Mémo sur les expressions régulières

// under regexp

// Par Sacha Schutz

Les expressions régulières, c'est super simple... Il suffit que quelqu'un les écrivent pour vous !
En effet, j'ai longtemps galéré avec les expressions régulières.. En fait, dès que j'avais affaire à elles, je demandais à quelqu'un de me l'écrire. C'était beaucoup plus rapide que de réfléchir par moi même! (technique souvent employée en programmation). Mais voilà, ça c'était avant !

Définition

Les expressions régulières permettent d'identifier dans un texte, des sous ensembles respectant un pattern particulier. Par exemple, imaginons que dans un article scientifique, je souhaite récupérer tous les noms d'auteur, sachant que le nom et le prénom commencent par une majuscule . Par exemple **Ishigaki S** ou **Rossini-Beri AA**. Je sais aussi que les noms ne contiennent jamais de chiffre ni de caractères spéciaux mis à part le tiret. L'expression régulière que j'écrirai sera la suivante :

```
\s[A-Z][a-z]+(\-[A-Z][a-z]+)?\s[A-Z]{1,2}
```

Incompréhensible n'est ce pas ? Ne vous inquiétez pas, je vais vous expliquer. Mais il va falloir pratiquer. Je vous conseil d'aller sur regexpal pour tester en ligne vos expressions régulières.

Expression simple

Le pattern le plus simple, est un groupe de lettre. Dans le texte ci dessous , je recherche le pattern 'biologie'.

exemple

> Les recherches alliant physique quantique, ingénierie électrique, chimie et **biologie**, sont particulièrement pertinentes, car elles pourraient donner naissance à des thérapies entraînant beaucoup moins d'effets secondaires que les médicaments.

Attention, ce n'est pas un mot que je cherche, mais un pattern. Ainsi, le pattern [et] me retournera le "*et*" seul et le "*et*" de "*effet*"

exemple

> Les recherches alliant physique quantique, ingénierie électrique, chimie **et** biologie, sont particulièrement pertinentes, car elles pourraient donner naissance à des thérapies entraînant beaucoup moins d'eff**et**s secondaires que les médicaments.

Caractère spéciaux

Il existe plusieurs caractères reconnus dans les expressions régulières permettant de faire des recherches plus complexe:

```
^ . [ ] $ ( ) * + ? | { } \
```

Vous ne pouvez donc pas rechercher ces symboles directement dans le texte. Pour cela, vous devez les "*échapper*" avec l'antislash [\ ]. Par exemple pour rechercher le pattern ' WTF???? '

```
WTF\?\?\?\?
```

Voyons maintenant la signification des autres caractères spéciaux ...

Le point

Le point représente n'importe quels caractères. Si par exemple vous voulez rechercher le mot 'ARN' et le mot 'ADN', le pattern sera :

```
A.N
```

Attention, quand je dis tous les caractères, c'est tous les caractères possibles! Ce pattern détectera aussi 'ATN' 'A-N' 'A?N' 'A.N' etc... Pour pouvoir détecter uniquement 'ARN' et 'ADN', on utilise des classes de caractères.

Les classes de caractères

Une classe de caractères représente toutes les substitutions autorisées dans notre pattern. Une classe est écrite à l'aide des crochets **[ ]** et contient la séquence substitutif. Pour détecter soit le mot ADN ou ARN :

```
A[RD]N
```

La classe '[RD]' signifie : *A cette endroit, le caractère est soit la lettre R, soit la lettre D*. Simple non ? Mais maintenant, si au lieu de R et D, vous voulez toutes les lettres de l'alphabet ?

```
A[ABCDEFGIJKLMNOPQRSTUVWXYZ]N
```

Ohé... Bein là, ça commence à faire lourd! Heureusement, les classes de caractères connaissent leurs alphabets. Cette expression peut être écrite :

```
A[A-Z]N
```

On peut faire la même chose pour les minuscules et les chiffres.

```
[a-z]  de a à z
[0-9]  de 0 à 9
[b-k]  de b à k
[2-5]  de 2 à 5
```

Bien entendu, on peut tout combiner.

```
[a-zA-Z] de a à z et de A à Z
[a-zA-Z0-9] Tous les caractères alpha numérique
```

Il est enfin possible d'inverser la sélection avec le chapeau [^].

```
[^a-z] Tous les caractères QUI NE SONT PAS de a à z
```

Attention, le chapeau entre crochet n'a pas du tout la même signification qu'à l'extérieur des crochets.

