Anchors (boundaries)

^	Start of string or line
$	End of string or line
\A	Start of input (ignores 'm' flag)
\Z	End of input (ignores 'm' flag)
\G	End of the previous match
\b	Word boundary (any position proceeded or followed - but not both - by a letter, digit or underscore)
\B	Non-word boundary

Character and Sets

\w	Word	[a-zA-Z0-9_]
\W	Non-word	[^a-zA-Z0-9_]
\d	Digit	[0-9]
\D	Non-digit	
\s	Whitespace (Form-feed, tab, vertical-tab, new line, carriage return and space)	[\f\t\x0b\n\r]
\S	Non-whitespace	
\x	Hexadecimal digit	\x00=null; \x0d=\r; [\x61-\x7a]=[a-z]
\O	Octal digit	
.	Any character (except new line \n)	

Groups

(...)	Capture group - captures a set of characters for a later expression
(?:...)	Non-capture group - groups an expression but does not capture. e.g. /((?:foo|fu)bar)/ matches "foobar" or "fubar" without "foo" or "fu" appearing as a captured subpattern
(?=...)	Lookahead - match on the characters following. e.g. /ab(?=c)/ match "ab" only when followed by "c"
(?!...)	Negative lookahead - match on characters that aren't following. e.g. /ab(?!c)/ match "ab" only when NOT followed by "c"
(?<=...)	Positive look-behind assertion. e.g. /(?<=foo)bar/ matches "bar" when preceded by "foo"
(?<!...)	Negative look-behind assertion. e.g. /(?<!foo)bar/ - matches "bar" when not preceded by "foo"
(?#...)	Comment e.g. (?# This comment is ignored entirely)

Unicode character support

\x0000-\xFFFF	Unicode hexadecimal character set
\x00-\xFF	ASCII hexadecimal character set
\cA-\cZ	Control characters

Unicode is not fully supported on all platforms. JavaScript prior to ES6 for example allows ASCII hex but not full Unicode hex.

Special Characters

\n	New line
\r	Carriage return
\t	Tab
\v	Vertical tab
\f	Form feed

Quantifiers

*	Zero or more
+	One or more
?	Zero or One (i.e. optional)
{n}	Exactly 'n' (any number)
{n,}	Minimum ('n' or more)
{n,m}	Range ('n' or more, but less or equal to 'm')

Flags (expression modifiers)

/m	Milti-line. (Makes ^ and $ match the start and end of a line respectively)
/s	Treat input as a single line. (Makes '.' match new lines as well)
/i	Case insensitive pattern matching.
/g	Global matching. (Don't stop after first match in a replacement function)
/x	Extended matching. (disregard white-space not explicitly escaped, and allow comments starting with #)

Escape Characters

In regular expressions, the following characters have special meaning and must be escaped: ^ $ [{ ( ) < > . * \ + | ?

Additionally the hyphen (-) and close square bracket (]) must be escaped when in an expression set ([...]).

e.g. /\(\d{3}\) \d{4}[\-]\d{4}/ matches "(nnn) nnnn-nnnn" or "(nnn) nnnn nnnn" (where n is a numeric digit).