# Regex Programming Porn

**Les regex comme vous ne avez jamais vus**

Human Talks Lyon, 08/10/13 / **@gautier_difolco**

*regular expressions is #programming #porn, you always feel guilty and dirty after that :'(*

Deux solutions :

- Plus de pr0n
- Plus de Regex

# Notes

- Présentation des regex PCRE (et non POSIX)
- Nous parlerons de regex pour éviter le débat expressions régulières vs expressions rationnelles
- Présentation et environnement de benchmarks disponibles sur **https://github.com/blackheaven/regex_programming_porn**

# Plan

- Constitutifs de base
- Changements du comportement d'évalutation
- Un exemple qui fait saigner du nez

# Constitutifs de base

# /PATTERN/OPTIONS

/at/

*A c**at** e**at**s a black f**at** r**at***

# Alternative monocaractère

/[cft]/

*A **c**a**t** ea**t**s a bla**c**k **f**a**t** ra**t***

# Exclusion monocaractère

`/[^a]/`

***A cat eats** a **black fat rat***

`/[^rt]/`

***A ca**t **eats a black fa**t r**a**t*

# Multiplicité

0 ou 1 `/at?/` - `/at{0,1}/` - `/at{,1}/`

*A c**at** e**at**s **a** bl**a**ck f**at** r**at***

0 à inf. `/at*/` - `/at{0,}/`

*A c**at** e**at**s **a** bl**a**ck f**at** r**at***

1 à inf. `/at+/` - `/at{1,}/`

*A c**at** e**at**s **a** bl**a**ck f**at** r**at***

n à m `/at{n,m}/`

*A c**at** e**at**s **a** bl**a**ck f**at** r**at***

n `/at{n}/`

*A c**at** e**at**s **a** bl**a**ck f**at** r**at***

# Classes de caractères

Intervals `/[a-z]/`

*H**ier, j'ai termin**é à 6**h**00.*

Classes prédefinies `/[\w]/` vs `/[[:alnum]]/`

***Hier, j'ai terminé à 6h00**.*

Jocker `/.\W/`

*Hie**r, j'**a**i**_termin**é**_**à**_6h00.*

**Non-équivalences à cause des accents dans certains langages**

Accents `/[=e=]/`

*Hi**e**r, j'ai t**e**rmin**é** à 6h00.*

# Capture / Alternatives multi-caractères

`/(ca|ts)/`

*A **ca**t ea**ts** a black fat rat*

`/(.)(?:la|at)/`

*A **c**at **e**ats a **b**lack **f**at **r**at*

En PCRE `/(bla|blac)/`

*A cat eats a **bla**ck fat rat*

En POSIX `/(bla|blac)/`

*A cat eats a **blac**k fat rat*

# Changements du comportement d'évalutation

# Ancres

Début `/^./`

***A** cat eats a black fat rat*

Fin `/.$/`

*A cat eats a black fat ra**t***

Délimiteur de mots `/.at\b/`

*A c**at** eats a black f**at** r**at***

Qui est différent de l'espace `/.at\s/`

*A c**at**_eats a black f**at**_r**at$***

# Options

## /PATTERN/OPTIONS

Insensibilité à la casse `/a/i`

***A** c**a**t e**a**ts **a** bl**a**ck f**a**t r**a**t*

Prise en compte des retours à la ligne des ancres `/at$/m`

*A ca**t\n***

*eats a black fat ra**t$***

Prise en compte des retours pour le point `/at./s` voire `/at./s`

*A ca**t\n***

*e**ats** a black fa**t**_rat*

# Commentaires

/at# Cherche at/x

*A c**at** e**at**s **a** bl**a**ck f**at** r**at***

# Comportements glouton

Chercher la plus longue correspondance `/a.*t/`

*A c**at eats a black fat rat***

Chercher la plus courte correspondance `/a.*?t/`

*A c**at** e**at**s **a black fat** r**at***

# Références aux correspondances

Les correspondances capturées sont accessible via \1 à \9

Chercher les balise à contenu `/<(\w+)>.*<\/\1>/`

```
<p></br></p>
```

# Motifs prospectifs

Positif `/(?=.*[A-Z])(?=.*\d).{5,}/`

*MyPassword*

Négatif `/(?=.*[A-Z])(?!.*\d).{5,}/`

***MyPassword***

# Motifs retrospectifs

Positif `/(?<=if) (true|false)/`

*if **true** while false*

Négatif `/(?<!if) (true|false)/`

*if true while **false***

# Motifs conditionnels

```
/if (true)?(?(1)(?! else)|false else true)/
```

*if true then false*

*if **true***

***if false then true***

```
/if (true)?(?(1)(?! else))/
```

*if true then false*

*if **true***

***if** false*

# Et plus encore

- Motifs retardés (récursifs) en Perl
- Motifs autonomes ("super-glouton") en JAVA

# Un exemple

```
/\b(?:[\w_][\w_\d]*)\b(?
<!if|else|while|for|return)\s*\b([\w_]
[\w_\d]*)\b(?!\s*\()(?!.*\b\1\b)/
```

```
/
\b
    (?:[\w_][\w_\d]*) # Groupe 1
\b
(?<!if|else|while|for|return) # Rétrospectif négatif 1
\s*
\b
    ([\w_][\w_\d]*) # Groupe 2
\b
(?!\s*\() # Prospectif négatif 1
(?!.*\b\1\b) # Prospectif négatif 2
/xm
```

```
/
\b
    (?:[\w_][\w_\d]*) # Identificateur de type non capturé
\b
(?<!if|else|while|for|return) # qui n'est pas un mot clef
\s*
\b
    ([\w_][\w_\d]*) # Identificateur capturé
\b
(?!\s*\() # Prospectif négatif 1
(?!.*\b\1\b) # Prospectif négatif 2
/xm
```

```
/
\b
    (?:[\w_][\w_\d]*) # Identificateur de type non capturé
\b
(?<!if|else|while|for|return) # qui n'est pas un mot clef
\s*
\b
    ([\w_][\w_\d]*) # Identificateur capturé
\b
(?!\s*\() # qui n'est pas le nom d'une fonction
(?!.*\b\1\b) # Prospectif négatif 2
/xm
```

```
/
\b
    (?:[\w_][\w_\d]*) # Identificateur de type non capturé
\b
(?<!if|else|while|for|return) # qui n'est pas un mot clef
\s*
\b
    ([\w_][\w_\d]*) # Identificateur capturé
\b
(?!\s*\() # qui n'est pas le nom d'une fonction
(?!.*\b\1\b) # et qui ne réapparaît plus
/xm
```

```
/
\b
    (?:[\w_][\w_\d]*) # Identificateur de type non capturé
\b
(?<!if|else|while|for|return) # qui n'est pas un mot clef
\s*
\b
    ([\w_][\w_\d]*) # Identificateur capturé
\b
(?!\s*\() # qui n'est pas le nom d'une fonction
(?!.*\b\1\b) # et qui ne réapparaît plus
/xm
```

## Liste les variables déclarées mais non utilisées

# Conclusion

- Utile pour extraire/valide des informations basiques (addressses email, dates, unités lexicalles)
- Nombreuses variations en fonction du langage ou de la bibliothèque
- Un "vrai" analyseur lexical/syntaxique devient rapidement vital

# Questions ?

Pour continuer l'entrainement : **http://regexcrossword.com/**