Python Regular Expressions

SINGLE CHARACTERS

Use	To match any character
[*set***]**	In that set
[^*set***]**	Not in that set
[*a–z***]**	In the *a-z* range
[^*a–z***]**	Not in the a-z range
.	Any except \n (new line)
*****char*	Escaped special character

CONTROL CHARACTERS

Use	To match	Unicode
\t	Horizontal tab	\u0009
\v	Vertical tab	\u000B
\b	Backspace	\u0008
\e	Escape	\u001B
\r	Carriage return	\u000D
\f	Form feed	\u000C
\n	New line	\u000A
\a	Bell (alarm)	\u0007

NON-ASCII CODES

Use	To match character with
*****octal*	First digit 0 followed by 2 octal digits or 3 octal digits
\x *hex*	2-digit hex character code
\u *hex*	4-digit hex character code

CHARACTER CLASSES

Use	To match character
\w	Word character. [0-9_a-zA-Z] and Unicode word characters
\W	Non-word character
\d	Decimal digit and Unicode digits
\D	Not a decimal digit
\s	White-space character [\t\n\r\f\v] and Unicode spaces
\S	Non-white-space char

QUANTIFIERS

Greedy	Lazy	Matches
*	*?	0 or more times
+	+?	1 or more times
?	??	0 or 1 time
{*n*}	{***n***}?	Exactly *n* times
{*n*,}	{***n***,}?	At least *n* times
{*n*,*m*}	{***n***,***m***}?	From *n* to *m* times

ANCHORS

Use	To specify position
^	At start of string or line
\A	At start of string
\Z	At end of string
$	At end of string or line
\b	On word boundary
\B	Not on word boundary

GROUPS

Use	To define
(*exp***)**	Indexed group
(?P<*name***>***exp***)**	Named group
(?:*exp***)**	Noncapturing group
(?=*exp***)**	Zero-width positive lookahead
(?!*exp***)**	Zero-width negative lookahead
(?<=*exp***)**	Zero-width positive lookbehind. *exp* is fixed width
(?<!*exp***)**	Zero-width negative lookbehind. *exp* is fixed width

INLINE OPTIONS

Option	Effect on match
i	Case-insensitive
m	Multiline mode
L	Locale specific
u	Unicode dependent
s	Single-line mode
x	Ignore white space

BACKREFERENCES

Use	To match
n	Indexed group
(?P=*name*)	Named group

ALTERNATION

Use	To match
a **|** *b*	Either *a* or *b*
(?(*n*) *yes* | *no***)**	*yes* if *group n* is matched *no* if *group n* isn't matched
(?(*name*) *yes* | *no***)**	*yes* if *name* is matched *no* if *name* isn't matched

SUBSTITUTION

Use	To substitute
\g<n**>**	Substring matched by group number *n*
\g<*name*>	Substring matched by group *name*

COMMENTS

Use	To
(?# *comment***)**	Add inline comment
#	Add x-mode comment to end

REGULAR EXPRESSION OPERATIONS

Module: re

Pattern matching with Compiled objects

To initialize with	Use constructor
Pattern	re.compile(pattern)
+ flags	re.compile(pattern,flags)

Finding and replacing matched patterns. Use compiled object methods for additional options and fine-tuning parameters

Use method	To
re.match	Find match at start of string
re.search	Find the first match
re.findall	Retrieve all matching strings
re.finditer	Retrive all matches
re.sub	Replace a matching string
re.split	Split text based on match

Getting info about regular expression patterns

Use compiled object API	To get
groupindex	Dictionary of Group names and group number
groups	Capturing Group Count
pattern	Pattern for compiled object

Processing a match

Use method	To
expand	Replace a match
group	Retrieve value of a group by number or name
groups	Retrieve all subgroups as a tuple
groupdict	Retrieve dictionary of named groups and values
start	Find starting index position of a group
end	Find ending index position of a group