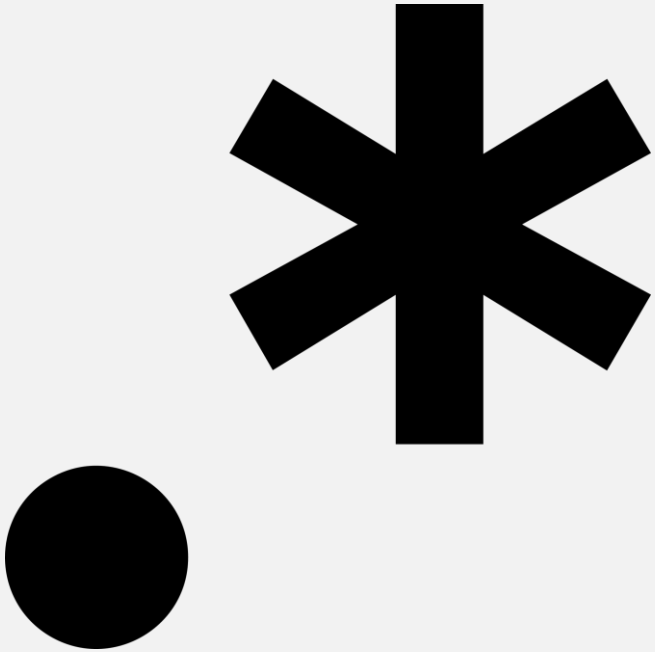
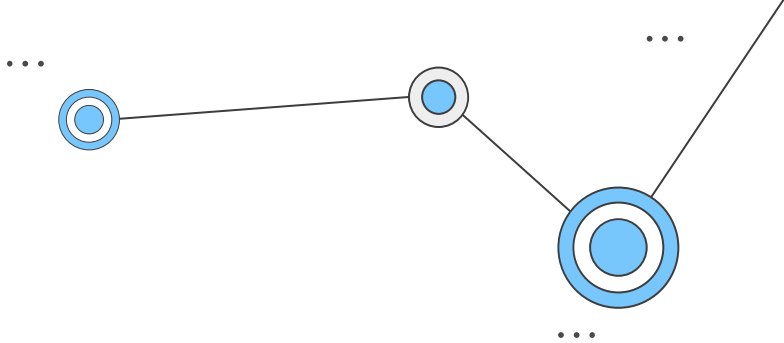
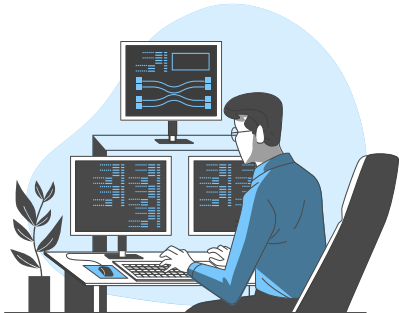




Khóa học Frontend

Bài 30: Học Regex cơ bản



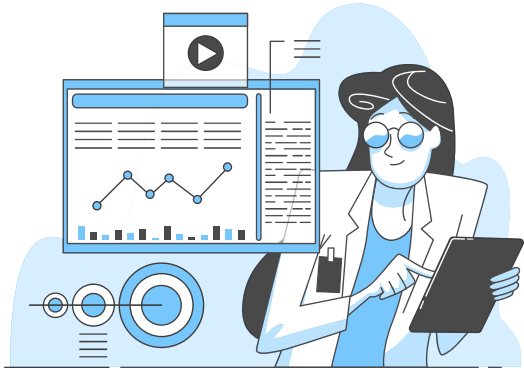
Nội dung

01 Khái niệm

02 Flag

03 Pattern

04 Bài tập thực hành



01

Khái niệm

01. Khái niệm

- **Regex** (**Reg**ular **Ex**pressions - Biểu thức chính quy)
- Regex là một công cụ giúp **tìm kiếm những từ** hoặc **nhóm từ có chung một đặc điểm** nào đó.
- Regex gồm 2 phần chính: **/ab+c/g**
  - **ab+c**: pattern
  - **g**: flag
- Công cụ để test: https://regexr.com

02

Flag

2.1. Flag g

- **g** (global search): Tìm kiếm ở phạm vi toàn cục.
- Giúp tìm kiếm cả đoạn văn.

2.2. Flag i

- **i** (case-insensitive search): Tìm kiếm không phân biệt chữ hoa, chữ thường.

2.3. Flag m *(Hướng dẫn sau)*

- **m** (multi-line search): Cho phép tìm kiếm trên nhiều dòng.

03

Pattern

3.1. Assertions (Điểm neo)

- Kí hiệu **^**
  - Để tìm kiếm chuỗi ở **đầu dòng**.
- Kí hiệu **$**
  - Để tìm kiếm chuỗi ở **cuối dòng**.
- Kí hiệu **\b** (boundary - ranh giới)
  - Để thiết lập ranh giới tìm kiếm chuỗi.

