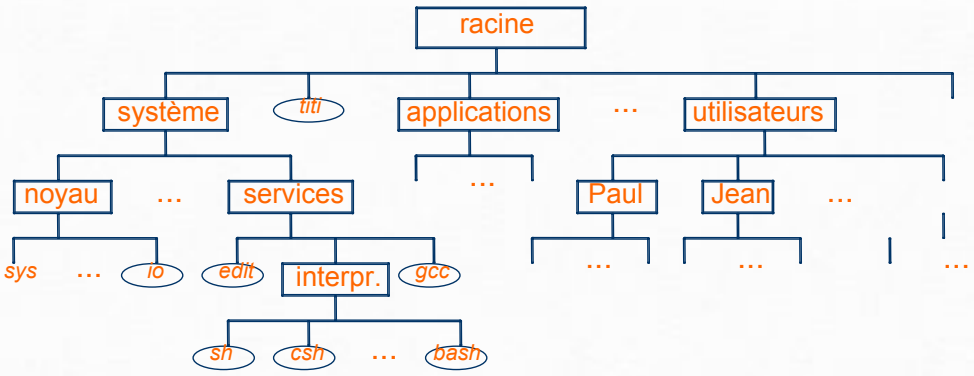

commandes
ls,cat,cp,mv,...
Appels systèmes
librairie i/o
fopen,scanf,printf
open
read
write
lseek
close
fork
exec
dup2
pipe
NOYAU
Matériel
Shell
sh,tcsh,bash :
job control :
– lire une commande
– créer un processus
– lui faire exec la cmde
autres librairies
math,graphique,...
table fichiers ouverts
table processus
logiciel de base
cc,perl,emacs
environnement graphique
Xwindow, kde, gnome

Système de Gestion de Fichiers

Organisation hiérarchique

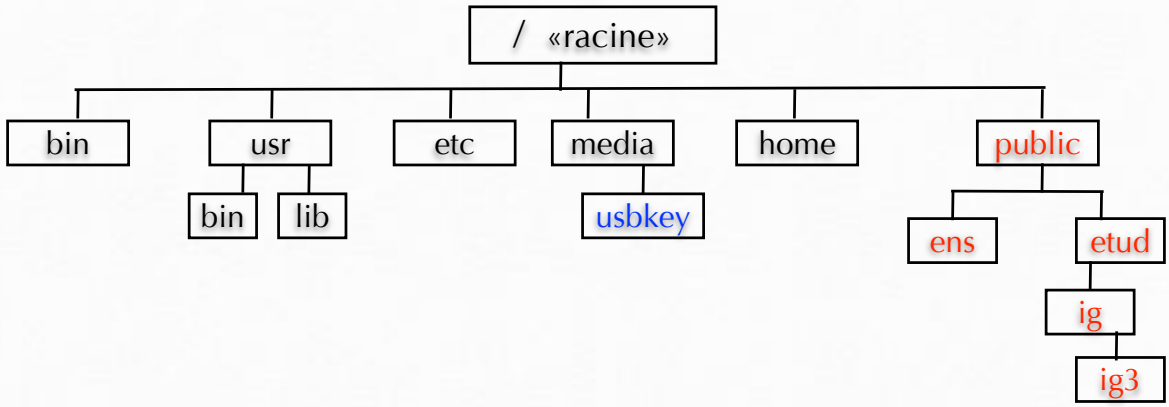
Les **répertoires** & **fichiers** sont organisés sous la forme d'une arborescence :





Montages

Les supports amovibles (zip, cd,dvd,clé USB) et les disques réseaux contiennent aussi une arborescence de fichiers qui est « greffée » sur l'arborescence générale



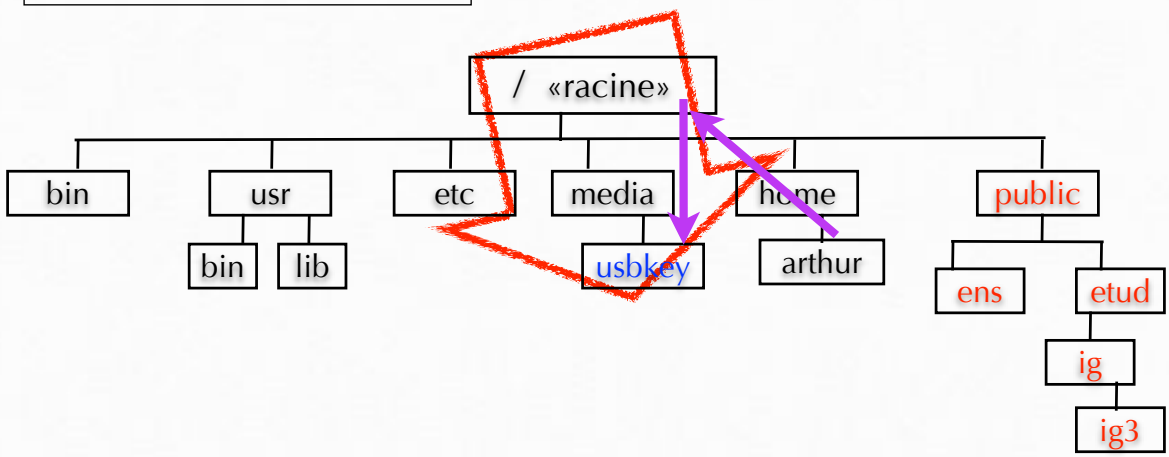
Tout utilisateur a une arborescence personnelle, montée depuis le réseau, greffée dans l'arborescence du système Linux présent sur la machine

