Contents

Biểu thức chính quy (hay regex) là một công cụ mạnh mẽ mà mỗi nhà phát triển nên biết. Nó có thể khớp với một chuỗi các ký tự dựa trên các thông số rất phức tạp mà có thể giúp bạn tiết kiệm rất nhiều thời gian khi xây dựng các trang web động.

Dù các nhà phát triển Web phải đối mặt với nhiều nhiệm vụ khác nhau hơn so với các nhà phát triển phần mềm, nhưng đa số trong đó vẫn có cùng mã nền tảng. Biểu thức chính quy hơi khó học lúc đầu, nhưng có thể rất mạnh mẽ khi được sử dụng một cách chính xác.

Phần khó khăn nhất là việc tìm hiểu cú pháp và học cách để viết mã regex của riêng bạn từ đầu. Để tiết kiệm thời gian, tôi đã chọn ra 30 đoạn mã regex khác nhau mà bạn có thể sử dụng trong các dự án của bạn. Và kể từ khi regex không giới hạn cho một ngôn ngữ cụ thể, bạn có thể áp dụng những đoạn dưới đây vào bất cứ ngôn ngữ nào từ Javascript đến PHP hoặc Python.

1. Độ mạnh của mật khẩu

```
^(?=.*[A-Z].*[A-Z])(?=.*[!@#$&*])(?=.*[0-9].*[0-9])(?=.*[a-z].*[a-z].*[a-z]).{8}$
```

Kiểm tra độ mạnh của một mật khẩu thường là chủ quan nên không có câu trả lời chính xác tuyệt đối. Nhưng tôi cảm thấy đoạn regex này là một điểm khởi đầu tuyệt vời nếu bạn không muốn phải viết riêng hàm kiểm tra độ mạnh mật khẩu của bạn từ đầu.

2. Mã màu Hex

```
\#([a-fA-F]|[0-9]){3, 6}
```

Mã màu hex rất phổ biến trong lĩnh vực phát triển web. Đoạn regex này có thể được sử dụng để lấy mã hex phù hợp từ chuỗi bất kỳ cho bất cứ mục đích nào.

3. Xác thực địa chỉ Email

```
/[A-Z0-9._%+-]+@[A-Z0-9-]+.+.[A-Z]{2,4}/igm
```

Một trong những nhiệm vụ phổ biến nhất đối với một nhà phát triển là kiểm tra một chuỗi có phải là một địa chỉ Email hay không. Đối với PHP, các bạn có thể sử dụng hàm filter_var: `filter_var($email, FILTER_VALIDATE_EMAIL)`

4. Địa chỉ IPv4

```
/\b(?:(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\b/
```

Tương tự như một địa chỉ email là địa chỉ IP - được sử dụng để xác định một máy tính cụ thể truy cập Internet. Biểu thức chính quy này sẽ kiểm tra một chuỗi xem nó có tuân theo cú pháp địa chỉ IPv4 hay không.

5. Địa chỉ IPv6

```
(([0-9a-fA-F]{1,4}:){7,7}[0-9a-fA-F]{1,4}|([0-9a-fA-F]{1,4}:){1,7}:|([0-9a-fA-F]{1,4}:){1,6}:[0-9a-fA-F]{1,4}|([0-9a-fA-F]{1,4}:){1,5}(:[0-9a-fA-F]{1,4}){1,2}|([0-9a-fA-F]{1,4}:){1,4}(:[0-9a-fA-F]{1,4}){1,3}|([0-9a-fA-F]{1,4}:){1,3}(:[0-9a-fA-F]{1,4}){1,4}|([0-9a-fA-F]{1,4}:){1,2}(:[0-9a-fA-F]{1,4}){1,5}|[0-9a-fA-F]{1,4}:((:[0-9a-fA-F]{1,4}){1,6})|:((:[0-9a-fA-F]{1,4}){1,7}|:)|fe80:(:[0-9a-fA-F]{0,4}){0,4}%[0-9a-zA-Z]{1,}|::(ffff(:0{1,4}){0,1}:){0,1}((25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])\.){3,3}(25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])|([0-9a-fA-F]{1,4}:){1,4}:((25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])\.){3,3}(25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9]))
```

Hoặc có thể bạn sẽ muốn kiểm tra một địa chỉ theo cú pháp IPv6 mới hơn với đoạn regex nâng cao hơn này. Sự khác biệt là rất nhỏ mặc dù nó quan trọng trong quá trình phát triển.

6. Dấu phân cách hàng nghìn

```
/\d{1,3}(?=(\d{3})+(?!\d))/g
```

Mã regex này hoạt động trên bất kỳ số nào và sẽ áp dụng bất cứ dấu phân cách nào bạn chọn cho mỗi chữ số thứ ba phân tách thành hàng ngàn, hàng triệu,...

