A Gentle Introduction to Regex

INDEX

`regex example here`

regex example breakdown here

Inputs	Match
`what` is the pattern?	y/n
what is the matter?	y/n
is your name Pa`t?`	y/n

Just a string

matches *exactly*

`pat`

exactly "pat"

Inputs	Match
what is the `pat`tern?	y
what is the matter?	n
is your name Pat?	n

`the`

exactly "the"

Inputs	Match
what is `the` pattern?	y
what is `the` matter?	y
is your name Pat?	n

The Star Operator `*`

Zero or more of what's right before it

`th*e`

t, then zero or more h, then e

Inputs	Match
what is `the` pat`te`rn?	y
what is `the` mat`te`r?	y
is your name Pat?	n

`at*`

a, then zero or more t

Inputs	Match
wh`at` is the p`att`ern?	y
wh`at` is the m`att`er?	y
is your n`a`me P`at`?	y

The Optional Operator `?`

Zero or more of what's right before it

`at?t`

a, zero or one 1, t

Inputs	Match
wh`at` is the p`att`ern?	y
wh`at` is the m`att`er?	y
is your name P`at`?	n

`att?`

a, t, zero or one t

Inputs	Match
wh`at` is the p`att`ern?	y
wh`at` is the m`att`er?	y
is your name P`at`?	y

`h?at`

zero or one h, a, t

Inputs	Match
w`hat` is the p`at`tern?	y
w`hat` is the m`at`ter?	y
is your name P`at`?	y

The Plus Operator `+`

One or more of what's right before it

`at+`

a, one or more t

Inputs	Match
wh`at` is the p`att`ern?	y
wh`at` is the m`att`er?	y
is your name P`at`?	y

`r +`

r, one or more spaces

Inputs	Match
what is the pattern?	n
what is the matter?	n
what is you`r` name Pat?	y

The Dot Operator `.`

Any one character

`e .`

e, space, any character

Inputs	Match
what is th`e p`attern?	y
what is th`e m`atter?	y
is your nam`e P`at?	y

`r..`

r, any character, any character

Inputs	Match
what is the patte`rn?`	y
what is the matter?	n
is you`r n`ame Pat?	y

The Hat Operator (outside `[ ]`) `^`

Start of input anchor

`^is`

start of input, i, s

Inputs	Match
what is the the pattern?	n
what is the matter?	n
`is` your name Pat?	y

The Dollar Sign Operator `$`

End of input anchor

`at.$`

a, t, any character, end of input

Inputs	Match
what is the pattern?	n
what is the matter?	n
is your name P`at?`	y

The Square Brackets `[ ]`

Your choice of character

`a[tm]e`

a, t or m, e

Inputs	Match
what is the pattern?	n
what is the matter?	n
is your n`ame` Pat?	y

`a[tm]+e`

a, one or more (t or m), e

Inputs	Match
what is the p`atte`rn?	y
what is the m`atte`r?	y
is your n`ame` Pat?	y

`[whP]+at`

one or more (w, h, P), a, t

Inputs	Match
`what` is the pattern?	y
`what` is the matter?	y
is your name `Pat`?	y

The Dash Operator (only inside `[ ]`) `[ - ]`

A range of characters

`[a-zA-Z-at]`

any lower/upper letter, a, t

Inputs	Match
w`hat` is the `pat`tern?	y
w`hat` is the `mat`ter?	y
is your name `Pat`?	y

`[a-z] [v-z]`

any lower letter, space, vwxyz

Inputs	Match
what is the pattern?	n
what is the matter?	n
i`s y`our name Pat?	y

`[0-9]`

any number character

Inputs	Match

Inputs	Match
what is the pattern?	n
what is the matter?	n
is your name Pat?	n

The Hat Operator (only inside `[ ]`) `[ ^ ]`

Those characters? None of them (like a NOT operator)

`[^a-zA-Z]`

any non-letter character

Inputs	Match
what`is`the`pattern`?`	y
what`is`the`matter`?`	y
is`your`name`Pat`?`	y

The Parentheses `( )`

Used for grouping

> **NOTE** Parentheses are primarily used to capture groups of characters for replacements, which is covered more later~

`^(is)`

start of input, i, s

> NOTE Doesn't really effect things...

Inputs	Match
what is the pattern?	n

Inputs	Match
what is the matter?	n
`is` your name Pat?	y

`at( is)?`

a, t, optional(space, i, s)

> NOTE Compare with previous example!

Inputs	Match
wh`at is` the pattern?	y
wh`at is` the matter?	y
is your name P`at`?	y

Escape Operator Characters `\`

`at?\?`

a, optional t, question mark

> NOTE The backslash can escape the above operators like so: `\* \? \^`
> This example is shown to demonstrate this feature.

Inputs	Match
what is the pattern?	n
what is the matter?	n
is your name P`at?`	y

Special Character Classes `\`

Convenient shorthands

Syntax	Meaning
`\s`	whitespace (space, tab, newline)
`\d`	digit
`\w`	word character (letter, digit, _)
`\S`	not whitespace
`\W`	not a word character