Biểu thức chính quy - Regular expression

Biểu thức chính quy là gì?

Biểu thức chính quy là một chuỗi các ký tự, ký hiệu thể hiện một khuôn mẫu xâu ký tự nhất định. Biểu thức chính quy được sử dụng trong việc kiểm tra định dạng xâu kí tự và tìm kiếm văn bản (text).

Biểu thức chính quy tên tiếng anh là Regular Expression gọi tắt là regex hoặc regexp

Trong lập trình nó được dùng với các hàm xử lý xâu kí tự, xử lý văn bản với các tác vụ cụ thể như: tìm và thay thế chuỗi, kiểm tra tính hợp lệ của dữ liệu, trích xuất chuỗi con từ một chuỗi ... Hầu hết các ngôn ngữ lập trình đều hỗ trợ `Regex` như C++, JAVA, PHP ... Thậm chí RegExp còn rất phổ biến trong các ứng dụng, công cụ khác nhau như rewrite URL mod_rewrite, tìm kiếm và thay thế trong các IDE

Thử xem xét một ví dụ về Regex - Giả sử bạn ứng dụng của bạn yêu câu người dùng phải tuân thủ quy tắc đặt tên: *(1)Tên được phép chứa các ký tự, các số, gạch dưới, gạch nối. (2) Tên phải có độ dài trong khoảng cho phép từ 3 đến 15 ký tự.*. Thì biểu thức chính quy biểu diễn quy tắc đó sẽ như sau:

```
^[a-z0-9_-]{3,15}$
```

- `^` Ký hiệu cho biết bắt đầu một dòng
- `[a-z0-9_-]` Cho phép tên chứa ký tự a-z, số từ 0 - 9, ký tự -, ký tự _
- `{3,15}` Tên dài 3 đến 15 ký tự
- `$` Điểm kết thúc dòng

Với biểu thức chính quy trên thì các tên như `xuanthu`, `xuan-xhu`, `xuanxhu_123` ... được chấp nhận vì dài trong khoảng 3 - 15, chữ đều viết thường.

Biểu thức RegExp cơ bản

Một biểu thức Regex chỉ là một mẫu các ký tự dùng để tìm kiếm trong text (chuỗi). Ví thụ một biểu thức `ước` có nghĩa phù hợp với nó là bắt đầu bằng `ư` theo sau là `ớ` và tiếp theo là `c`, mang biểu thức so với text thì thấy hợp mẫu như sau:

```
ước => Ước một điều ... mộng ước rất đơn sơ.
```

Biểu thức chính quy Regex là phân biệt chữ hoa chữ thường. Có cơ chế bật cờ thiết lập không phân biệt chữ hoa, chữ thường ở phần dưới

Các ký tự biểu diễn - Meta

Các ký hiệu biểu diễn thông tin tóm tắt lại và giải thích chi tiết từng mục ở phần sau

Ký tự Meta	Mô tả
.	Biểu diễn bất kỳ ký tự nào ngoài trừ ký tự xuống dòng
[ ]	Tập hợp ký tự. Phù hợp nếu có bất kỳ ký tự nào trong dấu []
[^]	Tập hợp ký tự phủ định. Phù hợp nếu không có ký tự nào trong []
*	Lặp lại 0 đến nhiều lần.
+	Lặp lại 1 hoặc nhiều lần
?	Tùy chọn có hay không cho mẫu phía trước
{n,m}	Độ dài tối thiểu là n tối đa là m
(xyz)	Biểu diễn một nhóm.
|	Biểu diễn thay thế, phép toán `or`
\	Biểu diễn ký tự đặc biệt `[ ] ( ) { } . * + ? ^ $ \ |`
^	Điểm bắt đầu của dòng.
$	Điểm kết thúc của dòng

Ký hiệu chấm .

Ký hiệu dấu chấm `.` là một meta đơn giản, nó biểu diễn bất kỳ ký tự nào ngoài trừ ký tự `return \r` hoặc `newline \n`. Ví dụ biểu thức `.oàn` thì có nghĩa là: một ký tự nào đó, tiếp theo đến ký tự `o`, tiếp theo đến `à` cuối cùng là `n`. Ví dụ dùng mẫu đó tìm trong chuỗi.

```
.oàn =>Sự hoàn hảo dường như không thể đạt được, nhưng nếu chúng ta theo đuổi sự hoàn hảo thì chúng ta sẽ chạm đến sự xuất sắc.
```

Tập hợp ký tự [ ]

Dùng `[]` để chứa tập hợp các ký tự. Có thể dùng dấu `-` để biểu diễn một dải các ký tự theo vị trí trong bảng chữ cái như `a-z`, `0-9` ..., biểu thức so sánh sẽ hợp mẫu nếu chứa bất kỳ ký tự nào trong đó (không cần quan tâm thứ tự)

Ví dụ biểu thức `[ưư]ớc` có nghĩa là: Có một chữ `ư` hoặc `ư`, theo sau bởi `ớ`, tiếp theo là `c`

```
[ưư]ớc => Ước một điều ... mộng ước rất đơn sơ.
```

Nếu `[]` chứa `.` thì nó biểu diễn ký tự `.` chứ không con ý nghĩa đại diện như trường hợp trên.

```
nh[.] => Thời gian cứ thế xoay vòng thật nhanh. Bao mùa chiếc áo phông phanh!
```

Tập hợp ngoại trừ [^]