7. Thêm HTTP vào trước liên kết

```
if (!s.match(/^[a-zA-Z]+:\/\//))
{
    s = 'http://' + s;
}
```

Cho dù bạn đang làm việc trong JavaScript, Ruby hay PHP, biểu thức này có thể tỏ ra rất hữu ích. Nó sẽ kiểm tra bất kỳ chuỗi URL nào để xem nếu nó có tiền tố HTTP/HTTPS hay không, và nếu không, thêm vào trước chuỗi đó cho phù hợp.

8. Lấy tên miền từ URL

```
/https?:\/\/(?:[-\w]+\.)?([-\w]+)\.\w+(?:\.\w+)?\/?.*/i
```

Mỗi tên miền trang web đều chứa giao thức ở đầu (HTTP hoặc HTTPS) và đôi khi có tên miền phụ với đường dẫn trang bổ sung. Bạn có thể sử dụng đoạn regex này để cắt qua tất cả những điều đó và chỉ trả lại tên miền.

9. Sắp xếp các từ khóa bằng cách đếm số từ

```
^[^\s]*$        matches exactly 1-word keyword
^[^\s]*\s[^\s]*$    matches exactly 2-word keyword
^[^\s]*\s[^\s]*     matches keywords of at least 2 words (2 and more)
^([^\s]*\s){2}[^\s]*$    matches exactly 3-word keyword
^([^\s]*\s){4}[^\s]*$    matches 5-words-and-more keywords (longtail)
```

Những ai dùng Google Analytics và Webmaster Tools sẽ thực sự thích biểu thức chính quy này. Nó có thể sắp xếp các từ khoá dựa trên số các từ được sử dụng trong một tìm kiếm.

Đây có thể là số lượng cụ thể (tức là chỉ có 5 từ) hoặc nó có thể phù hợp với một loạt các từ (tức là 2 hoặc nhiều hơn). Khi được sử dụng để sắp xếp và phân tích dữ liệu, nó là một trong những biểu thức chính quy mạnh mẽ.

10. Tìm một chuỗi Base64 hợp lệ trong PHP

```
\?php[ \t]eval\(base64_decode\(\'(([A-Za-z0-9+/]{4})*([A-Za-z0-9+/]{3}=|[A-Za-z0-9+/]{2}==)?){1}\'\)\)\;
```

Nếu bạn là một nhà phát triển PHP thì đôi khi bạn có thể cần phân tích qua code để tìm kiếm các đối tượng nhị phân đã mã hóa Base64 (ví dụ như tìm shell được giấu trong một tập tin PHP chẳng hạn). Đoạn regex này có thể áp dụng với tất cả code PHP và sẽ kiểm tra bất cứ chuỗi Base64 nào đang tồn tại.

11. Xác thực số điện thoại

```
^\+?\d{1,3}?[- .]?\(?(?:\d{2,3})\)?[- .]?\d\d\d[- .]?\d\d\d\d$
```

Ngắn gọn, ngọt ngào và đi thẳng vào vấn đề. Đoạn regex này sẽ xác thực bất cứ cú pháp số điện thoại nào dựa trên phong cách số điện thoại Mỹ.

Điều này có thể biến thành một chủ đề khá phức tạp, tôi khuyên bạn nên đọc lướt qua chủ đề stack nàycho câu trả lời chi tiết hơn.

12. Khoảng trắng ở đầu và cuối

```
^[ \s]+|[ \s]+$
```

Sử dụng đoạn mã này để lấy ra khoảng trắng ở đầu/cuối từ một chuỗi. Đây có thể không phải là một việc lớn, nhưng đôi khi nó ảnh hưởng đến đầu ra khi lấy kết quả ra từ một cơ sở dữ liệu hoặc áp dụng vào mã hóa tài liệu.

13. Lấy nguồn ảnh

```
\< *[img][^\>]*[src] *= *[\"\']{0,1}([^\"\'\ >]*)
```

Nếu vì một số lý do nào đó mà bạn cần lấy ra một nguồn ảnh trực tiếp từ HTML thì đoạn mã này là giải pháp hoàn hảo. Mặc dù nó có thể chạy trơn tru trên Backend, các nhà phát triển JS nên dựa vào phương thức .attr() của jQuery cho Frontend.