đặt ở đầu

```
/^Tôi/gi
```

Text Tests NEW

```
Xin chào các bạn! Tôi tên là Nam, tôi đã có vợ. Tôi đã xin vợ 15000đ để đi ăn sáng.
```

```
/tôi$/gi
```

Text Tests NEW

```
Xin chào các bạn! Tôi tên là Nam, tôi đã có vợ. Tôi đã xin vợ 15000đ để đi ăn sáng
```

đặt ở cuối

Expression

```
/\bABC/gi
```

Text Tests NEW

```
Tôi thích ABC và XYZ, nhưng không thích ABCXYZ và XYZABC
```

Expression

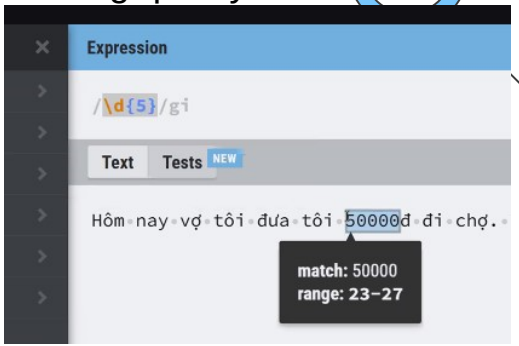
```
/\bABC\b/gi
```

Text Tests NEW

```
Tôi thích ABC và XYZ, nhưng không thích ABCXYZ và XYZABC
```

3.2. Character Classes (Các kiểu kí tự)

dùng qualify



để tìm số có 2 chữ số \d\d

để tìm số có 3 chữ số \d\d\d

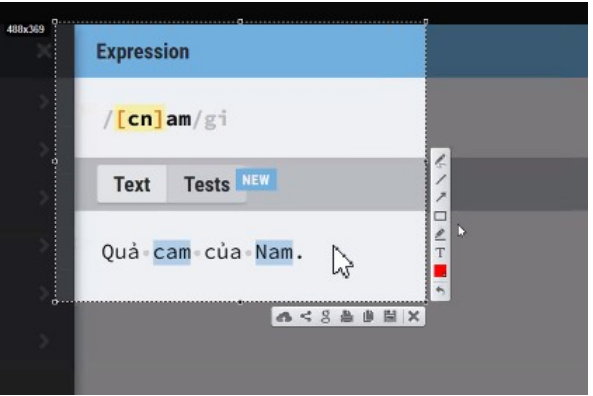
- Kí hiệu **\d**
  - Tìm các kí tự là số.
- Kí hiệu **\D**
  - Tìm các kí tự không phải là số.
- Kí hiệu **\w**
  - Tìm tất cả các kí tự (chữ hoa, chữ thường, chữ số, dấu gạch dưới, ngoại trừ khoảng trắng).
  - Tương tự: [a-zA-Z0-9]
  - Các kí tự tiếng Việt không được hỗ trợ.

không có dấu .

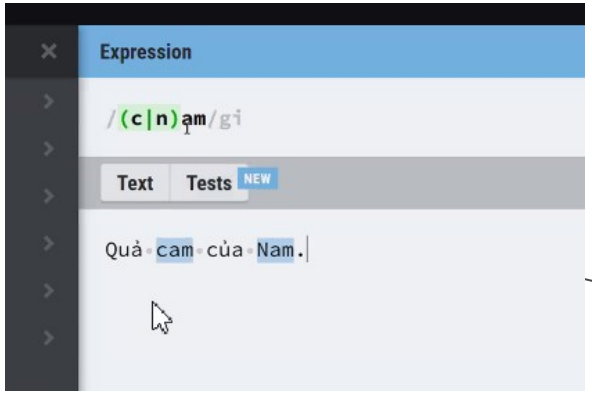
- Kí hiệu **\s**
  - Tìm các kí tự khoảng trắng.
- Kí hiệu **.** *(Dấu chấm)*
  - Tìm tất cả các kí tự.

3.3. Sets and Ranges
(Tập hợp kí tự và khoảng kí tự)

- Kí hiệu **[abc]** (Tập hợp các kí tự)
  - Có thể khớp với a, b hoặc c.
- Kí hiệu **[a-z]** (Khoảng kí tự)
  - Khớp với một trong các chữ từ a tới z.

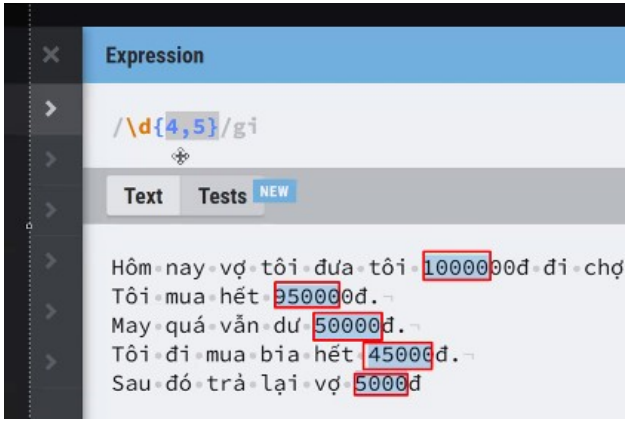


2 cách viết tương tự nhau



3.4. Quantifiers (Nhóm ký tự định lượng)

- Nhóm ký tự định lượng: Giúp không phải lặp đi lặp lại một pattern nhiều lần.
- Kí hiệu **{n}**
  - Xuất hiện **n** lần.
- Kí hiệu **{n, m}**
  - Xuất hiện trong khoảng **n** tới **m** lần.
- Kí hiệu **{n,}**
  - Xuất hiện **ít nhất n** lần.
- Kí hiệu **+**
  - Xuất hiện **1** hoặc **nhiều** lần.
  - Viết ngắn gọn cho **{1,}**.



