GREP, REGEX et SED

Julien Loiseau

2 mars 2017

Grep

```
$ grep [regex] [fichier]
$ grep toto fic.txt
```

-v	Affiche les lignes ne contenant pas la chaîne
-c	Compte le nombre de lignes contenant la chaîne
-n	Chaque ligne est numéroté
-x	Ligne correspondant exactement à la chaîne
-l	Nom des fichiers contenant la chaîne
-i	Ignorer la casse
-o	N'afficher que les occurrences valides
-E	Version Extended

Ne pas hésiter à décomposer :

```
$ grep toto fic.txt | grep titi | grep tutu
```

Regex

Se repérer dans la ligne :

^	Début de ligne
.	Une fois n'importe quel caractère (même newline)
$	Fin de ligne.

```
$ grep '^.$' fic.txt
```

Gérer des ensembles de caractères :

[*abc*]	Liste des caractères abc
[^*abc*]	Tous les caractères sauf abc

```
$ grep '[abcdefghijklmnopqrstuvwxyz]' fic.txt
```

Regex

Liste avec intervalle :

```
$ grep [a-z] fic.txt
```

Listes prédéfinies :

[[: *alnum* :]]	Caractères alphanumériques
[[: *alpha* :]]	Caractères alphabétiques
[[: *spaces* :]]	Espaces
[[: *blank* :]]	Espaces ou tabulations
[[: *cntrl* :]]	Caractères de contrôle
[[: *digits* :]]	Caractères numériques
...	...

Regex

Quantifieurs :

$\emptyset$	Une occurrence du caractère
*	Zéro et plus
\{*i*\}	Exactement i séquences avec $i \in [0, 255]$
\{*i*,*j*\}	Entre i et j séquences (inclus)
\{*i*,\}	i ou plus séquences

Extension GNU :

\+	Un et plus
\ ?	Zéro ou un
\<	Début du mot (Attention portabilité)
\>	Fin du mot
\1 \2	Première ou seconde occurrence

La sélection se fait toujours par les chaînes les plus grandes possibles.

Regex

Plusieurs expressions :

*regexp*1| *regexp*2	Occurrence de *regexp*1 ou de *regexp*2
*regexp*1 *regexp*2	Concaténation des deux expressions régulières

SED

La commande s'utilise de la manière suivante :

```
$ sed [script] [flux]
$ cat fic.txt | sed [script]
$ grep fic.txt | sed [script]
$ sed [script] < fic.txt > fic2.txt
$ grep fic.txt | sed [script] | grep [script]
```

Sortie par défaut sur stdout

On a plusieurs opérateurs, les délimiteurs sont les '/' avec '/../../' :

d	Delete, supprimer les lignes de la restriction
p	Print, dupliquer les lignes si pas l'option -n
q	Quit, quitte lorsque la condition est vérifiée
s	Substitution
i	Insert newline avant
a	Insert newline après
c	Change la ligne
g	Global, toutes occurrences

'!' pour inverser les opérateurs

On travaille avec des expression régulières :

```
$ sed '/regex/ d' file.txt
```

Supprimera par exemple les occurences de 'regex'

```
$ echo "toto␣toto" | sed 's/toto/truc/'
```

```
$ echo "toto␣toto" | sed 's/toto/truc/g'
```