

Anchors		Sample Patterns	
<code>^</code>	Start of line +	<code>([A-Za-z0-9-]+)</code>	Letters, numbers and hyphens
<code>\A</code>	Start of string +	<code>(\d{1,2}\V\d{1,2}\V\d{4})</code>	Date (e.g. 21/3/2006)
<code>\$</code>	End of line +	<code>([^\s]+(?:=\.(jpg gif png))\.\2)</code>	jpg, gif or png image
<code>\Z</code>	End of string +	<code>(^[1-9]{1}\$ ^[1-4]{1}[0-9]{1}\$ ^[50]\$)</code>	Any number from 1 to 50 inclusive
<code>\b</code>	Word boundary +	<code>(#[A-Fa-f0-9]{3}([A-Fa-f0-9]{3})?)</code>	Valid hexadecimal colour code
<code>\B</code>	Not word boundary +	<code>((?=\.\d)(?=[a-z])(?=[A-Z]).{8,15})</code>	8 to 15 character string with at least one upper case letter, one lower case letter, and one digit (useful for passwords).
<code>\<</code>	Start of word	<code>(\w+@[a-zA-Z_]+?\.[a-zA-Z]{2,6})</code>	Email addresses
<code>\></code>	End of word	<code>(\<(/?[^\>]+)\>)</code>	HTML Tags

Character Classes	
<code>\c</code>	Control character
<code>\s</code>	White space
<code>\S</code>	Not white space
<code>\d</code>	Digit
<code>\D</code>	Not digit
<code>\w</code>	Word
<code>\W</code>	Not word
<code>\xhh</code>	Hexadecimal character hh
<code>\Oxxx</code>	Octal character xxx

POSIX Character Classes	
<code>[:upper:]</code>	Upper case letters
<code>[:lower:]</code>	Lower case letters
<code>[:alpha:]</code>	All letters
<code>[:alnum:]</code>	Digits and letters
<code>[:digit:]</code>	Digits
<code>[:xdigit:]</code>	Hexadecimal digits
<code>[:punct:]</code>	Punctuation
<code>[:blank:]</code>	Space and tab
<code>[:space:]</code>	Blank characters
<code>[:cntrl:]</code>	Control characters
<code>[:graph:]</code>	Printed characters
<code>[:print:]</code>	Printed characters and spaces
<code>[:word:]</code>	Digits, letters and underscore

Assertions	
<code>?=</code>	Lookahead assertion +
<code>?!</code>	Negative lookahead +
<code>?<=</code>	Lookbehind assertion +
<code>?!= or ?<! </code>	Negative lookbehind +
<code>?></code>	Once-only Subexpression
<code>?()</code>	Condition [if then]
<code>?() </code>	Condition [if then else]
<code>?#</code>	Comment

Special Characters	
<code>\</code>	Escape Character +
<code>\n</code>	New line +
<code>\r</code>	Carriage return +
<code>\t</code>	Tab +
<code>\v</code>	Vertical tab +
<code>\f</code>	Form feed +
<code>\a</code>	Alarm
<code>[\b]</code>	Backspace
<code>\e</code>	Escape
<code>\N{name}</code>	Named Character

String Replacement (Backreferences)	
<code>\$n</code>	nth non-passive group
<code>\$2</code>	"xyz" in <code>/^(abc(xyz))\$/</code>
<code>\$1</code>	"xyz" in <code>/^(?:abc)(xyz)\$/</code>
<code>\$`</code>	Before matched string
<code>\$'</code>	After matched string
<code>\$+</code>	Last matched string
<code>\$&</code>	Entire matched string
<code>\$_</code>	Entire input string
<code>\$\$</code>	Literal "\$"

Quantifiers	
<code>*</code>	0 or more +
<code>*?</code>	0 or more, ungreedy +
<code>+</code>	1 or more +
<code>+</code>	1 or more, ungreedy +
<code>?</code>	0 or 1 +
<code>??</code>	0 or 1, ungreedy +
<code>{3}</code>	Exactly 3 +
<code>{3,}</code>	3 or more +
<code>{3,5}</code>	3, 4 or 5 +
<code>{3,5}?</code>	3, 4 or 5, ungreedy +

Ranges	
<code>.</code>	Any character except new line (<code>\n</code>) +
<code>(a b)</code>	a or b +
<code>(...)</code>	Group +
<code>(?:...)</code>	Passive Group +
<code>[abc]</code>	Range (a or b or c) +
<code>[^abc]</code>	Not a or b or c +
<code>[a-q]</code>	Letter between a and q +
<code>[A-Q]</code>	Upper case letter + between A and Q +
<code>[0-7]</code>	Digit between 0 and 7 +
<code>\n</code>	nth group/subpattern +

Pattern Modifiers	
<code>g</code>	Global match
<code>i</code>	Case-insensitive
<code>m</code>	Multiple lines
<code>s</code>	Treat string as single line
<code>x</code>	Allow comments and white space in pattern
<code>e</code>	Evaluate replacement
<code>U</code>	Ungreedy pattern

Metacharacters (must be escaped)		
<code>^</code>	<code>[</code>	<code>.</code>
<code>\$</code>	<code>{</code>	<code>*</code>
<code>(</code>	<code>\</code>	<code>+</code>
<code>)</code>	<code> </code>	<code>?</code>
<code><</code>	<code>></code>	

Note	
Note	Items marked + should work in most regular expression implementations.

Note	
Note	These patterns are intended for reference purposes and have not been extensively tested. Please use with caution and test thoroughly before use.