3.4. Quantifiers (Nhóm ký tự định lượng)

- Nhóm ký tự định lượng: Giúp không phải lặp đi lặp lại một pattern nhiều lần.

- Kí hiệu *****
  - Xuất hiện **0** hoặc **nhiều** lần.
  - Viết ngắn gọn cho **{0,}**.

- Kí hiệu **?**
  - Xuất hiện **0** hoặc **1** lần.
  - Viết ngắn gọn cho **{0,1}**.

3.5. Group (Nhóm)

- Kí hiệu **(ab)**
  - Khớp với một nhóm các kí tự.

kiểu như các kí tự ab sẽ được lặp lại (ab)+

04

Bài tập thực hành

Bài 1: Ký tự Cuối Cùng là "N"?

- Tạo ra một hàm nhận một chuỗi (một tên ngẫu nhiên). Nếu ký tự cuối cùng của tên là "n", trả về true, ngược lại trả về false.

- Ví dụ:
  - isLastCharacterN("Aiden") → true
  - isLastCharacterN("Piet") → false
  - isLastCharacterN("Bert") → false
  - isLastCharacterN("Dean") → true

- Ghi chú:
  - Hàm phải trả về một giá trị boolean (true hoặc false).
  - Hàm **test()** trong javascript để kiểm tra điều kiện đúng hay không và trả về true hoặc false.

Bài 2: 28Tech is amazing.

- Tạo ra một hàm nhận một chuỗi và thay đổi từ "amazing" thành "not amazing".
- Nếu trong chuỗi có từ 28Tech thì giữ nguyên chuỗi không thay đổi.

- Ví dụ:
  - amazing("28Tech is amazing.") ⟶ "28Tech is amazing."
  - amazing("Abc is amazing.") ⟶ "Abc is not amazing."
  - amazing("Xyz is amazing.") ⟶ "Xyz is not amazing."

- Ghi chú:
  - Hàm **replace()** trong javascript để thay thế một chuỗi thành một chuỗi mới.

Bài 3: Gõ tắt HTML

- Tạo ra một hàm nhận vào một chuỗi có dạng "tagName*n" sau đó trả về n thẻ tagName.

- Ví dụ:
  - convertStringToTagName("div*2") ⟶ " "
  - convertStringToTagName("p*1") ⟶ " "
  - convertStringToTagName("li*3") ⟶ "
"

Bài 4: Đếm số lượng chữ in hoa

- Cho một chuỗi các ký tự, đếm xem có bao nhiêu chữ in hoa trong đó?

- Ví dụ:
  - capitalLetters("fvLzpxmgXSDrobbgMVrc") ⟶ 6
  - capitalLetters("JMZWCneOTFLWYwBWxyFw") ⟶ 14
  - capitalLetters("mqeytbbjwqemcdrdsyvq") ⟶ 0

Bài 5: Loại phần tử khỏi một mảng

- Bạn được cho một mảng với các từ ngẫu nhiên nhưng chương trình của bạn không chấp nhận các từ bắt đầu bằng chữ cái viết hoa "C".
- Loại bỏ các từ không được chấp nhận và trả về mảng mới.

- Ví dụ:
  - accepted(["Ducks", "Bears", "Cats"]) ⟶ ["Ducks", "Bears"]
  - accepted(["cars", "trucks", "planes"] ⟶ ["cars", trucks", "planes"]
  - accepted(["Cans", "Worms", "Bugs", "Cold", "Beans"]) ⟶ ["Worms", "Bugs", "Beans"]

Bài 6: Kiểm tra email có hợp lệ không

- Viết một hàm cho người dùng nhập vào email và kiểm tra xem email có hợp lệ không.
- Ví dụ:
  - validateEmail("levana@gmail.com") ⟶ true
  - validateEmail("levanb12345@gmail.com") ⟶ true
  - validateEmail("levanc12.34@gmail.com") ⟶ true
  - validateEmail("levand@28tech.com.vn") ⟶ true
  - validateEmail("levane.123@stu.28tech.com.vn") ⟶ true
  - validateEmail("levane.123.stu.28tech.com.vn") ⟶ false
  - validateEmail("levane.123@gmail") ⟶ false
  - validateEmail("levane.123@gmail.") ⟶ false

Bài Tập

- *Gửi vào nhóm lớp trong buổi học.*