Cours 1
Traitement automatique de corpus

Nicolas Carrara

Lille 3.

Année 2018-2019

Linux

Linux, crée par Linus Torlavd :

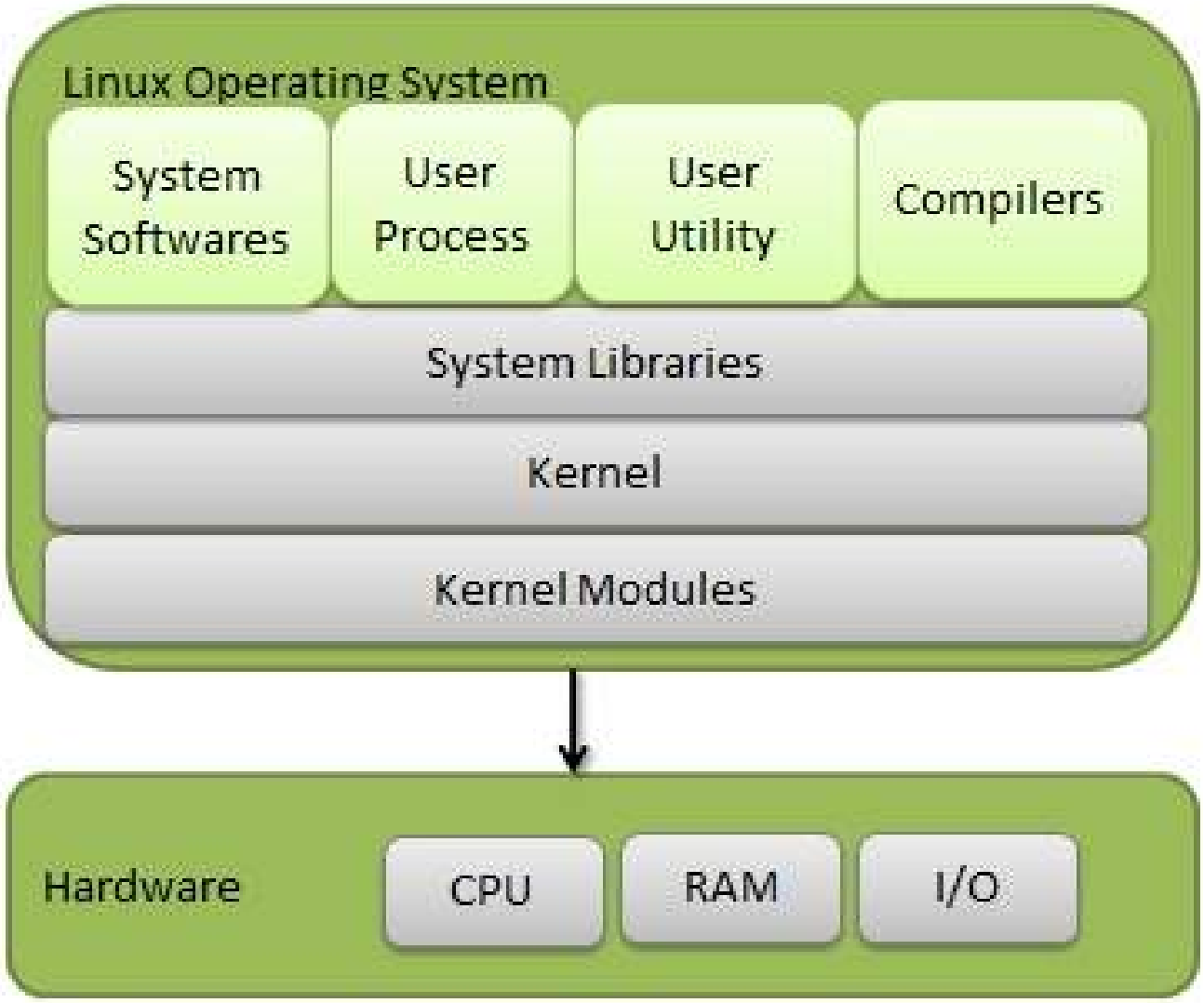
- au sens restreint : le noyau de système d'exploitation Linux. Logiciel libre. Open source.
- Au sens large : tout système d'exploitation fondé sur le noyau Linux (ubuntu, linuxmint ...).
- Plus de 90% des serveurs sont sous Linux.
- Des OS basés sur Linux peuvent être propriétaires.
- Android est un système linux. MacOS et Linux ont des racines communes (Unix)

