Expressions rationnelles

Table des matières

1 Introduction

Le filtrage de données en entrée est une étape fondamentale du développement sécurisé. Les données reçues sont vérifiées suivant plusieurs critères :

- la longueur et
- le contenu.

Dans l'état de l'art, toutes les données en entrée doivent être décrites en terme de longueur et de contenu. Un langage permettant de décrire ces règles sont les expressions rationnelles (regular expressions).
Deux langages principaux d'expressions rationnelles existent : **POSIX** et **perl**. Ici seul **POSIX** sera vu correspondant à **regex(7)**, mais **perl** fonctionne de la même façon avec seulement certaines notation qui changent, voir **pcre(3)**.

2 Syntaxe POSIX

La grande majorité des caractères correspondent à eux même, donc la chaîne « **foo** » correspond à la chaîne « **foo** » dans la chaîne « **brown foox.** ».
Un caractère spécial doit être échappé afin d'être détecté, par exemple « **\.** » correspond à « **.** » dans « **brown foox.** ».
Note : pour tester sur une ligne de commande il est possible d'utiliser la commande suivante :

```
echo 'brown foox.' | egrep '\.'
```

La sortie comprend la chaîne avec les caractères des sous-chaînes qui correspondent à la recherche en rouge.
Il est également possible de mettre la chaîne de caractères dans le fichier « **chaine** », l'expression dans le fichier « **RE** » et d'utiliser la commande suivante :

```
egrep -f RE chaine
```

et la sortie est la même que précédemment.
En ajoutant l'option **-o**, le programme n'affiche que les sous-chaînes correspondant à la recherche, une sous-chaîne par ligne

```
egrep -f RE -o chaine
```

Le caractère « **.** » remplace n'importe quel caractère, par exemple « **o.** » correspond à « **ow** » et « **oo** » dans « **brown foox.** ».
Le caractère « **^** » correspond au début de ligne, par exemple « **^.** » correspond à « **b** » dans « **brown foox.** ».
Le caractère « **$** » correspond à la fin de ligne, par exemple « **.$** » correspond à « **.** » dans « **brown foox.** ».
Le caractère « **+** » correspond à 1 ou plusieurs occurrences.
Le caractère « * » correspond à 0, 1 ou plusieurs occurrences.
Le caractère « **?** » correspond à 0 ou 1 occurrence.

Les caractères « **(** » et « **)** » permettent de créer des groupes.
Par exemples « **fo*** » correspond à « **foo** » car « ***** » ne s'applique qu'à « **o** » alors que « **(fo)*** » correspond à « **fo** » car « ***** » s'applique à « **fo** ».
Les caractères « **[** » et « **]** » permettent de définir des listes.
Par exemple « **of*** » correspond à « **o** », « **o** » et « **o** » car « ***** » ne s'applique qu'à « **f** » alors que « **[of]*** » correspond à « **o** » et « **foo** » car « ***** » s'applique à tout caractère « **o** » ou « **f** ».
Si le caractère « **-** » se trouve entre deux caractères alors il s'agit de tous les caractères du premier inclus au dernier inclus.
Par exemple « **[f-r]*** » correspond à « **ro** », « **n** » et « **foo** » car s'applique à toute suite de caractères de « **f** » à « **r** »
Il existe des ensembles prédéfinis notés entre « **[:** » et « **:]** » :

- alpha : caractères alphabétiques
- digit : caractères numériques
- alnum : caractères alpha-numériques
- lower : lettres minuscules
- upper : lettres majuscules
- print : caractères imprimables
- space : caractères d'espace
- punct : caractères de ponctuation

Par exemple :

```
[[:alpha:]]*
```

correspond à « **brown** » et « **fox** » car s'applique à toute suite de lettres.
L'intérêt d'utiliser ces classes prédéfinies est que les caractères correspondant dépendent de la langue, ainsi avec **LC_ALL=fr_FR** la classe alpha comprendra les lettres accentuées et autres.

```
$ echo 'azérty' | egrep -o '[[:alpha:]]'
a
z
r
t
y
$ echo 'azérty' | LC_ALL=fr_FR egrep -o '[[:alpha:]]'
a
z
é
r
t
y
```

Si le premier caractère d'une liste est « **^** » alors la liste contient tous les caractères sauf ceux de la même liste sans ce caractère. Par exemple « **[^f-r]*** » correspond à « **b** », « **w** », « » et « **x.** » car s'applique à toute suite de caractères autres que ceux de « **f** » à « **r** »
Les caractères « **{** » et « **}** » permettent de définir des répétitions.
Par exemple « **^.{2}** » correspond à « **br** » car elle correspond à 2 caractères en début de chaîne. Il n'y a pas de correspondance si la chaîne fait moins de 2 caractères.

Par exemple « **^.{,2}** » correspond à « **br** » car elle correspond à 0 à 2 caractères en début de chaîne.
Par exemple « **^.{2,}** » correspond à « **brown foox.** » car elle correspond à au moins 2 caractères en début de chaîne.
Par exemple « **^.{2,4}** » correspond à « **brow** » car elle correspond à entre 2 et 4 caractères en début de chaîne.
Note : les expressions rationnelles consomment autant de caractère qu'il est possible d'en consommer. Dans l'exemple précédent elles en consommeront 4 s'il y en a au moins 4.
Par exemple « **o.*o** » correspond à « **own foo** ». Pour s'arrêter au premier « **o** » il faut utiliser « **o[^o]*o** » qui correspond à « **own fo** ».

3 Exemples

Un code postal français est composé de 4 à 5 chiffres :

```
^[0-9]{4,5}$
```

les caractères « **^** » et « **$** » en début et fin sont importants car la chaîne n'est composée que de ces 4 à 5 caractères, ce qui n'est pas explicitement dit mais est sous-entendu.
Un login est composé de 1 à 8 caractères alpha-numériques commençant par une lettre

```
^[:alpha:][[:alnum:]]{0,7}$
```

Un nom de domaine est composé d'une suite d'au moins deux « noms » séparés par un point. Un « nom » est une suite de caractères alpha-numériques plus le tiret sauf le premier qui est une lettre.

```
^[[:alpha:]][-[:alnum:]]*(\.[[:alpha:]][-[:alnum:]]*)+$
```

Une adresse email est un nom d'utilisateur, suivi d'une arobase, suivi d'un nom de domaine. Un nom d'utilisateur est composé de lettres, de chiffres et des caractères « . » et « - », et commence par une lettre.

```
^[[:alpha:]][-.[:alnum:]]*@[[:alpha:]][-[:alnum:]]*(\.[[:alpha:]][-
[:alnum:]]*)+$
```