`/public` ou `/etudiant`

Désignations

Chaque fichier et répertoire peut-être désigné

- par un chemin absolu (depuis la racine, notée «/»)



- ou par un chemin relatif (depuis un répertoire de travail)

chemin relatif au répertoire de départ

Protection des informations

- Le problème : permettre un partage de fichiers entre utilisateurs mais aussi une protection

- L'ensemble des opérations autorisées sur un objet constitue ses droits d'accès (donnés par le propriétaire)

- Les droits peuvent évoluer dans le temps (*ex : un corrigé lisible par les étudiants*)

Les droits d'accès

Tout fichier ou répertoire a un utilisateur propriétaire et un groupe propriétaire

(par défaut, le créateur du fichier et le groupe auquel appartient le créateur du fichier)

Les droits d'accès à un fichier (ou répertoire) sont paramétrables par le propriétaire du fichier pour trois classes d'utilisateurs (ugo)

- lui-même = le propriétaire du fichier (**u**ser)

- les membres du groupe propriétaire (**g**roup)

- les autres utilisateurs du système (**o**thers)

Les droits d'accès

Trois types d'accès sont possibles (**rwx**). Ces accès ont un sens différents suivant qu'ils s'appliquent à un **répertoire** ou un fichier

- l'accès en lecture (**r**ead)
- l'accès en écriture (**w**rite)
- l'accès en exécution (**ex**ecute)

Les droits d'accès

Les droits des fichiers et répertoires peuvent être affichés par une commande Linux particulière *(cf partie sur le Terminal)*.

```
-rw-rw-r--  ........ fic1.txt
```

10 caractères sont associés à chaque fichier et répertoire :

- le 1er = nature du fichier (**d** pour répertoire et **tiret (-)** pour fichier simple)
- les 9 suivants = droits attribués à ce fichier ou rép.
  (3 pour le propriétaire, 3 pour le groupe, 3 pour les autres)

Chaque groupe de 3 précise dans l'ordre :
les droits en lecture, en écriture, enfin en exécution

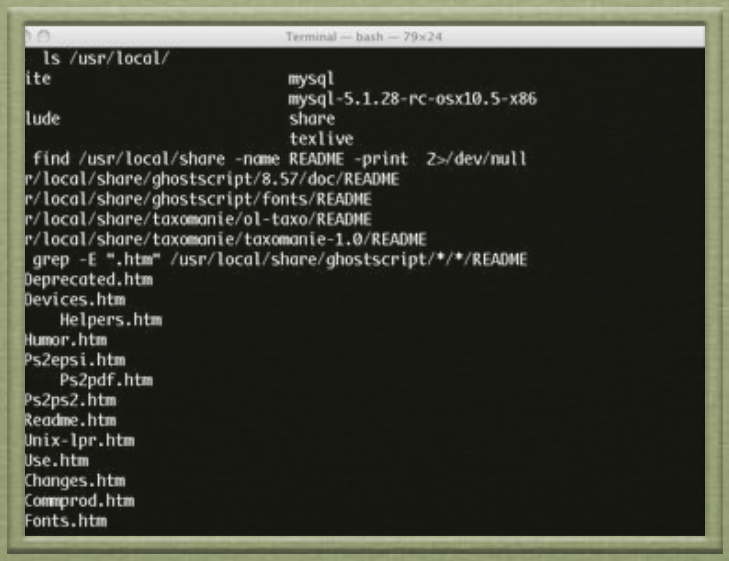
Pour chaque droit, si sa lettre apparait = le droit est donné,
si il y a un tiret (-) = le droit est refusé.

Les droits d'accès

Exemple :

Type droits		prop.	groupe	taille	date_modif.	nom
-rw-rw-r--	1	berry	ens	502	sept 2 9:40	fic1.txt
-rwx--x--x	1	mcvill	ig	205	aout 3 8:03	proj.tar
drwxr-x---	2	fiorio	dir	4096	sept 1 7:35	UNIX

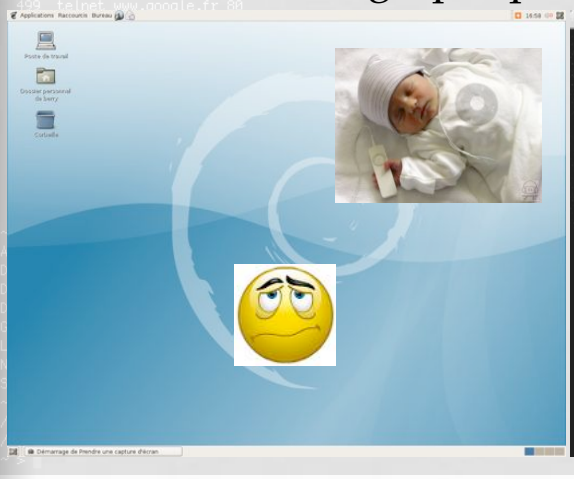
- *fic1.txt* est
- *proj.tar* est
- *UNIX* est un répertoire qui est

