< Learn These Shortcuts (/app/dojos/python-regex)

Python Regex Cheat Sheet

Characters I

.	Match any character except newline
^	Match the start of the string
$	Match the end of the string
*	Match 0 or more repetitions
+	Match 1 or more repetitions
?	Match 0 or 1 repetitions

Special Sequences I

\A	Match only at start of string
\b	Match empty string, only at beginning or end of a word
\B	Match empty string, only when it is not at beginning or end of word
\d	Match digits # same as [0-9]
\D	Match any non digit # same as [^0-9]

Characters II

*?	Match 0 or more repetitions non-greedy
+?	Match 1 or more repeitions non-greedy
??	Match 0 or 1 repetitions non-greedy
\	Escape special characters
[]	Match a set of characters
[a-z]	Match any lowercase ASCII letter
[lower-upper]	Match a set of characters from lower to upper
[^]	Match characters NOT in a set
A|B	Match either A or B regular expressions (non-greedy)

Special Sequences II

\s	Match whitespace characters # same as [\t\n\r\f\v]
\S	Match non whitespace characters #same as [^ \t\n\r\f\v]

\w	Match unicode word characters # same as [a-zA-Z0-9_]
\W	Match any character not a Unicode word character # same as [^a-zA-Z0-9_]
\Z	Match only at end of string

Characters III

{m}	Match exactly m copies
{m,n}	Match from m to n repetitions
{,n}	Match from 0 to n repetitions
{m,}	Match from m to infinite repetitions
{m,n}**?**	Match from m to n repetitions non-greedy (as few as possible)

RE Methods I

re.compile(pattern, flags)	Compile a regular expression of pattern, with flags
re.match(pattern, string)	Match pattern only at beginning of string
re.search(pattern, string)	Match pattern anywhere in the string
re.split(pattern, string)	Split string by occurrences of patern
re.sub(pattern, str2, string)	Replace leftmost non-overlapping occurrences of pattern in string with str2

Groups I

(match)	Use to specify a group for which match can be retrieved later
(?:match)	Non-capturing version parenthesis (match cannot be retrieved later)
(?P<name>)	Capture group with name "name"
(?P=name)	Back reference group named "name" in same pattern
(?#comment)	Comment

Match Objects I

match.group("name")	Return subgroup "name" of match
match.groups()	Return tuple containing all subgroups of match
match.groupdict()	Return dict containing all named subgroups of match
match.start(group)	Return start index of substring match by group
match.end(group)	Return end index of substring matched by group
match.span(group)	Return 2-tuple start and end indices of group in match

Flags I

(?)	Extension notation (used to set flags)
a	ASCII-only matching flag

i	Ignore case flag
L	Locale dependent flag
m	Multi-line flag
s	Dot matches all flag
x	Verbose flag

Lookahead / Behind I

(?=match)	Lookahead assertion - match if contents matches next, but don't consume any of the string.
(?!match)	Negative lookahead assertion - match if contents do not match next
(?<=match)	Positive lookbehind assertion - match if current position in string is preceded by match
(?<!match)	Negative lookbehind assertion - match if current position is not preceded by match
(?(id/name)yes|no)	Match "yes" pattern if id or name exists, otherwise match "no" pattern

Match Objects II

match.pos	Value of pos which was passed to search() or match()
match.endpos	Value of endpos which was passed to search() or match()
match.lastindex	Integer index of last matched capturing group
match.lastgroup	Name of last matched capturing group
match.re	The regular expression who match() or search() created this match
match.string	The string passed to match() or search()

RE Methods II

re.fullmatch(pattern, string)	Match pattern if whole string matches regular expression
re.findall(pattern, string)	Return all non-overlapping matches of pattern in string, as a list of strings
re.finditer(pattern, string)	Return an iterator yielding match objects over non-overlapping matches of pattern in string
re.subn(pattern, str2, string)	Replace left most occurrences of pattern in string with str2, but return a tuple of (newstring, # subs made)
re.purge()	Clear the regular expression cache

< Learn These Shortcuts (/app/dojos/python-regex)

(https://www.facebook.com/shortcutfoo) (https://www.twitter.com/shortcutfoo)