

Anchors/Ankers/Positions-Metacharakter

<code>^</code>	Am Anfang der Zeile	<code>^abc</code>	abc def ghi abc
<code>\$</code>	Am Ende der Zeile	<code>cba\$</code>	abc def hij fed cba
<code>\A</code>	Überprüft erste Buchstabe eines Strings	<code>\A(Apfel)</code>	Apfel kuchen Birnenkuchen
<code>\Z</code>	Überprüft das Ende eines Strings	<code>kuchen\Z</code>	Apfelkuchen Birnen kuchen
<code>\b</code>	Überprüft das Enden des Wortes	<code>kuchen\b</code>	Apfel kuchen Birnen kuchen
<code>\B</code>	Überprüft die Wörter/Zeichenketten innerhalb oder am Anfang	<code>o\B</code>	Lotto

Anchors/Ankers/Positions-Metacharakter werden dazu benutzt um innerhalb eines Strings zu suchen. Das unterscheidet diese von anderen Befehlen, da diese die einzelnen Charakter normalerweise betrachten.

Single-Meta-Charakter

<code>.</code>	Gib mir ein beliebiges Charakter (= a-z, A-Z, 0-9, ...) außer einer Nichtsichtbaren Charakter(z.B. Enter oder /s) zurück	<code>.</code>	abcdefghijklmn
<code>\d</code>	Gib mir alle Zahlen (Digits) zurück, die du findest	<code>\d</code>	abcd 123 efgh
<code>\D</code>	Gib mir alle Nicht-Zahlen (Digits) zurück, die du findest	<code>\D</code>	abcd 123 efgh
<code>\w</code>	Gib mir alle Wort-Charakter (a-z,A-Z,0-9) zurück	<code>\w</code>	abcd 123 efgh
<code>\W</code>	Gib mir alle nicht-Wort-Charakter zurück (!, ", , \$, %, &, /, (,), =, ?, ` , , ...)	<code>\W</code>	123 ?! 321
<code>\s</code>	Gib mir alle Leerzeichen (= whiteSpaces), ggf. Zeilenumbrüche, Enden der Zeilen, usw. zurück („ „)	<code>\s</code>	123 3434
<code>\S</code>	Gib mir alle nicht-Leerzeichen, oder ggf. Nicht- Zeilenumbrüche, -Enden der Zeilen, usw. zurück	<code>\S</code>	123 3434

Merke: Single-Meta-Charakter beziehen sich nur auf einzelne Charakter

Quantifiers

<code>?</code>	= Schaue die den vorherigen Charakter an, dieser kann einmal vorkommen, muss aber nicht vorkommen	<code>(er)?</code>	Rechererere er
<code>*</code>	(Greedy) Schau dir den vorherigen Charakter an, dieser kann beliebig (0-x) oft vorkommen	<code>\w.*</code>	Rechererer er robert licem
<code>*?</code>	(Ungreedy)	<code>\w.*?</code>	Rechererer er robert licem
<code>+</code>	Schau dir den vorherigen Charakter an, dieser muss einmal vorkommen, darf aber auch öfters vorkommen (1-x)	<code>r+</code>	Rechererer er robert licem
<code>{x,y}</code>	Schau dir den vorherigen Charakter an, dieser muss mindestens x mal vorkommen, und darf höchstens y vorkommen.	<code>r{1,2}</code>	Rechererer er robert licem



Quantifiers (cont)

	Oder-Operator	a x	abcdef...xyz
--	---------------	-----	--------------

Die Quantifier-Operatoren (+, *) sind greedy (= gierig). D.h. diese möchten immer das längstmögliche Charakterset zurückgeben.

Wenn man aber nun nicht an dem/oder nur an dem Inhalt interessiert ist, muss man diese Operatoren lazy machen. siehe hierzu Foliensatz

Ranges

(...)	Gruppe an Charaktern, die gesucht werden sollen	(kuchen)+	Apfel kuchen , Birnen kuchen , kuche, Kuchen
[...]	Verkürzte "Oder"-Schreibweise. Gib in der Range von ... - ... alle Charakter an.	[uch]	Apfel kuchen , Birnen kuchen , kuche, Kuchen
[^...]	Verkürzte "Oder"-Schreibweise. Gib in der Range von ... - ... alle Charakter außer diese an.	[^uch]	Apfelkuchen , Birkenkuchen , kuche, Kuchen
[...-...]	Gib alle Charkter in diesem Bereich an. z.B. 0-9A-Za-z	[a-z]	Apfelkuchen , Birkenkuchen , kuche, Kuchen

Assertion/Behauptungen

?(a)	Sobald du a gefunden hast, schaue dir die Charakter davor an. (Lookahead assertion)	
?!(a)		
?<=(a)	= Sobald du a gefunden hast, schaue dir die Charakter danach an. (Lookbehind assertion)	
?!<=(a)		
?<=(a) (...) ?	Suche nach a, sobald du dieses gefunden hast schaue dir alle Charakter an, bis du das nächste a findest.	(?<=(a)(\W \w)+(?=a)) a Kuchen leben a

Escapen von Meta-Charakter

^	[	.
\$	(	)
*	+	?
	<	>
\	{	

Wenn man nach Meta-Charakter sucht muss man diese **immer** Escapen d.h. ein \ davor setzen



By **ReneD**
cheatography.com/reneD/

Not published yet.
Last updated 16th October, 2018.
Page 2 of 2.

Sponsored by **Readable.com**
Measure your website readability!
<https://readable.com>