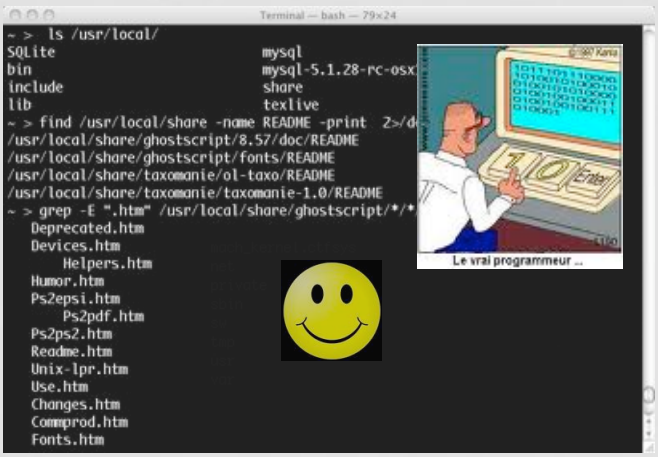


Le terminal de commandes

Le terminal de commandes

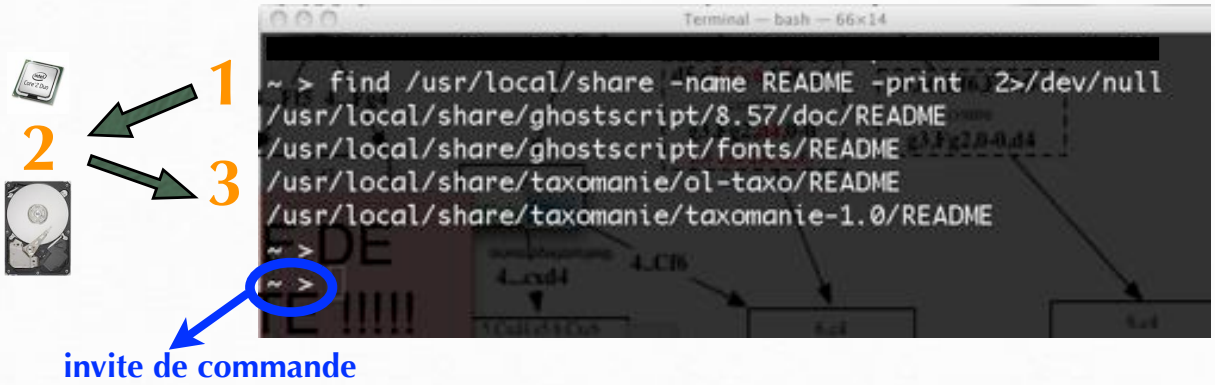
- Un terminal de commande est une application qui permet à l'utilisateur d'interagir avec le SE par le biais de commandes puissantes.
- Cette application permet de lui demander des traitements sophistiqués que l'on ne peut réaliser par le cliquodrome de l'interface graphique.





Le terminal de commandes

- Un terminal de commande exécute une boucle infinie composée de trois actions dans l'ordre :
  1. lire une commande de l'utilisateur
  2. exécuter la commande
  3. indiquer le résultat de la commande



Le terminal de commandes

- Format des commandes :

`nomCde [-option(s)] [argument(s)]`

espace espace

espaces entre chaque option, entre chaque argument !

Exemples:

`date`

`whoami`

affiche le nom de l'utilisateur connecté

`pwd`

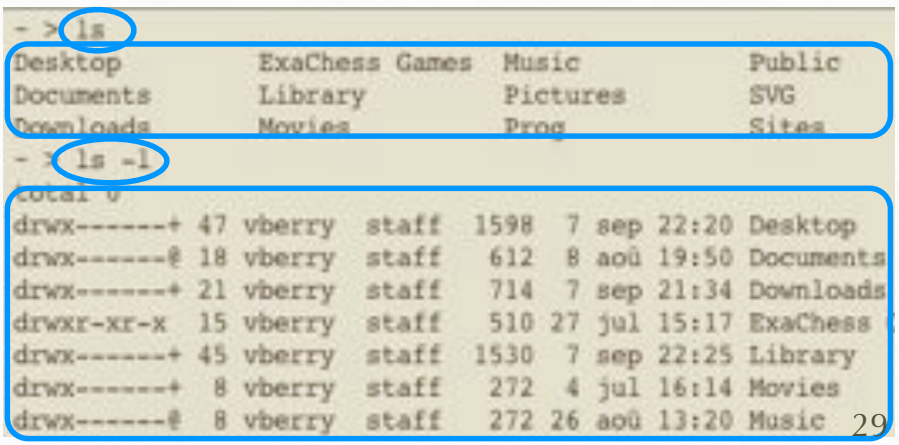
affiche le nom du répertoire courant

`ls -l`

liste le contenu d'un répertoire de façon détaillée

`man <cde>`

manuel en ligne



Commandes sur les fichiers

Changer les permissions sur un fichier : **chmod**

```
chmod <classe op perm, …>|nnn <fic>
```

classe :
- u : user
- g : group
- o : others
- a : all

op :
- = : affectation
- - : suppr.
- + : ajout

perm :
- r : lecture
- w : écriture
- x : exécution

chaque permission = une valeur = l'additon de

r	4
w	2
x	1
aucun	0

Exemples :

- Sur le fichier tp1.tex donner tous les droits à l'utilisateur, et lecture+exécution aux autres entités : chmod u=rwx,go=rx tp1.tex