Architecture d'un système Linux

3 composants :

- Kernel : noyau de Linux. L'interface entre le matériel et le système (pilotes, ordonanceur, réseau ...).
- System library : collection de logiciels déjà compilés, utilisables par des programmes.
- System utility : programmes de plus au niveau pour des tâches de base.

Architecture d'un système Linux



Chemins

On retrouve deux types de chemins sur linux :

- relatif : chemin par rapport à la position du répertoire courant.
- Absolu : chemin qui part de la racine "/".

Les chemins spéciaux :

- ~ : le home (/home/dupont).
- . : le dossier courant.
- .. : le dossier parent.
- / : la racine du système de fichiers.

Les fichiers

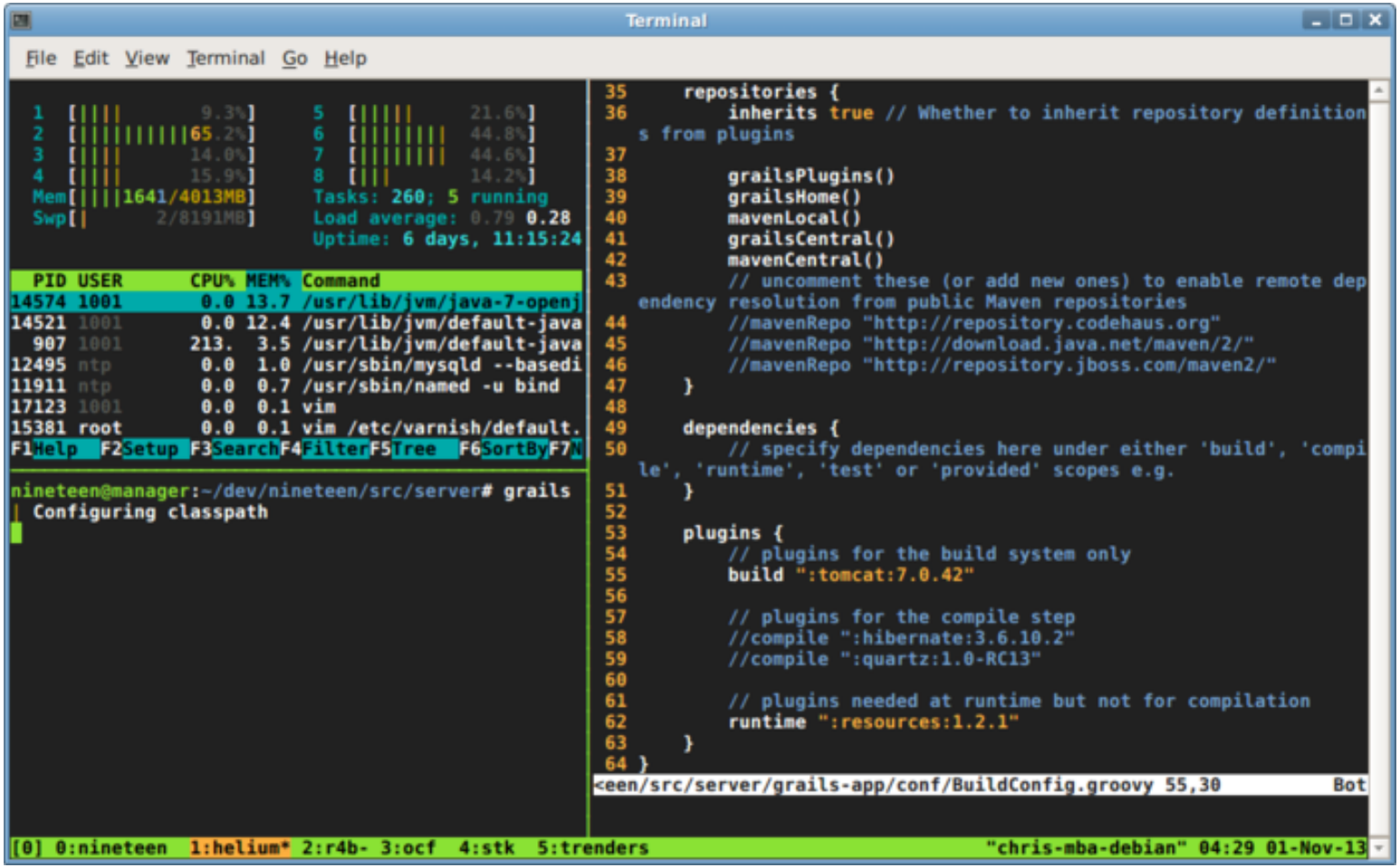
- Tout est fichier sur Linux.
- C'est un OS qui ne tient pas compte des extensions (png, txt jpg ...).
- Sensible à la casse (case sensitive).
- Attention avec les espaces dans les fichiers. Utiliser "mon fichier" ou mon fichier.
- Les fichiers cachés commencent par un point.