14. Xác thực ngày trong định dạng DD/MM/YYYY

```
^(?:(?:31(\/|-|\.)(?:0?[13578]|1[02]))\1|(?:(?:29|30)(\/|-|\.)(?:0?[1,3-9]|1[0-2])\2)
)(?:(?:1[6-9]|[2-9]\d)?\d{2})$|^(?:29(\/|-|\.)0?2\3(?:(?:(?:1[6-9]|[2-9]\d)?(?:0[48]|
[2468][048]|[13579][26])|(?:(?:16|[2468][048]|[3579][26])00))))$|^(?:0?[1-9]|1\d|2[0-
8])(\/|-|\.)(?:(?:0?[1-9])|(?:1[0-2]))\4(?:(?:1[6-9]|[2-9]\d)?\d{2})$
```

Các ngày khó khăn vì chúng có thể xuất hiện dưới dạng văn bản + số, hoặc chỉ là số với các định dạng khác nhau. PHP có một hàm ngày tuyệt vời nhưng điều này không phải luôn là lựa chọn tốt nhất khi lấy ra từ một chuỗi thô (raw). Cân nhắc việc thay thế bằng cách sử dụng biểu thức chính quy này cho các cú pháp ngày cụ thể.

15. Khớp ID Video Youtube

```
/https?:\/\/(?:youtu\.be\/|(?:[a-z]{2,3}\.)?youtube\.com\/watch(?:\?|#\!)v=)([\w-]{11
}).*/gi
```

Youtube đã giữ cấu trúc URL giống nhau trong nhiều năm chỉ bởi nó hoạt động. Đây cũng là trang chia sẻ video phổ biến nhất trên web, vì thế các video Youtube có xu hướng điều phối lưu lượng truy cập nhiều nhất.

Nếu bạn cần lấy ra một ID video Youtube từ một URL thì đoạn regex này là hoàn hảo và hoạt động tốt với tất cả các biến thể của cấu trúc URL Youtube.

16. Xác thực ISBN

```
/\b(?:ISBN(?:: ?| ))?((?:97[89])?\d{9}[\dx])\b/i
```

Sách được in theo một hệ thống số gọi là ISBN. Điều này có thể lấy khá khó khăn khi bạn xem xét sự khác biệt giữa ISBN-10 và ISBN-13.

Tuy nhiên đoạn regex đáng kinh ngạc này cho phép bạn để xác nhận một số ISBN và kiểm tra xem nó là ISBN10 hay 13. Tất cả các mã được viết bằng PHP vì vậy điều này có thể tỏ ra đặc biệt hữu ích cho các nhà phát triển web.

17. Kiểm tra mã bưu điện

```
^\d{5}(?:[-\s]\d{4})?$
```

Tác giả của đoạn regex này không chỉ chia sẻ nó miễn phí mà còn dành thời gian để giải thích nó. Bạn sẽ thấy nó hữu ích cho dù bạn đang khớp một mã bưu điện loại 5 chữ số hay phiên bản dài 9 chữ số dài hơn.

Hãy nhớ rằng nó chủ yếu dành cho hệ thống mã bưu điện của Mỹ nên có thể bạn sẽ cần điều chỉnh cho các quốc gia khác.

18. Tên người dùng Twitter hợp lệ

```
/@([A-Za-z0-9_]{1,15})/
```

Đây là một đoạn mã regex rất nhỏ để khớp tên người dùng Twitter tìm thấy trong một chuỗi. Nó kiểm tra dựa trên cú pháp đề cập (@mention).

19. Số thẻ tín dụng

```
^(?:4[0-9]{12}(?:[0-9]{3})?|5[1-5][0-9]{14}|6(?:011|5[0-9][0-9])[0-9]{12}|3[47][0-9]{13}|3(?:0[0-5]|[68][0-9])[0-9]{11}|(?:2131|1800|35\d{3})\d{11})$
```

Chứng thực số thẻ tín dụng thường bắt buộc một nền tảng máy chủ an toàn. Nhưng regex có thể được sử dụng cho các yêu cầu tối thiểu của một số thẻ tín dụng điển hình.

20. Tìm thuộc tính CSS

```
^\s*[a-zA-Z\-]+\s*[:]{1}\s[a-zA-Z0-9\s.#]+[;]{1}
```

Có thể rất hiếm để chạy regex trên CSS nhưng nó không phải là một tình huống lạ.

Đoạn mã này có thể được sử dụng để lấy ra mọi thuộc tính và giá trị CSS khớp từ selector cụ thể. Nó có thể được sử dụng cho bất kỳ lý do nào, ví dụ như loại bỏ thuộc tính trùng lặp.

21. Loại bỏ chú thích HTML

```
<!--(.*?)-->
```

Nếu vì bất cứ lý do nào mà bạn cần phải loại bỏ tất cả các chú thích từ một khối HTML thì đây là mã regex cần dùng. Bạn sẽ tìm thấy một ví dụ PHP sử dụng preg_replace.

