Les expressions régulières

Simon Gabay, Adrien Jeanrenaud

Faire une recherche

On est tous habitués à faire des recherches dans un texte, mais nos recherches sont souvent limitées. Par exemple,

- comment trouver d'un coup les deux premières occurrences sans la dernière: `arrivons, arrivez, arrivait` ? Nous aurions besoins d'une alternative `arriv+ons ET arriv+ez` .
- comment trouver tous les chiffres?

Les expressions régulières permettent ce type de recherche poussées, mais aussi le remplacement.

Classes de caractères

Les classes sont une manière de contourner des problèmes complexes en formant des ensembles de signes. Nous en utilisons tous les jours:

- La ponctuation
- Les lettres
- Les chiffres
- Les majuscules

Classes de caractères: exemples

Avec les expressions régulières, on peut construire ces ensembles selon ses besoins:

- `a` pour la lettre minuscule *a*
- `A` pour la lettre majuscule *A*
- `[0213465987]` pour les chiffres.
- `[abcdefghijklmnopqrstuvwxyz]` pour les lettres minuscules
- `[ahr8t]` pour une sélection selon des besoins spécifiques

Certaines classes étant très communes, il existe des versions abrégées:

- `[0-9]` pour les chiffres
- `[a-z]` pour les lettres minuscules

Classes de caractères: raccourcis

Certaines classes ne nécessitent pas l'emploi de crochets:

- `[0213465987]` = `[0-9]` = `\d`
- `\D` tout ce qui n'est pas dans `\d`
- `\w` tout ce que l'on peut trouver dans un mot (= `[a-zA-Z0-9_]`)
- `\W` tout ce qui n'est pas dans `\w`
- `\s` tout ce qui représente un espace: espace, tabulation, retour à la ligne, retour chariot, tabulation verticale...
- `\S` tout ce qui n'est pas dans `\s`

Echappement

Le signe `\` (antislash) est aussi utilisé pour échapper les caractères utilisés dans la syntaxe des regex. Ainsi le signe `+` permet d'indiquer qu'un caractère ou une classe sont répétés un nombre infini de fois (`a+` = `aaaaaa...`). `\+` permet de chercher le signe `+`

Exercice 1

Allez sur le site https://regex101.com, qui permet de tester des expressions régulières.

Tentez de capturer tous les mentions de siècles, peu importe le format dans le texte suivant:

```
Victor Hugo est né au XIXeme siècle, Jean Racine au
XVIIeme et Rabelais au 16eme.
```

Motifs

Un motif (en anglais *pattern*) est une recherche de caractères assortis de metacaractères ou de délimiteurs qui permettent de préciser la recherche. Prenons un exemple (retournez sur https://regex101.com):

```
En 1666, il a passé le pas de 66 portes, n'est-ce pas?
```

Si nous recherchons `pas` que se passe-t-il? et `[pas]` ?

Exercice 2

- Comment peut-on récupérer uniquement le substantif *pas* et pas l'adverbe avec une des classes vues précédemment?
- Comment peut-on récupérer l'adverbe de négation final avant le point d'interrogation? (attention il y a une astuce)

Quantificateurs

Reprenons le même exemple:

```
En 1666, il a passé le pas de 66 portes, n'est-ce pas?
```

Prenons le chiffre *6* (*1**666***, ***66***).

- Chercher `6` va nous donner autant de réponse qu'il y a d'occurrences
- Chercher `6+` va nous permettre de trouver les cas où *6* est répété x fois
- Comment récupérer l'occurrence ou *6* n'est répété que deux fois et pas trois?

Quantificateurs

Pour trouver une séquence précise, nous pouvons utiliser un quantifieur qui précise entre accolades le nombre de répétition voulues:

```
[6]{2}
```

Ainsi, une année sera trouvée avec la combinaison `[0-9][0-9][0-9][0-9]` ou `[0-9]{4}`.

Le quantificateur peut être encore affiné avec des bornes: `[0-9]{2,4}` permet de trouver un chiffre répété entre 2 et 4 fois.

Quantificateurs spéciaux

Quelques cas spécifiques sont les suivants:

- `*` (*joker* ou *wildcard*) permet de récupérer un caractère répété zéro fois ou plus (`[A-Z]{4}[0-9]*` récupère quatre lettres majuscules suivies ou non d'un chiffre de x caractères).
- `?` permet de récupérer un caractère absent ou présent (`pas?` récupère le début de *pa**s**tate* et ***pas*** et ***pas**ser*).
- `+` permet de récupérer un caractère répété une fois ou plus (`pas+` récupère ***pas*** et ***pass**er*, mais pas *patate*).