Travailler sur un OS linux-based

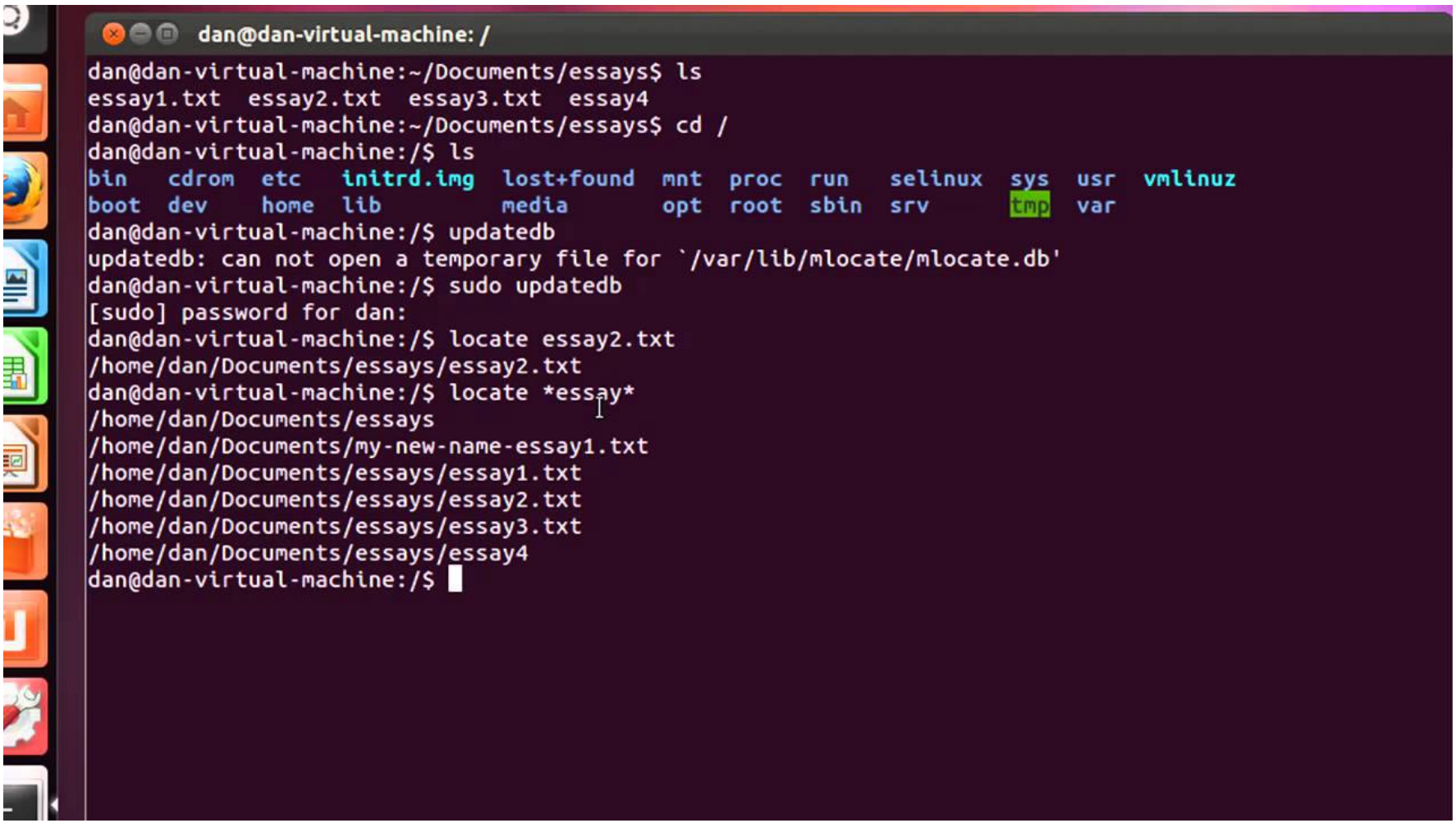
On peut intéragir avec Linux de deux manières (non mutuellement exclusives) :

