### Regular Expressions Anchors
- ^ Start of string, or start of line in multi-line pattern
- A Start of string
- $ End of string, or end of line in multi-line pattern
- Z End of string
- \b Word boundary
- \B Not word boundary
- \< Start of word
- \> End of word

### Regular Expressions Character Classes
- \c Control character
- \s White space
- \S Not white space
- \d Digit
- \D Not digit
- \w Word
- \W Not word
- \x Hexadecimal digit
- \O Octal digit

### Regular Expressions POSIX
- [upper:] Upper case letters
- [lower:] Lower case letters
- [alpha:] All letters
- [alnum:] Digits and letters
- [digit:] Digits
- [xdigit:] Hexadecimal digits
- [punct:] Punctuation
- [blank:] Space and tab
- [space:] Blank characters
- [cntrl:] Control characters
- [graph:] Printed characters
- [print:] Printed characters and spaces
- [word:] Digits, letters and underscore

### Regular Expressions Quantifiers
- * 0 or more
- + 1 or more
- ? 0 or 1
- (3) Exactly 3
- (3,) 3 or more
- (3,) 3, 4 or 5

Add a ? to a quantifier to make it ungreedy.

### Regular Expressions Escape Sequences
- \ Escape following character
- \Q Begin literal sequence
- \E End literal sequence

"Escaping" is a way of treating characters which have a special meaning in regular expressions literally, rather than as special characters.

### Regular Expression Common Metacharacters
- ^ [ -
- $ \{ "
- ( \ +
- ) | ?
- < >

The escape character is usually the backslash - \\

### Regular Expressions Special Characters
- \n New line
- \r Carriage return
- \t Tab
- \v Vertical tab
- \f Form feed
- \x\x Octal character \x\x
- \xhh Hex character hh

### Regular Expressions Groups and Ranges
- . Any character except new line (\n)
- (a|b) a or b
- (...) Group
- (?::) Passive (non-capturing) group
- [abc] Range (a or b or c)
- [^abc] Not a or b or c
- [a-q] Letter from a to q
- [A-Q] Upper case letter from A to Q
- [0-7] Digit from 0 to 7
- \n nth group/subpattern

Ranges are inclusive.

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

### Regular Expressions String Replacement
- $n nth non-passive group
- $2 "xyz" in "(abc\(xyz))"$
- $1 "xyz" in "(^\((abc)(xyz)\))"$
- $' Before matched string
- $³ After matched string
- $+ Last matched string
- $& Entire matched string

Some regex implementations use \ instead of $.

---

**Cheatography**

Regular Expressions Cheat Sheet  
by Dave Child (DaveChild) via cheatography.com/1/cs/5/  

**Regular Expressions Anchors**
- ^ Start of string, or start of line in multi-line pattern  
- A Start of string  
- $ End of string, or end of line in multi-line pattern  
- Z End of string  
- \b Word boundary  
- \B Not word boundary  
- \< Start of word  
- \> End of word

**Regular Expressions Character Classes**
- \c Control character  
- \s White space  
- \S Not white space  
- \d Digit  
- \D Not digit  
- \w Word  
- \W Not word  
- \x Hexadecimal digit  
- \O Octal digit

**Regular Expressions POSIX**
- [upper:] Upper case letters  
- [lower:] Lower case letters  
- [alpha:] All letters  
- [alnum:] Digits and letters  
- [digit:] Digits  
- [xdigit:] Hexadecimal digits  
- [punct:] Punctuation  
- [blank:] Space and tab  
- [space:] Blank characters  
- [cntrl:] Control characters  
- [graph:] Printed characters  
- [print:] Printed characters and spaces  
- [word:] Digits, letters and underscore

**Regular Expressions Quantifiers**
- * 0 or more  
- + 1 or more  
- ? 0 or 1  
- (3) Exactly 3  
- (3,) 3 or more  
- (3,) 3, 4 or 5

Add a ? to a quantifier to make it ungreedy.

**Regular Expressions Escape Sequences**
- \ Escape following character  
- \Q Begin literal sequence  
- \E End literal sequence

"Escaping" is a way of treating characters which have a special meaning in regular expressions literally, rather than as special characters.

**Regular Expression Common Metacharacters**
- ^ [ -
- $ \{ "
- ( \ +
- ) | ?
- < >

The escape character is usually the backslash - \\

**Regular Expressions Special Characters**
- \n New line  
- \r Carriage return  
- \t Tab  
- \v Vertical tab  
- \f Form feed  
- \x\x Octal character \x\x
- \xhh Hex character hh

**Regular Expressions Groups and Ranges**
- . Any character except new line (\n)
- (a|b) a or b
- (...) Group
- (?::) Passive (non-capturing) group
- [abc] Range (a or b or c)
- [^abc] Not a or b or c
- [a-q] Letter from a to q
- [A-Q] Upper case letter from A to Q
- [0-7] Digit from 0 to 7
- \n nth group/subpattern

Ranges are inclusive.

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

**Regular Expressions String Replacement**
- $n nth non-passive group
- $2 "xyz" in "(abc\(xyz))"$
- $1 "xyz" in "(^\((abc)(xyz)\))"$
- $' Before matched string
- $³ After matched string
- $+ Last matched string
- $& Entire matched string

Some regex implementations use \ instead of $.

---

**Cheatographer**

Dave Child (DaveChild) cheatography.com/davechild/  
www.addedbytes.com

**Cheat Sheet**

This cheat sheet was published on 19th October, 2011 and was last updated on 14th November, 2012.

**Sponsor**

FeedbackFair, increase your conversion rate today!  
Try it free!  
http://www.FeedbackFair.com