  OU chmod 755 tp1.tex

- Ajouter les droits d'exécutions à toutes les entités sur le fichier script.pl :

  chmod a+x script.pl

Commandes sur les fichiers

source -> destination

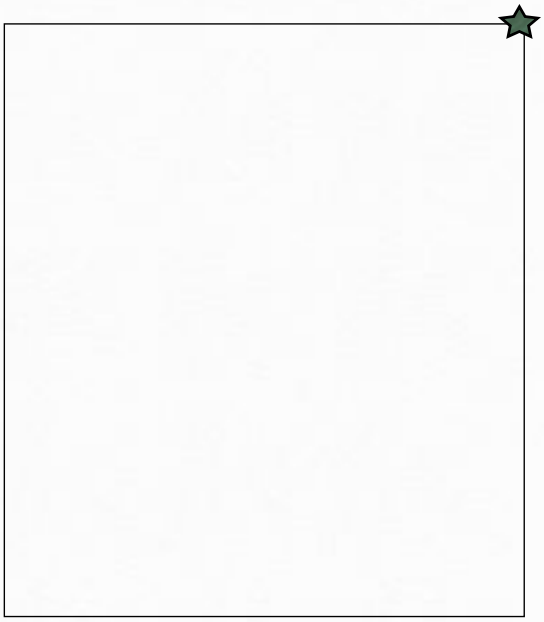
copier

déplacer/renommer

effacer

afficher le contenu

ou



Commandes sur les répertoires

le répertoire d'accueil : **~**

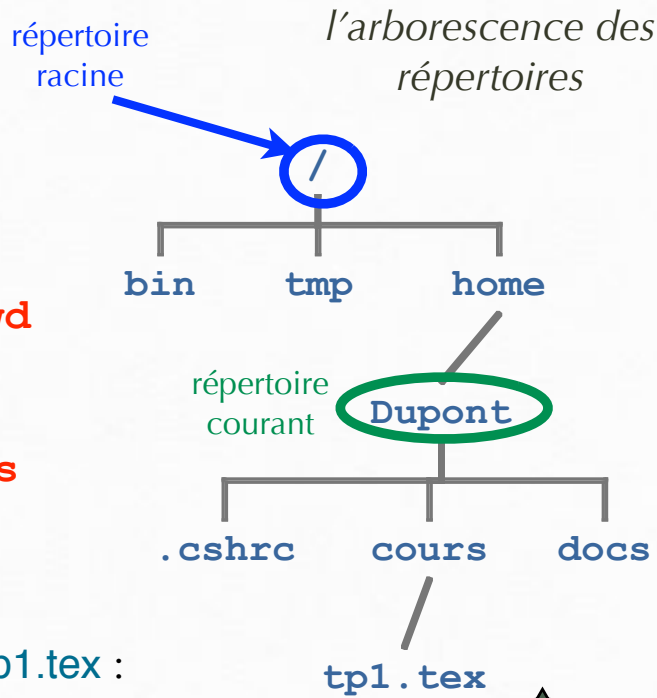
le répertoire courant : **.**

le répertoire supérieur : **..**

connaître le rép. courant : **pwd**

changer de répertoire : **cd**

lister le contenu d'un rép. : **ls**

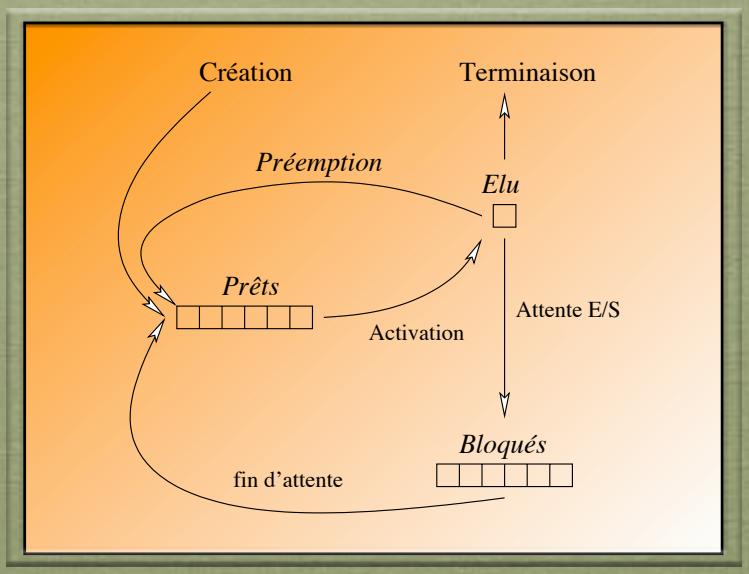


QUIZZ

Chemin d'accès au fichier tp1.tex :

relatif (à la racine):

absolu :



Les processus

Les processus

Un processus est la forme logique sous laquelle le S.E. manipule un programme *en cours d'exécution* (aspect dynamique)

A quoi ça sert-y donc c'truc là ?