Les ancres

Les ancres permettent de préciser la position des caractères. Ainsi dans l'exemple d'une exclamation de rire:

```
ah ah ah
```

- `^` sélectionne les mots en début de ligne (`^ah` sélectionne le premier *ah*)
- `$` est le symétrique de `^` et sélectionne la fin de ligne (`ah$` sélectionne le dernier *ah*)
- `\b` permet de sélectionner les début et fin de mots (`\bah\b` sélectionne tous les *ah*)

Exercice 3

```
1. Nos avem bailat au cosols XXV s que ac
Bertolmeus Patriarcha que sai devia per Jo. Forneir.
2. Item XX s. e VI d per S. Tondedor.
3. Item XX s que ac D. l'uchaires.
4. Item II s que esmendesmes a D. de Ranos per son
sobrecot que afolet al cosolat.
5. Item XX d als garsos que portont los gatjes.
6.. Item XL s que agront lhi cosol W. de Bualet sos cors
cant donem la letra de Riom a S. Bonome.
7. Item IX £ per los borzes de Paris.
8 Item XXV £ que ac P. Albanels de part P. Bonome a.lo
jorn de la Saint Peire de feureir 22 fevrier 1274 n.s.
```

Dans ce registre consulaire de Montferrand [source], sélectionnez les montants dépensés (soit les chiffres romains suivis de *s*, *s.*, *d*, *£*).

Groupes de motifs

Il est possible de former des groupes à l'aide de parenthèses, qui seront notamment utiles s'il faut les manipuler (par exemple dans le cas d'un rechercher-remplacer).

```
En l'an 1902 Hugo est né, et en l'an 1885 il est mort.
```

Si l'on recherche les années, il existe plusieurs solutions qui regroupent différement l'information:

- Solution 1: `an\s([0-9]{4})`
- Solution 2: `(an\s[0-9]{4})`
- Solution 3: `(an)\s([0-9]{4})`

Réutilisation

Il est possible de réutiliser un groupe déjà défini en utilisant une variable. La valeur de la variable est entre parenthèses, et est appelée par `\` suivi de sa position. Ainsi:

```
Le cantique des cantiques
```

Je peux capturer l'expression *cantique des cantiques* par la regex:

```
Le\s(cantique)\sdes\s\1s
```

Substitution

Lorsque nous souhaitons recopier le groupe dans le cadre d'une substitution, nous utilisons encore une variable entre parenthèses, mais qui est appelée par `$` suivi de la position du groupe. Ainsi `(an)\s([0-9]{4})` (pour *an 1902* par exemple):

- Remplacé par `$1\s$2`, il renverra le même résultat (*an 1902*).
- Remplacé par `année\s$2`, remplacera le premier groupe par la nouvelle chaîne de caractères, mais laissera le deuxième groupe intact (*an 1902* -> *année 1902*).

Exercice 4

Nos reprennons le texte de l'exercice 3:

```
1. Nos avem bailat au cosols XXV s que ac
Bertolmeus Patriarcha que sai devia per Jo. Forneir.
2. Item XX s. e VI d per S. Tondedor.
3. Item XX s que ac D. l'uchaires.
4. Item II s que esmendesmes a D. de Ranos per son
sobrecot que afolet al cosolat.
5. Item XX d als garsos que portont los gatjes.
6.. Item XL s que agront lhi cosol W. de Bualet sos cors
cant donem la letra de Riom a S. Bonome.
7. Item IX £ per los borzes de Paris.
8 Item XXV £ que ac P. Albanels de part P. Bonome a.lo
jorn de la Saint Peire de feureir 22 fevrier 1274 n.s.
```

- Remettre les points d'abréviation pour les devises (*s*, *d*) quand ils manquent.
- Remplacer ces abréviations par des formes complètes (*sous*, *deniers*, *livres*).
- Encoder les montants avec des balises XML `<price>$</price>`)

Négation

Il est possible de chercher l'inverse d'un caractère ou d'une classe en utilisant le signe `^` entre crochets:

- `[^I]` est l'inverse de `[I]` (ou `I`): tout ce qui n'est pas un *I* majuscule
- Il ne faut pas confondre `[^a-z]` (tout sauf une minuscule) et `(^a-z)` (qui ne commence pas par une minuscule).

pellent volontiers quoniam, en latin, qui signifie parce que, mais désigne souvent les parties nobles dans les récits populaires du moyen âge. On voit que les Saxons et les Turcs manifestent à l'égard de Croniamantal le même sentiment en lui appliquant des surnoms identiques, mais dont l'origine est encore mal expliquée. On suppose que c'est une allusion euphémique à ce qui se trouvait dans le rapport médical du médecin marseillais Ratiboul sur la mort de Croniamantal. D'après cette pièce officielle, tous les organes de Croniamantal étaient sains et le médecin légiste ajoutait en latin, comme fit l'aide-major Henry pour Napoléon : partes viriles exiguitatis insignis, sicut pueri.