22. URL trang cá nhân Facebook

```
/(?:http:\/\/)?(?:www\.)?facebook\.com\/(?:(?:\w)*#!\/)?(?:pages\/)?(?:[\w\-]*\/)*([\w\-]*)/
```

Facebook rất phổ biến và có nhiều sự sắp đặt URL khác nhau. Trong một tình huống mà bạn đang lấy URL hồ sơ từ người sử dụng thì đoạn regex này có thể hữu ích để phân tích chuỗi và xác nhận rằng chúng có cấu trúc đúng. Đoạn này có thể làm chính xác điều đó và nó hoàn hảo cho tất cả các kiểu liên kết FB.

23. Kiểm tra phiên bản Internet Explorer

```
^.*MSIE [5-8](?:\.[0-9]+)?(?!.*Trident\/[5-9]\.0).*$
```

Trình duyệt IE vẫn đang có thị phần người dùng lớn và các nhà phát triển thường phải kiểm tra phiên bản IE để xử lý tính tương thích của trang web.

Đoạn này có thể được sử dụng trong JavaScript để kiểm tra phiên bản của Internet Explorer (5-11) đang được sử dụng dựa trên chuỗi User-Agent.

24. Bóc tách giá

```
/(\$[0-9,]+(\.[0-9]{2})?)/
```

Giá cả đi kèm trong một loạt các định dạng có chứa số thập phân, dấu phẩy và ký hiệu tiền tệ. Biểu thức này có thể kiểm tra tất cả các định dạng khác nhau để lấy ra một mức giá từ bất kỳ chuỗi nào.

25. Phân tích tiêu đề Email

```
/\b[A-Z0-9._%+-]+@(?:[A-Z0-9-]+\.)+[A-Z]{2,6}\b/i
```

Với một dòng code này bạn có thể phân tích thông qua một tiêu đề email để lấy ra thông tin "to" từ tiêu đề. Nó có thể được sử dụng song song với nhiều email liên kết với nhau.

26. Khớp một loại tập tin cụ thể

```
/^(.*\.(?!(htm|html|class|js)$))?[^.]*$/i
```

Khi bạn đang làm việc với các định dạng tập tin khác nhau như .xml, .html và .js, nó có thể giúp kiểm tra các tập tin cả nội bộ (local) và được tải lên bởi người dùng. Đoạn này lấy ra một phần mở rộng tập tin để kiểm tra xem nó có giá trị nằm trong một danh sách các phần mở rộng hợp lệ hay không.

27. Khớp một chuỗi URL

```
/[-a-zA-Z0-9@:%_\+.~#?&//=]{2,256}\.[a-z]{2,4}\b(\/[-a-zA-Z0-9@:%_\+.~#?&//=]*)?/gi
```

Đoạn này có thể sử dụng cho cả chuỗi HTTP và HTTPS để kiểm tra xem nó có khớp với cú pháp tên miền mức cao nhất (TLD) hay không. Cũng có một phiên bản đơn giản hơn đoạn này sử dụng RegExp của JavaScript.

28. Thêm rel="nofollow" vào các liên kết

```
(<a\s*(?!.*\brel=)[^>]*)(href="https?://)((?!(?:(?:www\.)?'.implode('|(?:www\.)?', $follow_list).'))[^"]+)"((?!.*\brel=)[^>]*)(?:[^>]*)>
```

Nếu bạn đang làm việc với hàng loạt mã HTML có thể nó thật khủng khiếp khi phải làm thủ công các nhiệm vụ lặp đi lặp lại. Biểu thức chính quy là hoàn hảo cho công việc này và họ sẽ tiết kiệm được rất nhiều thời gian.

Đoạn regex này có thể lấy ra tất cả liên kết từ một khối HTML và thêm thuộc tính rel="nofollow" vào mỗi phần tử.

29. Khớp Media Query

```
/@media([^{]+)\{([\s\S]+?})\s*}/g
```

Chia tách các Media Query CSS vào từng tham số và thuộc tính của chúng. Điều này có thể giúp bạn phân tích CSS bên ngoài với sự tập trung trực tiếp hơn vào cách mà code hoạt động.

30. Cú pháp tìm kiếm Google

```
/([+-]?(?:'.+?'|".+?"|[^+\- ]{1}[^ ]*))/g
```

Bạn có thể xây dựng mã regex của riêng bạn cho các thao tác tìm kiếm nội dung bằng cách sử dụng cú pháp độc quyền của Google. Các dấu cộng (+) biểu thị các từ khóa bổ sung và dấu trừ (-) biểu thị rằng nên bỏ qua và loại bỏ khỏi kết quả.

Đó là một đoạn khá phức tạp nhưng được sử dụng đúng cách thì nó có thể cung cấp một cơ sở cho việc xây dựng thuật toán tìm kiếm của riêng bạn.