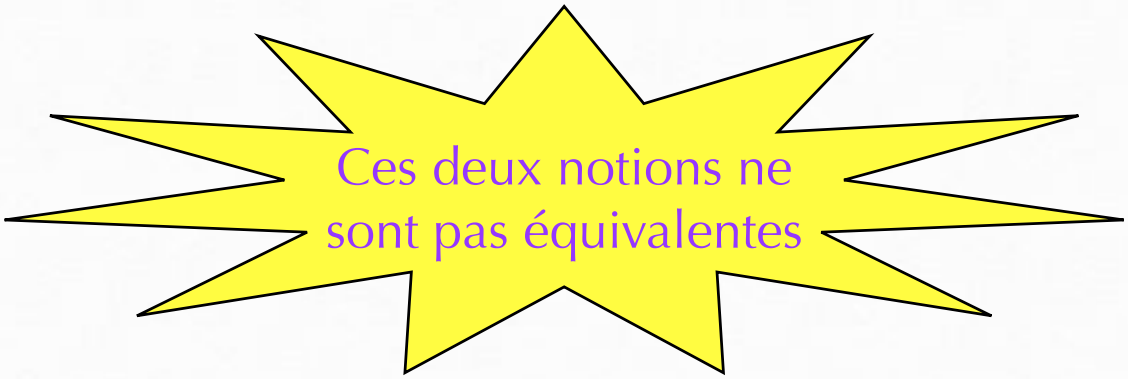
- à faire plusieurs activités en même temps :

  lire les nouvelles sur son navigateur pendant que le courrier se relève dans le gestionnaire de mail et qu'un EDI compile le projet qu'on doit rendre pour le lendemain.

- à permettre à plusieurs utilisateurs d'utiliser un même ordinateur en même temps :

  chacun peut utiliser des applications, quelles soient différentes ou déjà utilisées par d'autres utilisateurs.

processus ≠ programme



- Il y a plus dans un processus que dans un programme :

  vous lancez la commande ps (liste des processus)
  ≠ je lance la commande ps (contexte du user).

- Il y a plus dans un programme que dans un processus :
  - le programme `mozilla` utilise 3 à 4 processus
  - suivant le moment où vous lancez un programme, le nombre de processus créés diffère (*multi-threading*). Exemple = serveur web.

Les processus

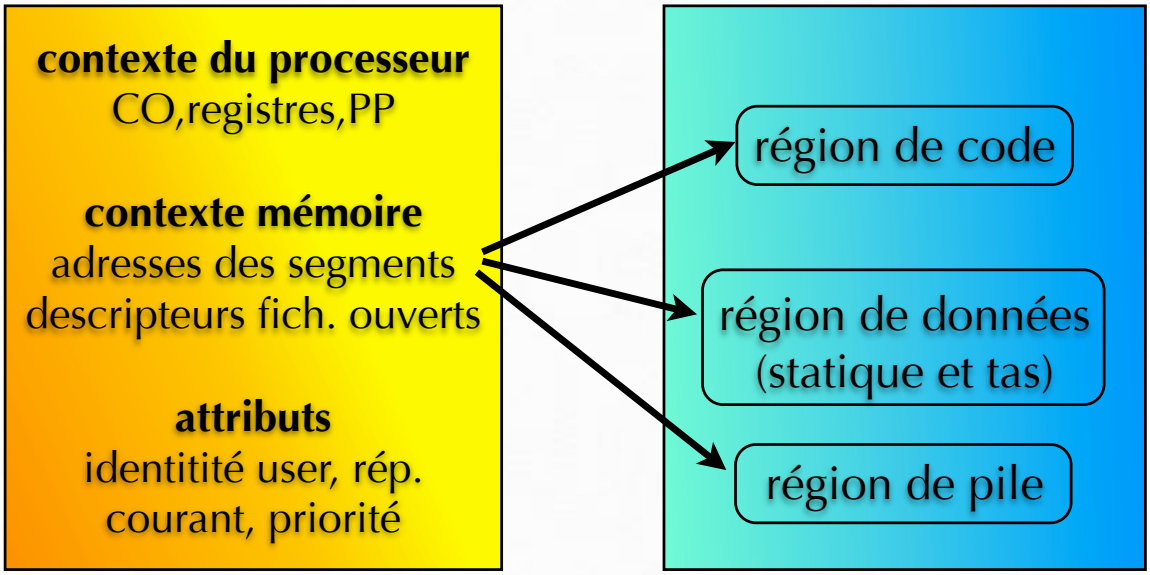
A votre avis, quelles informations le système d'exploitation stocke-t-il pour un processus en cours d'exécution ?

- . . .
- . . .
- . . .
- . . .
- . . .

- . . .
- . . .
- . . .
- …
- …

Les processus

Tout processus a un contexte : l'ensemble des ressources utilisées par le programme pour pouvoir se dérouler



Bloc de Contrôle d'un Processus

Espace d'adressage en mémoire principale

Les processus

Difficultés pour le SE :

- il existe peu de processeurs (souvent 1 à 4) et beaucoup de programmes veulent y accéder en même temps
- chaque processeur n'exécute traditionnellement qu'une seule instruction à la fois*
- le SE doit donc gérer le partage du temps du processeur entre les processus : ordonnanceur.