Thông thường thì `^` biểu diễn điểm bắt đầu của chuỗi, tuy nhiên nếu nó nằm ở vị trí sau dấu `[` của cặp `[]` thì nó lại mang ý nghĩa tạo ra tập hợp ký tự loại trừ (phụ định). Ví dụ biểu thức `[^n]hanh` có nghĩa là bất kỳ ký tự nào ngoại trừ ký tự `n`, theo sau bởi `h`, tiếp theo bởi `a`, `n` và `h`

```
[^n]hanh => Thời gian cứ thế xoay vòng thật nhanh. Bao mùa chiếc áo phông phanh!
```

Lặp lại với ký tự *

Ký hiệu `*` cho biết có sự lặp lại 0 hoặc nhiều lần mẫu phù hợp đứng phía trước nó. Ví dụ mẫu `a*` có nghĩa là ký tự `a` lặp lại 0 hoặc nhiều lần là phù hợp. Nếu nó đi sau tập hợp thì lặp tập hợp đó lặp lại 0 hoặc nhiều lần. ví dụ `[a-z]*` có nghĩa là dòng có số lượng bất kỳ các ký tự chữ viết thường thì phù hợp.

`*` có thể sử dụng với `.` để biểu diễn bất kỳ chuỗi nào, hay dùng mẫu `(.*)`
`*` có thể sử dụng với ký tự trắng `\s` để biểu diễn bất kỳ khoảng trắng nào.
Ví dụ `\s*mình\s*` có nghĩa bắt đầu bởi không hoặc nhiều khoảng trắng, tiếp theo là ký tự `m`, `ì`, `n`, `h` tiếp theo là không hoặc nhiều khoảng trắng.

```
"\s*mình\s*" => Đừng so sánh mình với bất cứ ai trong thế giới này.
Nếu bạn làm như vậy có nghĩa bạn đang sỉ nhục chính bản thân mình. Bill Gates
```

Lặp lại với ký tự +

Ký hiệu `+` tương tự như `*` nhưng lặp lại 1 hoặc nhiều. Ví dụ
`có.+!` có nghĩa ký tự bắt đầu bằng `có` theo sau ít nhất một ký tự nào đó, tiếp theo là ký tự `!`.

```
có.+! => Nếu bạn có cố gắng mới thành công!: có cố gắng mới thành công!.
```

Mẫu phía trước có hay không đều được với ?

Trong biểu thức Regex thông thường `?` là một tùy chọn cho biết mẫu phía trước nó có thể có hoặc không. Ví dụ `[h]?ôn` nghĩa là tùy chọn có `h` hoặc không, theo sau là `ô`, tiếp theo là `n`

```
[h]?ôn => Người đàn ông khôn ngoan là người biết ít hơn, nhưng hiểu nhiều hơn.
```

Biểu diễn độ dài {}

`{}` là biểu diễn số lượng, nó chỉ ra số lần mà một ký tự hoặc một nhóm các ký tự lặp lại. Ví dụ `[0-9]{2,3}` có nghĩa là có tối thiểu 2 tới 3 ký tự số.

Bạn có thể bỏ đi số thứ 2, ví dụ [0-9]{2,} có nghĩa là chuỗi có 2 hoặc nhiều ký tự số. Nếu bỏ đi ký tự `,` ví dụ `[0-9]{3}` có nghĩa là chuỗi chính xác có 3 ký tự.

Nhóm mẫu (...)

Nhóm ký tự là một mẫu (`pattern`) con được viết biên trong `()`. Ví dụ `(ab)*` lặp lại **ab** 0 hoặc nhiều lần. Chúng ta cũng dùng ký hiệu `|` bên trong nhóm như là phép toán `or` để xác định nhóm. Ví dụ `n(g|h)` có nghĩa bắt đầu bằng `n` theo sau là một mẫu, mẫu đó hoặc là chữ `g` hoặc là chữ `h`

```
n(g|h) =>Nếu có một ai đó làm chậm bước chân của bạn, hãy nhẹ nhàng rẽ sang hướng khác.
Chạy thử vidu09
```

Biểu diễn thay thế |

Xem ví dụ trên

Biểu diễn ký tự đặc biệt với \

Do một số ký hiệu đã được dùng đã biểu diễn Regex như : `{ } [ ] / \ + * . $ ^ | ?` nên để biểu diễn các ký tự đó dùng ký hiệu `\` trước ký tự.

`"(f|c|m)at\.?"` => The fat cat sat on the mat.

Bắt đầu của dòng ^

Sử dụng `^` để cho biết sẽ kiểm tra sự phù hợp nếu ký tự đầu tiên của chuỗi hợp mẫu. Ví dụ `^a` thì chuỗi phù hợp có dạng như `abcxyz`, nếu vẫn chuỗi đó nó lại không phù hợp với `^b`.
`^(T|t)he` có nghĩa là `T` hoặc `t` bắt đầu của chuỗi, theo sau là `he`

Điểm kết thúc của chuỗi $

Cho biết kết thúc dòng phải thỏa mãn mẫu phía trước $

Ngược lại với `^` ví dụ `(at\.)$` nghĩa là cuối chuỗi có `at.` thì là phù hợp.
`"(at\.)$"` => The fat cat. sat. on the mat.

Ký hiệu tắt cho tập hợp

Viết tắt	Diễn tả
.	Bất kỳ ký tự nào ngoại trừ xuống dòng
\w	Chữ,số, và _, tương đương với: `[a-zA-Z0-9_]`
\W	Ngoài bảng chữ cái, tương đương với: `[^\w]`
\d	Các số: `[0-9]`
\D	Không phải số: `[^\d]`
\s	Là ký tự trắng, tương đương với: `[\t\n\f\r\p]`
\S	Không phải ký tự trắng: `[^\s]`
\b	Ranh giới từ

Tham khảo thêm cú pháp RegEx của thư viện chuẩn C++ regex

https://www.cplusplus.com/reference/regex/ECMAScript/