- Utilisation classique : interface graphique
- utilisation avancée : terminal.

Le terminal


Terminal
File Edit View Terminal Go Help
1 [|||| 9.3%]
2 [||||||||||65.2%]
3 [|||| 14.0%]
4 [|||| 15.9%]
Mem[||||1641/4013MB]
Swp[| 2/8191MB]
5 [||||| 21.6%]
6 [|||||||| 44.8%]
7 [|||||||| 44.6%]
8 [||| 14.2%]
Tasks: 260; 5 running
Load average: 0.79 0.28
Uptime: 6 days, 11:15:24
PID USER CPU% MEM% Command
14574 1001 0.0 13.7 /usr/lib/jvm/java-7-openj
14521 1001 0.0 12.4 /usr/lib/jvm/default-java
907 1001 213. 3.5 /usr/lib/jvm/default-java
12495 ntp 0.0 1.0 /usr/sbin/mysqld --basedi
11911 ntp 0.0 0.7 /usr/sbin/named -u bind
17123 1001 0.0 0.1 vim
15381 root 0.0 0.1 vim /etc/varnish/default.
F1Help F2Setup F3SearchF4FilterF5Tree F6SortByF7N
nineteen@manager:~/dev/nineteen/src/server# grails
| Configuring classpath
35 repositories {
36 inherits true // Whether to inherit repository definition
s from plugins
37
38 grailsPlugins()
39 grailsHome()
40 mavenLocal()
41 grailsCentral()
42 mavenCentral()
43 // uncomment these (or add new ones) to enable remote dep
endency resolution from public Maven repositories
44 //mavenRepo "http://repository.codehaus.org"
45 //mavenRepo "http://download.java.net/maven/2/"
46 //mavenRepo "http://repository.jboss.com/maven2/"
47 }
48
49 dependencies {
50 // specify dependencies here under either 'build', 'compi
le', 'runtime', 'test' or 'provided' scopes e.g.
51 }
52
53 plugins {
54 // plugins for the build system only
55 build ":tomcat:7.0.42"
56
57 // plugins for the compile step
58 //compile ":hibernate:3.6.10.2"
59 //compile ":quartz:1.0-RC13"
60
61 // plugins needed at runtime but not for compilation
62 runtime ":resources:1.2.1"
63 }
64 }
<een/src/server/grails-app/conf/BuildConfig.groovy 55,30 Bot
[0] 0:nineteen 1:helium* 2:r4b- 3:ocf 4:stk 5:trenders "chris-mba-debian" 04:29 01-Nov-13

