Regular Expression

Võ Lê Ngọc Thịnh

Ngày 25 tháng 3 năm 2025

Mục lục

1 Regular Expression là gì?

Regular Expression hay còn gọi là biểu thức chính quy được dùng để xử lý chuỗi nâng cao thông qua biểu thức riêng của nó, những biểu thức này sẽ có những nguyên tắc riêng và ta phải tuân theo nguyên tắc đó thì biểu thức mới hoạt động được. Ngoài cái tên gọi Regular Expression ra thì nó còn có thể viết tắt thành RegEx.

Nguyên tắc hoạt động của biểu thức RegEx là so khớp dựa vào khuôn mẫu (pattern), khuôn mẫu được xây dựng từ các quy tắc căn bản của biểu thức RegEx. RegEx thường được dùng trong lập trình, xử lý văn bản, tìm kiếm, và các ứng dụng liên quan.

Đặc điểm của RegEx:

1. **Cú pháp độc lập ngữ cảnh:**
    - RegEx không phụ thuộc vào ngữ cảnh và có thể đại diện cho một loạt các tập hợp ký tự hoặc thứ tự sắp xếp ký tự.
    - Các tập hợp ký tự này được diễn giải theo các quy tắc của ngôn ngữ hiện tại hoặc môi trường thực thi.
2. **Các loại RegEx phổ biến:**
    - Basic Regular Expressions (BRE).
    - Extended Regular Expressions (ERE).

2 Mô tả mẫu với Regex

2.1 Định nghĩa

Mỗi RE đại diện cho một tập hợp các chuỗi (set of strings), được định nghĩa như sau:

The RE	**Representation (Biểu diễn)**
Empty RE	Tập hợp chuỗi rỗng, không có phần tử nào (empty set of strings, with 0 elements)
A character	Tập hợp chuỗi với một phần tử duy nhất, chính là ký tự đó (set of strings with one element, itself)
RE enclosed in parentheses	Tập hợp chuỗi giống với RE gốc, không bao gồm dấu ngoặc (the same set of strings as the RE without the parentheses)
RE consisting of two **concatenated** REs	Tích Descartes (cross product) của các tập hợp chuỗi được biểu diễn bởi các thành phần riêng lẻ
RE consisting of the **or** of two REs (|)	Hợp (union) của các tập hợp chuỗi được biểu diễn bởi từng thành phần riêng lẻ
RE consisting of the **closure** of an RE (*)	Tập hợp chuỗi bao gồm chuỗi rỗng (the empty string) hoặc hợp của các chuỗi từ kết quả nối lặp lại nhiều lần RE

2.2 Quy tắc cơ bản về Regex

1. Nối chuỗi (Concatenation) A B ⇒ Xác định tập hợp chuỗi $\{AB\}$, tức là tạo thành một chuỗi có hai ký tự bằng cách ghép A và B.

2. Hoặc (Or) Một toán tử | giữa hai lựa chọn nghĩa là cả hai đều thuộc tập hợp chuỗi.

- Ví dụ: $A|B$ xác định tập hợp $\{A, B\}$.
- **Ghép chuỗi có độ ưu tiên cao hơn toán tử "or"** ⇒ $AB|BCD$ xác định tập hợp $\{AB, BCD\}$.

3. Lặp lại (Closure) Cho phép một phần của mẫu được lặp lại tùy ý, bao gồm cả không xuất hiện (0 lần).

- A^* xác định $\{\in$ (chuỗi rỗng), $A, AA, AAA, ...\}$.
- A^*B xác định $\{B, AB, AAB, AAAB, ...\}$.
- AB^* xác định $\{A, AB, ABB, ABBB, ...\}$.

4. Dấu ngoặc đơn (Parentheses) Dùng dấu ngoặc đơn để ghi đè quy tắc độ ưu tiên mặc định.

- Ví dụ: $C(AC|B)D$ xác định tập $\{CACD, CBD\}$.

Biểu thức chính quy (RE)	**Khớp**	**Không khớp**
(A|B)(C|D)	AC, AD, BC, BD	Mọi chuỗi khác
A(B|C)*D	AD, ABD, ACD, ABCCBD	BCD, ADD, ABCBC
A*[(A*BA*BA*)]*	AAA, BBAABB, BABAAA	ABA, BBB, BABBAAA

2.3 Toán tử và Ký hiệu trong RegEx

1. Toán tử cơ bản

Tên hoặc ý nghĩa	**Ký hiệu (Notation)**	**Ví dụ**
Nối chuỗi (Concatenation)	Không có ký hiệu	`AB` (khớp với "AB")
Hoặc (Or)	`|`	`A|B` (khớp với `A` hoặc `B`)
Nhóm (Parentheses)	`()`	`(A|B)*C` (khớp với `AC`, `ABC`, `AAC`,...)

