Regular expressions

1.1. Cheatsheet

Regex Cheatsheet (MIT)

Davechild Regular Expressions (Cheatography)

1.2. QuickStart

From http://www.rexegg.com/regex-quickstart.html[1]

Character	Legend	Example	Sample Match
\d	One digit	file_\d\d	file_25
\w	One "word character": letter, underscore or digit	\w-\w\w\w	A-b_1
\s	One white space character (e.g.: a tab)	ab\s\s\sc	ab c
\D	One character that is not a digit	\D\D\D	ABC
\W	One character that is not a word character	\W\W\W\W\W	*-+=)
\S	One character that is not a space	\S\S\S\S	Yoyo

Quantifier	Legend	Example	Sample Match
+	One or more	Version \w-\w+	Version A-b1_1
{3}	Exactly three times	\D{3}	ABC
{2,4}	Two to four times	\d{2,4}	156
{3,}	Three or more times	\w{3,}	regex_tutorial
*	Zero or more times	A*B*C*	AAACC
?	Once or none	plurals?	plural

Character	Legend	Example	Sample Match
.	Any character except new line	a.c	abc
.	Any character except new line	.*	whatever, man.
\\.	A period (special character: needs to be escaped by a \\)	a\\.c	a.c
\\	Escapes a special character	\\.*\\+\\? \\$\\ \\/\\\\	.*+? $/\\
\\	Escapes a special character	\\\\{\\(\\)\\}\\	{()}

Logic	Legend	Example	Sample Match
|	OR operand	22|33	33
()	Capturing group	A(nt|pple)	Apple (captures "pple")
\1	Contents of Group 1	r(\w)g\1x	regex
\2	Contents of Group 2	(\d\d)\+(\d\d)=\2\+\1	12+65=65+12
(?:	Non-capturing group	A(?:nt|pple)	Apple

Character	Legend	Example	Sample Match
\t	Tab	T\t\w{2}	T ab
\r	Return character	see below	
\n	New line character	see below	
\r\n	New line in Windows	AB\r\nCD	AB CD

Quantifier	Legend	Example	Sample Match
+	The + (one or more) is "greedy"	\d+	12345
?	Makes quantifiers "lazy"	\d+?	1 in **1**2345
*	The * (zero or more) is "greedy"	A*	AAA
?	Makes quantifiers "lazy"	A*?	empty in AAA
{2,4}	Two to four times, "greedy"	\w{2,4}	abcd
?	Makes quantifiers "lazy"	\w{2,4}?	ab in **ab**cd

Character	Legend	Example	Sample Match
stuff	One of the characters in the brackets	AEIOU	One uppercase vowel
-	Range indicator	a-z	One lowercase letter
stuff	One of the characters in the brackets	AB1-5w-z	One of either: A,B,1,2,3,4,5,w,x,y,z
stuff	One of the characters in the brackets	A-Z+	GREAT
^x	One character that is not x	^a-z{3}	A1!
\d\D	One character that is a digit or a non-digit	\d\D+	Any characters, inc- luding new line

Anchor	Legend	Example	Sample Match
	Beginning of line (but means "not" inside ^brackets)	abc .*	abc (line start)
$	End of line	.*? the end$	this is the end
\A	Beginning of string	\Aabc\d\D*	abc (string... ...start)
\Z	End of string	\d\D*the end\Z	this is... ...the end
\b	Word boundary	Bob.*\bcat\b	Bob ate the cat
\B	Not a word boundary	Bob.*\Bcat\B.*	Bobcats

Character	Legend	Example	Sample Match
:alpha:	Letters	[8:alpha:]+	WellDone88
:alnum:	Letters and numbers	[[:alnum:]]{10}	ABCDE12345
:punct:	Punctuation marks	[[:punct:]]+	?!.,:;

Modifier	Legend	Example	Sample Match
(?i)	Case-insensitive	(?i)Monday	monDAY
(?s)	The dot (.) matches new line characters (\r\n)	(?s)From A.*to Z	From A to Z
(?m)	Treats the string as multiple lines, so that and $ can match in several places	(?m)1\r\n2$\r\n^3$	1 2 3
(?x)	Comment mode (aka whitespace mode)	(?x) # this is a # comment abc # write on multiple # lines d # spaces must be # in brackets	abc d

Lookaround	Legend	Example	Sample Match
(?=	Positive lookahead	(?=\d{10})\d{5}	01234 in**01234**56789
(?<=	Positive lookbehind	(?<=\d)cat	cat in 1**cat**
(?!	Negative lookahead	(?!theatre)the\w+	theme
(?<!	Negative lookbehind	\w{3}(?<!mon)ster	Munster

1.3. Styles

There perls & bash regexp styles, which are the common ones, maybe. There are other styles (vim, R, ...), but I don't plan to be fully comprehensive, here, since most styles are similar or can be run in perl-style with some flag/param/argument.