Le terminal


dan@dan-virtual-machine: /
dan@dan-virtual-machine:~/Documents/essays$ ls
essay1.txt essay2.txt essay3.txt essay4
dan@dan-virtual-machine:~/Documents/essays$ cd /
dan@dan-virtual-machine:/$ ls
bin cdrom etc initrd.img lost+found mnt proc run selinux sys usr vmlinuz
boot dev home lib media opt root sbin srv tmp var
dan@dan-virtual-machine:/$ updatedb
updatedb: can not open a temporary file for `/var/lib/mlocate/mlocate.db'
dan@dan-virtual-machine:/$ sudo updatedb
[sudo] password for dan:
dan@dan-virtual-machine:/$ locate essay2.txt
/home/dan/Documents/essays/essay2.txt
dan@dan-virtual-machine:/$ locate *essay*
/home/dan/Documents/essays
/home/dan/Documents/my-new-name-essay1.txt
/home/dan/Documents/essays/essay1.txt
/home/dan/Documents/essays/essay2.txt
/home/dan/Documents/essays/essay3.txt
/home/dan/Documents/essays/essay4
dan@dan-virtual-machine:/$

Navigation (textuelle)

Différentes commandes permettent la navigation dans le terminal :

- pwd : print working directory
- ls : list, ls -a pour afficher les fichiers cachés avec. ls -l pour les détails.
- cd : change directory

Manipulation de fichier

Les commandes de manipulation de fichiers :

- rm : remove
- rmdir : remove directory
- file : connaître l'extension d'un fichier
- touch : créer un fichier five
- cp : copy
- mv : move
- ln -s : créer un lien symbolique

Si vous supprimez un fichier via rm, pas de deuxième chance avec la corbeille.

Les wildcards

Des caractères spéciaux sont utilisables pour décrir des fichiers :

- * : zero ou un caractère
- ? : un seul caractère
- [] : un ensemble de caractère

Les flux de redirections | la sortie standard

A la suite d'une commande, au lieu d'afficher la sortie standard dans la console, on peut

- Dump les résultats dans un nouveau fichier : >
- ou à la fin d'un fichier : >>.

Les flux de redirections | la sortie d'erreur

On peut aussi faire la distinction entre la sortie standard et la sortie d'erreurs :

- `2>` (ou `2>>`) pour les erreurs.
- `2>&1` pour rediriger les erreurs de la même façon que la sortie standard (fusion).

Les flux de redirection | exemples

- `ls -l > resultat_ls.txt`
- `cat fileunknown.csv > std.txt 2> err.log`
- `cat fileunknown.csv >> std.txt 2>&1`

Les flux de redirections : les entrées

On peut spécifier l'entrée d'une commande avec un fichier ou depuis le clavier

- < : depuis un fichier
- << : depuis le clavier.

Exemples :

- `cat < a_afficher.txt`
- `sort -n <<`. Trier des nombres, taper FIN pour terminer la saisie.

Les flux de redirections : chaîner les commandes

On peut aussi rediriger vers une autre commande avec la barre : |. Par exemple :

- `ls -l | grep '-rw-rw-r--'`
- `cat fichier.txt | grep 'salut'`

Les permissions

Les fichiers sont rigoureusement protégés avec un système de permissions.
Différentes permissions :

- r : read
- w : write
- x : execute

Différents groupes :

- owner : une seul personne
- group : un groupe de personnes
- others : tout le reste

Utiliser la commande chmod pour modifier les permissions.
Utiliser ls -l pour connaitre les propriétaires du fichier (ownver, group)

Les permissions

Par exemple

- chmod u+x file.txt
- chmod g-w file.txt
- chmod 775 file.txt (mode binaire)

Le language SH

Exécuter des instructions du terminal avec le language de script SH (shell script).

- Créer un fichier test.sh
- Ajouter un entête #!/bin/sh, c'est le chemin (commenté) de l'interpréteur.

Astuces terminal

- sudo : avant une commande, passe en mode admin.
- man (manual) : manuel d'une commande (shell ou fonction C)
- tab : tab completion
- fleche du haut : historique
- ctrl+r : chercher dans l'historique
- La commande clear : vider visuellement le terminal courant
- La commande exit : ferme le terminal
- ctrl+alt+t : ouvre un nouveau terminal (sur ubuntu seulement)

Maintenance via terminal

- apt-get install <package> : installe un logiciel, une bibliothèque ...
- apt-cache search <package> : chercher dans les paquets dispos.
- apt-get update : mets à jour à la liste des paquets.
- apt-get upgrade : mets à jours les paquets (et le système).