*même si la technologie superscalaire et superpipeline permettent dans un certain sens de repousser cette limite

Organisation des processus

Cette organisation **dépend de chaque SE** :

- pour ***Unix***, ils sont gérés sous la forme d'une hiérarchie :
  - Cette hiérarchie a une racine.
  - Sur un Linux, le processus `init` est lancé par le noyau à la fin de son initialisation.
  - À son tour, il crée des processus fils pour gérer les différents services du système (les *démons*, ex `sshd`,`httpd`, etc) dont un programme permettant à un utilisateur (ou plusieurs) de se connecter au système.
  - A la connexion une sous-arborescence de processus utilisateurs sont créés.
  - quand un processus se termine, tous ses descendants aussi.
- ***Windows (NT,2000, XP, ...)*** n'ont pas d'arborescence de processus :
  - le gestionnaire de processus demande simplement au gestionnaire d'objets la création d'un nouvel objet de type processus;

Un processus dans tous ses états

Lors de son existence (de son lancement au moment où il se termine), un processus peut passer par différents états :

- **Prêt** : il est prêt à continuer son exécution mais doit attendre qu'un autre processus libère le processeur
- **Actif** : il *possède* le processeur et exécute ses instructions
- **Endormi** : il ne fait rien et attend une condition pour redevenir *prêt* (fin d'une E/S, communication Internet, etc)
- **Zombi** : il a terminé son exécution, et va disparaître dès que son processus parent sera averti.

Un processus dans tous ses états

Lors de son existence (de son lancement au moment où il se termine), un processus peut passer par différents états :

- **Prêt** : il est prêt à continuer son exécution mais doit attendre qu'un autre processus libère le processeur
- **Actif** : il *possède* le processeur et exécute ses instructions
- **Endormi** : il ne fait rien et attend une condition pour redevenir *prêt* (fin d'une E/S, communication Internet, etc)
- **Zombi** : il a terminé son exécution, et va disparaître dès que son processus parent sera averti.

Commandes sur les processus

- Liste des processus : ps

```
[gaston:~] ps
  PID   TT     STAT       TIME    COMMAND
 3899   pts/0   S       0:00.08     -zsh
 4743   pts/0   S+      0:00.14     emacs
 5190     ?     S       0:00.04     mozilla
```

temps CPU utilisé

numéro de processus

Pseudo terminal associé

état du processus:

R	actif (***r**unning*)
T	bloqué
P	en attente de **p**age
D	en attente de **d**isque
S	endormi (***s**leeping*)
IW	swappé
Z	**z**ombie

commande exécutée

Question : que fait l'option -l à votre avis ? et -u ? et -x ?

Liste des processus … et encore plus

D'autres commandes utiles :

- `top`, `pstree` et `htop`
- A propos : combien d'entre vous on un Mac ?
- Certaines commandes manquent par défaut (FreeBSD version de Unix)

➡ **Macport** ou **Brew** pour installer de nouvelles commandes

- Démonstration avec Brew pour htop
- Au fait à quoi sert la commande `sudo` ?

Commandes sur les processus

- Tuer un processus :

  `kill -9 <PID>`

- Certains processus «prennent la main»

  exemple: `find / Archibald -print > liste`

- NB* : Arrêter le processus lancé en avant plan dans un terminal : CTRL-C

- Solution : lancer un processus en arrière-plan : &
  (c-a-d, sans bloquer le terminal)

  NB* : noter la commande open -a <nomApp> params
  (OSX only)

* Notez Bien que

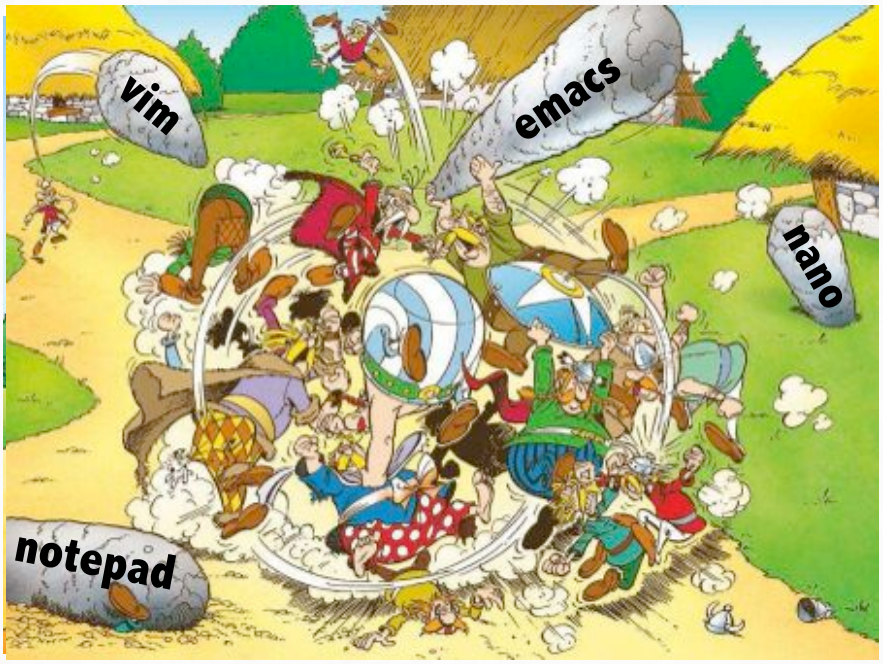
Avant et arrière plan (suite)

- **Interrompre** momentanément un processus lancé dans le terminal en ayant oublié d'indique le «&» : CTRL-Z
- **Relancer** un processus interrompu par CTRL-Z :
  - en avant plan : `fg`
  - en arrière plan : `bg`

L'éditeur nano

Un éditeur de texte

Pour vos travaux sous Unix (hors traitement de texte), il va vous falloir choisir un éditeur de textes.....
Demandons aux informatix-iens ce qu'ils utilisent :



Demander à un informaticien quel éditeur il utilise c'est comme demander s'il est *supporter* de l'OM, de l'OL ou du PSG : ça ne se discute pas !