1.4. Ubuntu

Packages in Ubuntu 13.10 to help with regular expressions:

1.4.1. codeblocks-contrib: Regular expression testbed

http://codeblocks.org[2]

Click to expand

Plugin regexp for Codeblocks editor (Regular expression testbed)

1.4.2. kiki: Tool for python regular expression testing

~~http://project5.freezope.org/kiki~~ (broken)

Click to expand

A free environment for regular expression testing (ferret). It allows you to write regexes and test them against a sample text, providing
extensive output about the results. It is useful for several purposes:

- exploring and understanding the structure of match objects generated by the re module, making Kiki a valuable tool for people new to regexes.
- testing regexes on sample text before deploying them in code.

Kiki can function on its own or as plugin for the Spe Python editor.

1.4.3. redet: regular expression development and execution tool

http://www.billposer.org/Software/redet.html[3]

Click to expand

Redet allows the user to construct regular expressions and test them against input data by executing any of a variety of search programs, editors, and programming languages that make use of regular expressions. When a suitable regular expression has been constructed it may be saved to a file.

Redet stands for Regular Expression Development and Execution Tool. For each program, a palette showing the available regular expression syntax is provided. Selections from the palette may be copied to the regular expression window with a mouse click. Users may add their own definitions to the palette via their initialization file. Redet also keeps a list of the regular expressions executed, from which entries may be copied back into the regular expression under construction. The history list is saved to a file and restored on startup, so it persists across sessions.

So long as the underlying program supports Unicode, Redet allows UTF-8 Unicode in both test data and regular expressions.

1.4.4. rgxg: command-line tool to generate regular expressions

http://rgxg.sf.net[4]
rgxg (ReGular eXpression Generator) is a command-line tool to generate (extended) regular

expressions.

It can be useful to generate (extended) regular expressions to match for instance a specific number range (e.g. 0 to 31 or 00 to FF) or all addresses of a CIDR block (e.g. 192.168.0.0/24 or 2001:db8:aaaa::/64).

```
xavi@coprinus:~$ rgxg
Usage: rgxg COMMAND [ARGS]

The available rgxg commands are:
  alternation     Create regex that matches any of the given patterns
  cidr            Create regex that matches all addresses of the given CIDR block
  escape          Escape the given string for use in a regex
  range           Create regex that matches integers in a given range

Type 'rgxg help COMMAND' for help information on a specific command.

Type 'rgxg version' to see the version of rgxg.
xavi@coprinus:~$ rgxg help escape
Usage: rgxg escape [options] STRING

    -h         display this help message
```

1.4.5. txt2regex: A Regular Expression "wizard", all written with bash2 builtins

Click to expand

^txt2regex$ is a Regular Expression "wizard", all written with bash2 builtins, that converts human sentences to RegExs. With a simple interface, you just answer to questions and build your own RegEx for a large variety of programs, like awk, emacs, grep, perl, php, procmail, python, sed and vim. There are more than 20 supported programs.

1.4.6. visual-regexp: Interactively debug regular expressions

http://laurent.riesterer.free.fr/regexp/[5]

Click to expand

This Tcl script shows the result of running a regular expression, making debugging relatively easy. It

also assists in the construction
of regular expressions.

1.5. PHP online live regexp help

https://www.phpliveregex.com/[6]

1.6. Online tools

- https://regex101.com/[7]
- https://regexr.com/[8]
- https://www.regextester.com/[9]

Alias names for this page:
regexp | regexps | Regular Expression | regexpr

---

[1] http://www.rexegg.com/regex-quickstart.html

[2] http://codeblocks.org

[3] http://www.billposer.org/Software/redet.html

[4] http://rgxg.sf.net

[5] http://laurent.riesterer.free.fr/regexp/

[6] https://www.phpliveregex.com/

[7] https://regex101.com/

[8] https://regexr.com/

[9] https://www.regextester.com/