PHP PCRE Cheat Sheet

Functions

preg_match(pattern, subject[, submatches])
preg_match_all(pattern, subject[, submatches])
preg_replace(pattern, replacement, subject)
preg_replace_callback(pattern, callback, subject)
preg_grep(pattern, array)
preg_split(pattern, subject)

Base Character Classes

\w	Any “word” character (a-z 0-9 _)
\W	Any non “word” character
\s	Whitespace (space, tab CRLF)
\S	Any non whitepsace character
\d	Digits (0-9)
\D	Any non digit character
.	(Period) – Any character except newline

Meta Characters

^	Start of subject (or line in multiline mode)
$	End of subject (or line in multiline mode)
[	Start character class definition
]	End character class definition
|	Alternates, eg (a|b) matches a or b
(	Start subpattern
)	End subpattern
\	Escape character

Quantifiers

n*	Zero or more of n
n+	One or more of n
n?	Zero or one occurrences of n
{n}	n occurrences exactly
{n,}	At least n occurrences
{,m}	At most m occurrences
{n,m}	Between n and m occurrences (inclusive)

Pattern Modifiers

i	Caseless – ignore case
m	Multiline mode - ^ and $ match start and end of lines
s	Dotall - . class includes newline
x	Extended– comments & whitespace
e	preg_replace only – enables evaluation of replacement as PHP code
S	Extra analysis of pattern
U	Pattern is ungreedy
u	Pattern is treated as UTF-8

Point based assertions

\b	Word boundary
\B	Not a word boundary
\A	Start of subject
\Z	End of subject or newline at end
\z	End of subject
\G	First matching position in subject

Subpattern Modifiers & Assertions

(?:)	Non capturing subpattern	((?:foo|fu)bar) matches foobar or fubar without foo or fu appearing as a captured subpattern
(?=)	Positive look ahead assertion	foo(?=bar) matches foo when followed by bar
(?!)	Negative look ahead assertion	foo(?!bar) matches foo when *not* followed by bar
(?<=)	Positive look behind assertion	(?<=foo)bar matches bar when preceded by foo
(?<!)	Negative look behind assertion	(?<!foo)bar matches bar when *not* preceded by foo
(?>)	Once-only subpatterns	(?>\d+)bar Performance enhancing when bar not present
(?(x))	Conditional subpatterns	(?(3)foo|fu)bar Matches foo if 3rd subpattern has matched, fu if not
(?#)	Comment	(?# Pattern does x y or z)