L'éditeur nano

un éditeur présent sur la plupart des serveurs auxquels on accède à distance

`nano [options] fichier.ext`

L'essentiel :

- CTRL + G : *get help*, **liste des fonctions** / raccourcis
- CTRL + O : **enregistrer** le fichier ('Y' : sauver les changements)
- CTRL + R : **charger** un fichier
- CTRL + X : **quitter** Nano.

Explorateur de fichier rudimentaire mais utile

- (disponible avoir effectué CTRL+O ou CTRL+R) : **CTRL-T**

L'éditeur nano

```
nano [options] fichier.ext
```

Sélection d'un morceau de texte :

- Ctrl + ^ (accent citronfraise) : commencer à **sélectionner** un morceau de texte
- Meta + ^ : **copier** le texte sélectionné *
- Ctrl + K : **couper** le texte sélectionné (sinon la ligne en cours) --> dans le presse-papier) ;
- Ctrl + U : **coller** la ligne de texte que vous venez de couper ;

* Dans nano sur Mac, la touche Meta est Esc

NB : pour se déplacer dans le texte, utiliser les touches flèches du clavier (pas la souris)

L'éditeur nano

Autres fonctions utiles

- **Undo** : après avoir utilisé l'option **-u** au lancement : M-U (undo) et M-E (redo) !! expérimental et > v2.2.6 !!
- Ctrl + W : **rechercher** dans le fichier :
  - sur la ligne du bas, indiquez le mot que vous cherchez
  - la touche «entrée» sans indiquer de mot recherche l'**occurrence suivante** du mot que vous venez de chercher
  - les flèches (haut et bas) vous permettent d'accéder aux **recherches précédentes**

L'éditeur nano

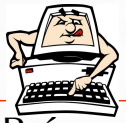
Pour la programmation

- Ctrl + C : afficher la **position** (ligne / colonne) de votre curseur (utile pour le déboggage d'un fichier de code / configuration)
- M + X : gagne de la place en masquant les lignes d'aides en bas de l'écran
- option -i = **indentation automatique** : quand on crée une nouvelle ligne (contexte de programmation)
- **Coloration syntaxique** : inclure les bons fichiers dans le fichier **`.nanorc`**

TP1 et TP2

les exemptés :

IUT Informatique & Gestion (autres IUT + foreigners voir avec moi maintenant / see with me now!)



DEVOIRS

Présence non nécessaire aux séances 1&2 de TP **mais** feuilles de TP lues et supposées résolues pour semaine prochaine

http://www.lirmm.fr/~vberry/teachingFR.html

L'éditeur (x)emacs

L'éditeur (x)emacs

Pour ceux qui veulent un éditeur plus puissant que nano

Cela dit, ce choix forcé vous procurera quelques avantages :

- c'est un bon outil pour la programmation :
  - colorisation syntaxique pour tout langage de prog°
  - indentation du code automatique
  - comparateur de codes
  - outils de débogage de code
  - inclut un terminal de commande
- emacs est gratuit et multi plate-forme (rentabilité de l'effort)
- il est extrêmement paramétrable (raccourcis, modes,...)
- il est bien connu de vos enseignants (assistance, ...)
- il dispose de nombreux sites et forums de fans
- c'est un outil multi usages (calendrier, messagerie, tetris, ...)

L'éditeur (x)emacs

Comme tout éditeur puissant

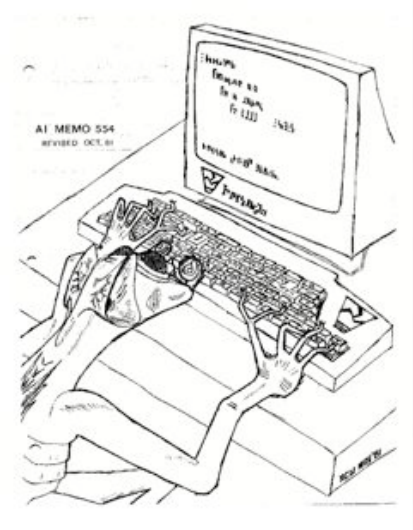
- il peut être difficile à apprendre
- il nécessite de la mémoire
- les raccourcis claviers sont nombreux et parfois **alambiqués**

Surtout la touche CTRL et la touche Meta :

Windows: 

Macintosh: 

Unix: 



Mutation fréquente chez les utilisateurs emacs

Pour vous consoler

- la courbe d'apprentissage monte rapidement .....après les premières séances

Notations des raccourcis claviers

La notation emacs pour les raccourcis consiste à écrire séquentiellement les touches à presser, préfixées des modificateurs.

- Les préfixes standards sont
  - C pour CTRL,
  - M pour Meta (alt sur la plupart des claviers),
  - S pour shift
- **C-x C-f** veut dire "maintenir la touche CTRL, puis appuyer sur x et f",
- **C-x f** signifie "maintenir CTRL, appuyer sur x, lâcher CTRL, appuyer sur f"
- **C-M-f** "appuyer sur f en maintenant les touches CTRL et Meta".