Chapeau et dollar

Imaginez que vous voulez faire un détecteur de politesse dans un e-mail. Vous voulez tester si les messages commencent bien par *bonjour* et se terminent par *merci*. Pour réaliser cette prouesse technique, vous avez le symbole [^] signifiant '*rien avant*'. Et le symbole [$] signifiant '*rien après*'.

```
Bonjour
```

> **Bonjour** professeur, connaissez vous l'étymologie du mot "**Bonjour**" ? Merci

```
^Bonjour
```

> **Bonjour** professeur, connaissez vous l'étymologie du mot "Bonjour" ? Merci

```
Merci$
```

> Bonjour professeur, connaissez vous l'étymologie du mot "Bonjour" ? **Merci**

Les quantificateurs

Les quantificateurs sont les symboles: [* + ?]. Un quantificateur applique une règle au caractère qui le précède. (J'ai mis du temps à comprendre...)

Le point d'interrogation

Le point d'interrogation signifie : *le caractères est présent ou non*. Par exemple, si je veux chercher toutes les occurrences du mot *ARN* ou *ARNm*

```
ARNm?
```

On dit **ARN** ou **ARNm** ?

L' étoile

L'étoile signifie : *le caractère peut être absent ou répété une infinité de fois*. Par exemple si je veux récupérer toutes les occurrences du mot *Broom* à *Brooooooom*!

```
BRo*M
```

Démarrage du faucon millénium : **BRooM**!
Démarrage du faucon millénium : **BRoooooM**!
Démarrage du faucon millénium : **BRM**!.. WTF!?

le plus

Dans l'exemple précédent, le pattern détecte aussi le mot "BRM". Le caractère "plus" et comme l'étoile, mais signifie : *le caractère doit être présent une fois ou répété une infinité de fois*.

```
BRo+M
```

Démarrage du faucon millénium : **BRooM**!
Démarrage du faucon millénium : **BRoooooM**!
Démarrage du faucon millénium : BRM!.. haha!

Maintenant, que se passe-t-il si le caractère qui précède est un point, comme vu plus haut.

```
BR.+M
```

Démarrage du faucon millénium : **BRiiiiiM**!
Démarrage du faucon millénium : **BRaaaaaaM**!
Démarrage du faucon millénium : **BRaaaiiiiyaaaaamaaaM**!

Oui, c'est magique! Cette expression régulière signifie : '*répète n'importe quel caractères une ou plusieurs fois*'. Et vous pouvez l'appliquer au point, mais aussi à une classe de caractère. Dans l'exemple suivant, on répète une lettre majuscule:

```
[A-Z]+
```

Et si je veux détecter la répétition d'un mot ou d'un groupe de caractère ? Il suffit d'utiliser les parenthèses. Par exemple :

```
(chat)+
```

chatchatchatchatchat

Et pour finir, on peut spécifier le nombre de répétition à l'aide des accolades *{}*

```
(chat){3}    # 3 exactement
```

chatchatchatchatchat

```
(chat){3,5}  # 3 à 5 fois
(chat)(3,)   # Au minimum 3
```

Les classes abrégées

Pour finir, afin d'éviter de se fouler les doigts à écrire de longs patterns, vous pouvez utiliser ces raccourcis :

```
\d      # "digit" signifie [0-9]
\D      # "Not digit" signifie [^0-9]
\w      # "word" signifie [a-zA-Z0-9_]
\W      # "Not world" signifie [^a-zA-Z0-9_]
\t      # Tabulation
\n      # Saut de ligne
\r      # Retour chariot
\s      # Espace blanc
\S      # N'est pas un espace blanc
```

Où utiliser les expressions régulières ?

Partout!! Les expressions régulières vous vous permettre de faire des extractions de texte, des remplacements, des tests de validité sur des emails ou des IPs, des filtres pour vos logs systèmes et bien d'autre utilisation! L'autre jour, j'ai failli faire un malaise en voyant une collègue remplacer lignes après lignes, dans World, des numéros de titre...! En 2 secondes, c'était bouclé depuis sublime text!
Mais les expressions régulières, c'est surtout l'apanage des ninja sous linux. Avec la commande **grep** et **sed** et surtout le langage **Perl** vous allez pouvoir épater la galerie! Il ne vous reste plus qu'à vous entrainer! Souvenez vous, dès que vous faites une tâche répétitive sur du texte, il s'agit sûrement d'un boulot pour un regexp.

Référence

openclassrooms.com
wikipedia
http://www.expreg.com/

Ce site est versionné sur GitHub. Vous pouvez corriger des erreurs en vous rendant à cette adresse

ALSO ON **DRIDK.ME**

Un hook git pour mon blog

il y a 6 ans • 1 commentaire

Sacha Schutz, bioinformatique génétique médecine

Changer l'humanité avec le « gene drive »

il y a 5 ans • 7 commentaires

Sacha Schutz, bioinformatique génétique médecine

Euler et l'assemblage des génomes

il y a 5 ans • 2 commentaires

Sacha Schutz, bioinformatique génétique médecine

La s de l

il y a

Sacl bioi méc