Au demeurant, il est des pays où la notion de la virilité croniamantalesque a complètement disparu. C'est ainsi qu'en Moriane les nègres le nomment Tsatsa ou Dzadza ou Rsoussour, noms féminins, car ils ont féminisé Croniamantal comme les Byzantins ont féminisé le vendredi saint en en faisant sainte Parascève.

II

Procréation

A deux lieues de Spa, sur la route bordée d'arbres tordus et de buissons, Viersélin Tigoboth, musicien ambulant qui arrivait à pied de Liége, battait le briquet pour allumer sa pipe. Une voix de femme cria :

Alternatives

Il est possible de laisser ouverte une alternative entre deux solutions avec le signe `|` (*pipe*). Ainsi `a | b` signifie *a ou b*.

Lazy et *greedy*

Pour les regexs, on parle de mode:

- *greedy* (glouton) essaie de correspondre à un élément autant de fois que possible.
- *lazy* (paresseux) essaie de correspondre à un élément aussi peu de fois que possible.

Lazy et *greedy*: activation

Le mode de recherche par défaut est le mode glouton: pour activer le mode paresseux il faut ajouter un point d'interrogation après le quantificateur:

- `*` -> `*?` pour zéro occurrence ou plus.
- `+` -> `+?` pour une ou plusieurs occurrences.
- `?` -> `??` pour zéro ou une occurrence.
- ...

Observez la différence entre `.*?s` et `.*s` pour l'exemple suivant:

```
Les pas des portes.
```

Groupe non-capturant

Il est possible de définir des groupes qui ne seront pas capturés (on ne pourra pas les récupérer avec une variable de type `$1` ou `\1`) en utilisant `?:` au début du groupe.

Admettons que je veuille récupérer tous les prénoms de roi (`\w+`) sans le numéro qui le suit (`[A-Z]+`):

```
Louis VIII
Charles IX
Guillaume X
Louis XI
```

Je peux chercher l'expression `(\w+)\s(?:[A-Z]+)`.

Lookaround

Il existe des groupes non-capturants spécifiques: les *lookaround* qui se positionnent dans le flux du texte. On parle de:

- *lookahead* pour regarder devant
- *lookbehind* pour regarder derrière

Lookahead

Le *positive lookahead* suit une syntaxe proche du groupe non-capturant et commence par `?=`. On pourra ainsi sélectionner les dates av. JC et après JC:

```
150 av. JC
300 ap. JC
```

Pour cela on utilise l'expression régulière `[0-9]+(?=\sav\.\sJC)`.

Il est possible de faire un *negative lookahead* avec la syntaxe `?!` : `[0-9]+(?!\sav\.\sJC)` sélectionera l'inverse (en l'occurrence les dates après JC).

Lookbehind

Le même principe existe pour ce qui se trouve avant le groupe qui nous intéresse:

- *Positive lookbehind* avec `?<=`
- *Negative lookbehind* avec `?<!`

```
juillet 1900
janvier 1900
```

La syntaxe reste la même: `(?<=juillet\s)[0-9]{4}` pour sélectionner la date de juillet, et `(?<!juillet\s)[0-9]{4}` pour tout sauf cette date.

Solutions

1. `[A-Z|\d+]+eme`
2. `le\spas` et `pas\?`
3. `[A-Z]+\ss?\.?£?d?`
4.
   - `([A-Z]+)\ss\s` et `$1 s.`
   - `([A-Z]+)\sd\s` et `$1 d.`
   - `([A-Z]+)\ss\.` et `$1 sous`
   - `([A-Z]+)\sd\.` et `$1 deniers`
   - `([A-Z]+)\s£\.` et `$1 livres`
   - `([A-Z]+)\s(£|d|s)` et `<price>$1 $2</price>`
5.
   - `\n` remplacé par rien
   - `\.\n` remplacé par `\.\n\n`
   - `\n([A-Z]+)\n` remplacé par `\n\n§$1\n`