Ressources

- https ://ryanstutorials.net/linuxtutorial/

Regex

Une expression régulière (regex, Stephen Cole Kleene 1956) est :

- une chaîne de caractère.
- Elle décrit un ensemble de chaîne de caractères possible.
- Les wildcards (*,[] et ?) sont une forme très simplifiée des regex.

Regex | Exemples

- Le pattern ex-(a ?e|æ|é)quo : ex-équo, ex-equo, ex-aequo et ex-æquo
- Le pattern 6,66*$: 6,6, 6,666 , 6,6666 , ...

Regex | Les quantificateurs

- ? : zéro ou une occurence de l'expression précédente
  - Chiens ? : Chien, Chiens
  - 11 ?0 ? : 110, 11 ,10
- * : zéro ou plusieurs occurence de l'expression précédente
  - gooo*gle : google, gooogle, goooogle ...
- + : une ou plusieurs occurences de l'expression précédente
  - goo+gle : google, gooogle, goooogle ...
- . : un seul caractère
  - ...DarkVador... : XxXDarkVadorXxX, 666DarkVadorXxx, MdrDarkVador_59

Regex | Autres opérateurs

- () : groupement de caractères
  - Etudiant(es) ? : Etudiant, Etudiantes
- | : le choix entre deux expressions
  - a|b : a,b
  - Etudiant(es|s) : Etudiants, Etudiantes
- [] : liste de caractères
  - [0123456789] : 0,1,2 ...
  - [^*aeiouy*] : 0,b,z, ! ...
- $\{n\}$: précise le nombre d'occurences de l'expression
  - $bo\{2\}h!$: booh !
  - $bo\{2,3\}h!$: booh !, boooh !
  - $bo\{3,\}h!$: boooh !, booooh !, . . . , booo ∞ ooooh !

Regex | caractères spéciaux

- ^ : début de ligne
- $: fin de ligne.
- \ : métacaractère
  - \$: le caractère dollard.
  - \) : la parenthèse fermante.
- On peut aussi utiliser [] pour échapper les métacaractères (sauf^) :
  - [.?*] : . ? ou *
  - [\^*abc*] :^,a,b ou c.

Regex | classe de caractère

Elles dépendent du moteur de Regex. Exemples :

- Caractère alphanumérique en POSIX : `[:alnum:]`,
- en java $\backslash p\{Alnum\}$
- et en ASCII : A-Za-z0-9

Utilisation :
Voir la page wikipédia pour plus de détails sur les classes.

Regex | le standard POSIX

POSIX est une famille de normes techniques. Deux types de normes pour les regex

- BRE (« Basic Regular Expression ») : $\{\}$,(), ?,+ ne sont pas des métacaractères. Utiliser $\backslash$ pour les rendre méta.
- ERE (« Extented Regular Expression ») : tout ce dont on a parlé précédement.

Regex | Les utilitaires

Différentes commandes linux utilisent des regex :

- grep : historiquement :g/re/p sur ed (éditeur de texte), global regexp print.
- sed : stream editor, édite un flux séquentiel de données textuelles (ligne par ligne).
- rename : renommer un fichier.

Par defaut, grep et sed sont en BRE. Modifiable vers ERE avec une option : -E pour grep et -r pour sed. Rename est en ERE.

Python, Java, vim ... possèdent des moteurs de Regex.

grep

Exemples d'execution de grep

- cat fichier.txt | grep 'bonjour'
- cat fichier.txt | grep '\(bonjour\)|\(bonsoir\)'
- cat fichier.txt | grep -E '(bonjour)|(bonsoir)'

sed

Les commandes :

- s : substitute, remplace un pattern par un autre.
- g : global, applique la modification sur toutes les occurences du pattern.

Exemples d'execution de sed

- sed 's/day/night/g' old >new
- sed 's/[0123456789]/ceci_est_un_chiffre/' old >new
- sed 's/\(oe\)/œ/' <test.txt
- sed -r 's/(oe)/œ/' <test.txt

Pour dans deux semaines :

- Lire le cours python
  https ://www.w3schools.com/python/. Lire seulement la section "python tutorial" jusqu'à "Python Functions" inclue.