

RegExp cheat sheet

Character classes

Characters or ranges of characters inside square brackets `[]` match any of those characters.

e.g. `[hnc]ow` matches how, now and cow
e.g. `[1-4]` matches 1, 2, 3 and 4
e.g. `[A-Za-z]` matches any letter of the alphabet, upper- or lower-case

Shortcuts

`\d` means a digit, `[0-9]`

e.g. `\d\d` matches 00 – 99

`\s` means a whitespace character, `[\t\r\n]` (space, tab, newline)

`\w` means a 'word' character, `[A-Za-z0-9_]`

`.` means anything at all (except possibly a newline)

Quantifiers

Numbers inside curly brackets `{}` mean: match one or more repetitions of whatever came *immediately* before.

`{number}` means exactly *number*
`{min, max}` means a range between *min* and *max* inclusive
`{min,}` means at least *min*

e.g. `\d{16}` matches a credit card number (no spaces)
e.g. `ba{1,3}d` matches bad, baad and baaad
e.g. `\s{1,}` matches any amount of white space

Shortcuts

`?` means none or one, `{0,1}`
e.g. `expressions?` matches expression and expressions

`*` means zero or more, `{0,}`
e.g. `10*1` matches 11, 101, 1001, 10001, 100001, ...

`+` means one or more, `{1,}`
e.g. `\w+` matches a whole word

Groups

Round brackets `()` define groups, and these have several uses:

Quantified sequences

e.g. `(in)?flammable` matches flammable, inflammable

Alternatives with `|`

e.g. `\d+(st|nd|rd|th)` matches 1st, 2nd, 33rd, 404th, ...

Bracketed groups can be referenced as `$n` in your replacement text: `$1` is the first group, `$2` the second, ... (while `$0` is the whole match)

e.g. `19(\d0)s -> $1s` replaces 1960s -> 60s, 1980s -> 80s, etc.

e.g. `(March|April|May) (\d\d?), (\d{4}) -> $2 $1 $3` replaces May 4, 2014 -> 4 May 2014, etc.

Anchors

`^` matches the start of a line

`$` matches the end of a line

e.g. `^\d+$` matches any integer, but *only* if it's the only thing on a line

`\b` matches a word boundary

e.g. `ing\b` matches within going but not ingot

Negation

Inside a character class, `^` means not

e.g. `[^,.]` matches any single character except a comma or a full stop

Capitalised shortcuts have reversed meanings:

`\D` means a non-digit

`\S` means non-whitespace

`\W` means a non-word character

`\B` means not a word boundary

e.g. `ing\B` matches within ingot but not going

Greediness

By default, RegExp quantifiers are greedy: they match the longest sequence possible

e.g. in the text 1,2,3,4,5 `,.*`, matches `,2,3,4,`

If that's not what you want, there are two options:

1. Use an extra `?` to specify non-greediness, and match the shortest sequence possible (giving `??`, `*?`, `+?` and `{?}`)

e.g. `,.*?`, matches `,2,`

2. Be clearer about what you want to match

e.g. `,[^\,]*`, also matches `,2,`

Backreferences

We saw earlier that we can use the text matched by a capture group in our replacement expression — `$1`, `$2`, ...

But we can also use capture group text in our search expression — `\1`, `\2`, ...

e.g. `\b(\w)(\w)(\w)(\w?)\3\2\1\b` matches palindromes of 6 or 7 letters — e.g. redder, rotator, repaper, ...