2. Toán tử lặp (Quantifiers)

Tên hoặc ý nghĩa	**Ký hiệu (Notation)**	**Ví dụ**
Lặp 0 hoặc nhiều lần	`*`	`A*` (khớp `"", A, AA, AAA,...`)
Lặp ít nhất 1 lần	`+`	`(AB)+` (khớp `AB, ABAB, ABABAB,...`)
0 hoặc 1 lần	`?`	`(AB)?` (khớp `""` hoặc `AB`)
Lặp số lần cụ thể	`{n}`	`(AB){3}` (khớp `ABABAB`)
Lặp trong khoảng cụ thể	`{n,m}`	`(AB){1,2}` (khớp `AB` hoặc `ABAB`)

3. Tập hợp ký tự (Character Sets)

Tên hoặc ý nghĩa	**Ký hiệu (Notation)**	**Ví dụ**
Bất kỳ ký tự nào	`.`	`A.B` (khớp với `AOB`, `A-B`, `ACB`,...)
Tập hợp cụ thể	`[...]`	`[AEIOU]*` (chỉ chứa nguyên âm)
Khoảng giá trị	`[a-z]`, `[0-9]`	`[A-Z]`, `[0-9]`
Phủ định tập hợp	`[^...]`	`[^AEIOU]*` (không chứa nguyên âm)

4. Ký hiệu đặc biệt (Escape Sequences)

Tên hoặc ý nghĩa	Ký hiệu (Notation)	Ví dụ
Ký tự số	`\d`	`\d{3}` (khớp với 3 chữ số)
Không phải số	`\D`	`\D+` (chuỗi không có số)
Ký tự chữ và số	`\w`	`\w+` (tương đương với `[a-zA-Z0-9_]`)
Không phải chữ và số	`\W`	`\W+` (không phải chữ cái/số)
Khoảng trắng	`\s`	`\s+` (chuỗi chứa khoảng trắng)
Không phải khoảng trắng	`\S`	`\S+` (không chứa khoảng trắng)

5. Toán tử Vị trí (Anchors)

Tên hoặc ý nghĩa	Ký hiệu (Notation)	Ví dụ
Bắt đầu chuỗi	`^`	`^Hello` (khớp với "Hello" ở đầu chuỗi)
Kết thúc chuỗi	`$`	`world!$` (khớp với "world!" ở cuối chuỗi)
Ranh giới từ	`\b`	`\bword\b` (khớp với từ "word" nguyên vẹn)
Không phải ranh giới từ	`\B`	`\Bword\B` (khớp với "word" bên trong từ khác)

3 Ứng dụng của Regex

3.1 Tìm kiếm và thao tác văn bản

Regex hỗ trợ tìm kiếm và chỉnh sửa văn bản một cách linh hoạt và hiệu quả. Các ứng dụng bao gồm:

- Tìm kiếm mẫu văn bản cụ thể trong tài liệu hoặc cơ sở dữ liệu.
- Thay thế các chuỗi tìm thấy bằng các giá trị mới, đặc biệt hữu ích trong tái cấu trúc mã nguồn hoặc làm sạch dữ liệu.

Ví dụ: Regex `\b\d{3}\b` có thể dùng để tìm tất cả các số gồm 3 chữ số trong một văn bản.

3.2 Kiểm tra và xác thực đầu vào

Regex đóng vai trò quan trọng trong việc xác thực dữ liệu đầu vào, đảm bảo rằng thông tin người dùng cung cấp tuân thủ các định dạng cụ thể. Ứng dụng phổ biến:

- Kiểm tra định dạng email, số điện thoại, ngày tháng trong ứng dụng web.

- Nâng cao tính bảo mật và độ tin cậy của hệ thống.

Ví dụ: Regex giúp kiểm tra tính hợp lệ của một email: `/[\w._%+-]+@[\w.-]+\.[a-zA-Z]{2,4}/`

3.3 Trích xuất và phân tích dữ liệu

Regex hỗ trợ trích xuất chi tiết từ các tệp nhật ký (logs) hoặc dữ liệu bán cấu trúc:

- Trích xuất timestamps, địa chỉ IP, và thông báo lỗi từ log.
- Phân tích dữ liệu từ các file như CSV, XML hoặc JSON, chuyển đổi qua các thành phần dễ hiểu.

Ví dụ: Một mẫu Regex có thể phân tách cấu trúc dữ liệu nhật ký để lấy thông tin chi tiết từ log.

3.4 Bảo mật mạng

Biểu thức chính quy đóng vai trò quan trọng trong bảo mật mạng. Các ứng dụng bao gồm:

- Phân tích lưu lượng mạng để phát hiện các mẫu tấn công nguy hiểm.
- Xác thực định dạng hợp lệ của địa chỉ IP trong log, ngăn chặn lỗi từ dữ liệu không phù hợp.

Ví dụ: Regex kiểm tra IP hợp lệ có thể như `\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}`.