Don’t Regret the RegEx

Learn it > Use it > Forget it > Repeat

[d( )] DASKOM 1337

Me

Fakhri - Fai - f4r4w4y

Github = https://github.com/fakhrip
Twitter = https://twitter.com/0xfa1
Web = https://justak.id

01

Background

To let you know the importance of regex

02

Introduction

A (Hopefully) gentle introduction to regex

03

Usage

Some examples of how and when to use regex

04

Challenge(s)

Lets see how far have you understand this...

01. Background

To let you know the importance of regex


ME
Why can't you just be normal?
regex
^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$

Search through codebase

- Find one or more functions
- Find one or more matching variables
- Look for some misconfigurations
- Check if there are some missed secrets that forgot to be removed/redacted

More general usage(s)

- Find files with matching filename
- Find files with matching contents
- Find that one directory you have been looking for ages
- Replace a string to another one inside one or more files
- Copy/Move/Delete/Read/(Do any things you want) with the matching files / dirs

For the leet boisss

- Clean a dataset
- Language parsing (hmm yeah why not ?)
  ex: Beyond regular expressions: Robust parsing of text input | by Alexander Olsson | Storytel Tech

- Parse a json/xml tree
- Make your automation script looks more smarter lol

02.

Introduction

A (Hopefully) gentle
introduction to regex

Literal Characters

Basically any characters that you can think of in your head + what actually exist on the keyboard

QWERTYUIOPASDFGHJKLZXCVBNM1234567890...

Meta-characters

This is a special characters that has a special meaning

**** = Marks the next character as either a special character or a literal.
^ = Matches the beginning of input.
$ = Matches the end of input.
\n = Matches a newline character.
\r = Matches a carriage return character.
[abc] = A character set. Matches any one of the enclosed characters.

and more

A good reference for this topic:
https://www.ibm.com/docs/en/rational-clearquest/9.0.1?topic=tags-meta-characters-in-regular-expressions

Quantifiers

This is meta-characters that has a purpose to quantify the matchers

***** = Matches the preceding character zero or more times.
+ = Matches the preceding character one or more times.
? = Matches the preceding character zero or one time.
{only} = ***only*** is a non-negative integer. Matches exactly ***only*** times.
{min,} = In this expression, ***min*** is a non-negative integer. Matches the preceding character at least ***min*** times.
{min,max} = The ***min*** and ***max*** variables are non-negative integers. Matches the preceding character at least ***min*** and at max ***max*** times.

A good reference for this topic:
https://www.ibm.com/docs/en/rational-clearquest/9.0.1?topic=tags-meta-characters-in-regular-expressions

Boolean operation a.k.a Assertions

This is meta-characters that has a purpose to do boolean operation, “do x if y stuff”

x|y = **Text alternation**, Matches either x or y.
x(?=y) = **Lookahead assertion**, Matches "x" only if "x" is followed by "y".
x(?!y) = **Negative lookahead assertion**, Matches "x" only if "x" is not followed by "y".
(?<=y)x = **Lookbehind assertion**, Matches "x" only if "x" is preceded by "y".
(?<!y)x = **Negative lookbehind assertion**, Matches "x" only if "x" is not preceded by "y".

A good reference for this topic:
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions/Assertions

Flags

This is actually language dependants, but in most cases (im talking about javascript)

g = Global search.
i = Case-insensitive search.
m = Multi-line search.
u = "unicode"; treat a pattern as a sequence of unicode code points.
s = Allows . to match newline characters.
y = Perform a "sticky" search that matches starting at the current position in the target string.

A good reference for this topic:
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions#advanced_searching_with_flags

03.

Usage

Some examples of how and when to use regex

grex

https://github.com/pemistahl/grex

I know RegEx is hard, painful, and complex, and let me tell you this secret,

its okay to not understand this, and just use **grex**

EX:

```
>> $ grex test tost tast tist tust

^t[aeiou]st$

>> $ grex color colours

^colo(?:urs|r)$
```

Other Bonus

https://regex-generator.olafneumann.org

Just type in what you need to parse using regex, click some button, and there you go

note: you sometimes need to tweak and play around with the result first


REGEX MORE LIKE REG EZ
imgflip.com

Enough for the shortcut, lets get down to the business real quick

RegEx + Grep

```
grep --include=\*.py -rnw . -e "^def .*\(\).*:$" -A1
```
> search for a function definition (including the first next line) in all python files throughout current directory

```
grep -rnw -P "(\d{1,4})\.(\d{1,4})\.(\d{1,4})\.(\d{1,4})" .
```
> find files with ip address in its content in all files throughout current directory

```
grep --include=\*.c -rnw -P "//" .
```
> find all comments in all files throughout current directory

RegEx + Grep + Grex

```
grep -P "$(grex -s 'hello world')" test.txt
```

> search for hello world string inside test.txt file

RegEx + Sed

```
sed -iE "s/^hello world$/halo dunia/g" test.txt
```
> search for **hello world** string and change it to **halo dunia** in test.txt file

```
sed -riE "s/(\w+)\s(\w+)/\2 \1/g" test.txt
```
> find string with 2 word delimited by a space, ex: **fakhri putra** and reverse it, ex: **putra fakhri** in test.txt file

04.

Challenge(s)

Lets see how far have you understand this...

Daskom1337 CTF Platform

link will be shared soon in the discord server

- There will be 5 challenges (at most)
- Each challenge will have different level of difficulty
- The challenge will run for unlimited times, but the one that will be counted for this week champion is until 5 september (after next week WCC)
- Challenges will goes public after certain amount of times, so that other people can also join in and hack together with us

[d( )] DASKOM 1337

“Learning regex is like learning vim, you dont have to, but once you did it, it always feels like you can do everything”

from a person who tried to make a laughable yet serious joke XD

Thanks!

Do you have any questions?

[d( )] DASKOM 1337