Disséquons l'animal (x)emacs

barre de menus

Buffers

lignes de modes
ou *barres d'état*

minibuffer ou ligne de commande

Vocabulaire (x)emacs

- Tout document est édité dans une zone mémoire appelé buffer
- Chaque *buffer* correspond(ra) à un fichier
- Plusieurs *buffers* peuvent être montrés à l'écran à la fois en séparant la zone d'édition, ou bien être ouverts dans de nouvelles fenêtres (frames)
- **Rappel** : l'édition effectuée à l'écran n'affecte que le buffer en mémoire principale, pour répercuter les changements dans le fichier correspondant sur disque, il faut sauver le buffer :

  menu `Files` puis `Save Buffer`

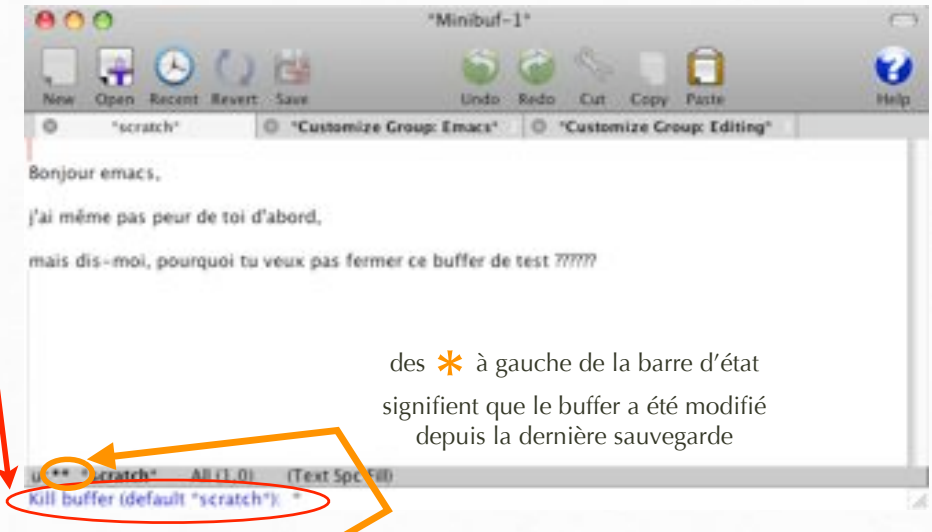
  ou bien

  C-x C-s (Save buffer)

Erreurs les plus fréquentes

- Ne pas remarquer qu'emacs peut vous poser des questions dans le minibuffer

(emacs sur OS X)



..... et quitter emacs en croyant que vous avez sauvé un buffer alors qu'il n'en est rien !

Don't worry

La commande à toujours se rappeler :

C-g

vous sort de toute situation compliquée ou délicate :

demande à emacs d'arrêter ce qu'il est en train de faire

(à utiliser de façon répétée parfois avant de se faire obéir de la bête)

Entrer et sortir d'(x)emacs

Débuter l'édition d'un fichier (Open | New) :

- depuis un terminal :
  `emacs nomfichier`
- depuis emacs :

`C-x C-f` (find file)

Arrêter l'édition d'un fichier :

- depuis le programme emacs, au choix :
  menu `Files` puis `Kill Buffer`

`C-x C-k` (kill buffer)

Quitter emacs :

- depuis le programme emacs, au choix :
  menu `Files` puis `Exit Emacs`
  ou bien

`C-x C-c`
(interrompre Emacs)

Raccourcis d'édition

Mouvements du curseur :

C-d **d**elete char
C-a beginning of line
C-e **e**nd of line
C-o new line (enter)

Copier/Couper/Coller :

C-[space]	set a mark (start of region)
C-w	cut the region from mark to current cursor location
M-w	copy region
C-y	paste region
C-k	cut end of line

Chercher/Remplacer :

C-s find word forward
C-r find word in reverse
Use C-s and C-r to jump forward\back
Press [enter] to stop the cursor on the current match
M-% replace with prompt
M-x 'replace-string' replace without prompting

Les commandes du minibuffer

Le raccourci M-x permet de passer dans le *minibuffer* pour taper des commandes emacs

- On peut ensuite taper une commande emacs comme

  `replace-string`

- La touche TAB assure la complétion automatique, comme dans un *Terminal*

Note : dans le cas de cette commande, Emacs nous demande ensuite dans le minibuffer le mot à remplacer et le mot par lequel il doit le remplacer

Raccourcis d'édition

Travailler avec les buffers :

C-x **b** select another **b**uffer
C-x **k** **k**ill current buffer

Edition multi fenêtres pour comparaison / réutilisation :

C-x **2** split a window in **2** horizontal buffers
C-x 3 split buffer in 2 vertical buffers
C-x 1 delete all other windows
C-x 0 delete this window
C-x o jump to 'other' window (use to cycle through windows).

Fonctions utilitaires pour la programmation :

M-x 'shell' start a shell in a new buffer!
M-x 'dired' start a directory editor/browser
M-x 'compile' compile cmd, also 'recompile' cmd
M-x 'w3m' web browser (see refs)

Pour démarrer .... et aller plus loin

- http://www.linux-france.org/article/appli/emacs/tut/
- http://www.gnu.org/software/emacs/tour/
- http://ensiwiki.ensimag.fr/index.php/Raccourcis_claviers_Emacs