Regular Expression Reference Sheet

Author: Joff Thyer, © 2021, Rivergum Security LLC.

Version 20210426-A1

Types of Regular Expressions

POSIX BRE	POSIX Basic Regular Expression
POSIX ERE	POSIX Enhanced Regular Expression
PCRE	Perl Compatible Regular Expression

PCRE Character Classes

\w	single word character [A-Za-z0-9_]
\W	NOT a single word character
\d	single digit
\D	NOT a single digit
\s	space character (tab,space,newline)
\S	NOT a space character
\b	border/transition (\w <-> \W)
[A-F0-9]	custom character set with range
[^A-F0-9]	NOT/negation of character set
•	ANY single character / wildcard

POSIX Character Classes

[:alpha:]	single letter
[:alphanum:]	single letter or number
[:blank:]	space and tab
[:cntrl:]	control characters
[:digit:]	single digit
[:graph:]	visible characters
[:lower:]	lowercase letters
[:print:]	visible/printable characters w/ spaces
[:punct:]	punctuation and symbols
[:space:]	space character (tab,space,newline)
[:upper:]	uppercase letters
[:xdigit:]	hexadecimal digit

POSIX ERE Lookahead and Lookbehind

(?<=XXX)	Following XXX (lookbehind)
(?=XXX)	Followed by XXX (lookahead)
(?<!XXX)	Not Following XXX
(?!XXX)	Not Followed by XXX

Linux/UNIX Command Line Tool Regular Expression Support

grep -P	grep PCRE regular expressions
grep -E	grep POSIX ERE regular expressions
sed 's/ +//'	stream editor ERE regular expression removes spaces
bash (ERE)	[[$email =~ [a-zA-Z0-9_\-\.\+]+@[a-zA-Z0-9\.\-]+]] && echo "Valid Email Addr!"
vi	search/replace leading line spaces across whole file :1,$s/^ \+//g

Quantifiers

?	match previous character 0 or 1 time
*	match previous character zero or more times
+	match previous character one or more times
{x}	match previous character exactly x times
{x,y}	match previous character between x and y times

Disable Greedy After Quantifier with ?

[A-Z].+[^A-Z]	Greedy expression for one or more of anything between starting uppercase and ending NOT uppercase letter.
[A-Z].+?[^A-Z]	Disable Greedy behavior on same expression. Helps with bookended sorts of matches.

Anchors / Assertions

^	assert that characters following must match from the beginning of the string data
$	assert that prior characters must match only at the end of string data

Behavior Modifiers

(?i)	disable case sensitivty in this pattern
(?m)	match beyond the end of a newline in the data
(?s)	wildcard (dot) also matches a newline character

Logical OR

(0[1-9]|1[0-2])	parentheses and pipe "|" symbol define logical OR in pattern for matching
(?:0[1-9]|1[0-2])	adding the ?: characters after first parenthesis will disable capturing group behavior

Capturing Groups

(\d{1,3})\.(\d{1,3})\. (\d{1,3})\.(\d{1,3})	Group 1, Group 2, Group 3, Group 4 Group numbering starts from 1. Extracting data from sub-groups depends on language/tool implementation.

PCRE Backreferencing Groups

([\'"])(https?://.+?)(\1)	The "\1" in group 3 backreferences group 1 (red)
(?P<quote>[\'"])(https?://.+?)(?P=quote)	Python allows us to created named groups and backreference

Python RE Module Functions

re.findall(<pattern>, <data>)	Find all occurances of pattern in data and return a Python list with results.
re.match(<pattern>, <data>)	Find occurrances of pattern from the beginning of data, and return a "rexp" object to use with group().
re.search(<pattern>, <data>)	Find occurrances of pattern anywhere in data, and return a "rexp" object to use with group(). (>CPU usage)
re.compile(<pattern>)	Pre-compile a regular expression and return a regular expression object for use with findall()/match() or search()
rexp.group(<n>)	Extract captured group data from regular expression object. Group numbers count from 1. Group argument can be a string for named groups.

Note: Use the raw string modifier for patterns in python! re.findall(r'\d+', 'my age is 25')

Example Regular Expressions

VISA Card Number	^4[0-9]{12}(?:[0-9]{3})?$
Master Card Number	^(?:5[1-5][0-9]{2}|222[1-9]|22[3-9][0-9]|2[3-6][0-9]{2}|27[01][0-9]|2720)[0-9]{12}$
Single Octet of IPv4 Address	(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)

PowerShell Regular Expression Support

PS C:> Get-ChildItem -Path X: | Select-String -Pattern '\d\d\d-\d\d-\d\d\d\d'	Match a US Social Security Number in File System
PS C:> '199.12.13.14' -match '[\d\.]{7,15}' True	Match an IPv4 address in a naïve fashion
PS C:\> $m = "My CC number is 1234-4321-9876-1212" PS C:\> $m -replace '\d{4}', 'xxxx' My CC number is xxxx-xxxx-xxxx-xxxx	Perform a regular expression based replacement of text