Regular Expressions (Regex) Examples Cheatsheet

A comprehensive cheat sheet providing practical examples of regular expressions for various use cases. Master regex with these examples!

Basic Matching

Literal Matching

Pattern:	`hello`
Matches:	The literal string “hello”.
Example:	`/hello/ =~ "hello world"` `# => 0`
Pattern:	`world`
Matches:	The literal string “world”.
Example:	`/world/ =~ "hello world"` `# => 6`

Character Classes

Pattern:	`[aeiou]`
Matches:	Any single vowel (a, e, i, o, or u).
Example:	`/[aeiou]/ =~ "hello"` `# => 1`
Pattern:	`[0-9]`
Matches:	Any single digit (0 to 9).
Example:	`/[0-9]/ =~ "abc123def"` `# => 3`

Anchors

Pattern:	`^hello`
Matches:	“hello” only at the *beginning* of the string.

Example:	`/^hello/ =~ "hello world"`
	`# => 0`
	`/^hello/ =~ "world hello"`
	`# => nil`

Pattern:	`world$`
Matches:	“world” only at the *end* of the string.

Example:	`/world$/ =~ "hello world"`
	`# => 6`
	`/world$/ =~ "world hello"`
	`# => nil`

Quantifiers

Asterisk (*)

Pattern:	`a*`
Matches:	Zero or more occurrences of “a”.

Example:	`/a*/ =~ "bbbbb"`
	`# => 0 (matches empty string at the beginning)`
	`/a*/ =~ "aaaaab"`
	`# => 0`

Question Mark (?)

Pattern:	`a?`
Matches:	Zero or one occurrence of “a”.

Example:	`/a?/ =~ "bbbbb"`
	`# => 0 (matches empty string at the beginning)`
	`/a?/ =~ "aaaaab"`
	`# => 0`

Curly Braces ({})

Pattern:	`a{3}`
Matches:	Exactly three occurrences of “a”.

Example:	`/a{3}/ =~ "aaab"`
	`# => 0`
	`/a{3}/ =~ "aab"`
	`# => nil`

Pattern:	`a{2,4}`
Matches:	Between two and four occurrences of “a”.

Example:	`/a{2,4}/ =~ "aaab"`
	`# => 0`
	`/a{2,4}/ =~ "aaaaab"`
	`# => 0`
	`/a{2,4}/ =~ "aab"`
	`# => 0`

Plus (+)

Pattern:	`a+`
Matches:	One or more occurrences of “a”.

Example:	`/a+/ =~ "bbbbb"`
	`# => nil (no match)`
	`/a+/ =~ "aaaaab"`
	`# => 0`

Special Characters and Escaping

Dot (.)

Pattern:	`a.b`
Matches:	“a”, followed by any character (except newline), followed by “b”.

Example:	`/a.b/ =~ "acb"`
	`# => 0`
	`/a.b/ =~ "a\nb"`
	`# => nil`

Escaping Special Characters

Pattern:	`\.`
Matches:	A literal dot character.
Example:	`/\./ =~ "abc.def"` `# => 3`
Pattern:	`\*`
Matches:	A literal asterisk character.
Example:	`/\*/ =~ "abc*def"` `# => 3`

Character Classes Shorthand

Pattern:	`\d`
Matches:	Any digit (0-9).
Example:	`/\d/ =~ "abc123def"` `# => 3`
Pattern:	`\w`
Matches:	Any word character (a-z, A-Z, 0-9, and _).
Example:	`/\w/ =~ "abc_123"` `# => 0`

Grouping and Capturing

Basic Grouping

Pattern:	`(abc)+`
Matches:	One or more occurrences of the group “abc”.
Example:	`/(abc)+/ =~ "abcabcabc"` `# => 0`

Capturing Groups

Pattern:	`(\d+)-(\d+)-(\d+)`
Matches:	A date format like “YYYY-MM-DD”. Captures the year, month, and day in separate groups.

Example:	`/(\d+)-(\d+)-(\d+)/ =~ "2023-10-26"`
	`# => 0`
	`puts $1 # Output: 2023`
	`puts $2 # Output: 10`
	`puts $3 # Output: 26`

Non-Capturing Groups

Pattern:	`(?:abc)+`
Matches:	One or more occurrences of the group “abc”, but without capturing the group.

Example:	`/(?:abc)+/ =~ "abcabcabc"`
	`# => 0`
	`puts $1 # Output: nil`