

Anchors/Ankers/Positions-Metacharakter

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

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

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

Quantifiers

?	= Schau die den vorherigen Charakter an, dieser kann einmal vorkommen, muss aber nicht vorkommen	(er)?	Rechererere er
*	(Greedy) Schau dir den vorherigen Charakter an, dieser kann beliebig (0-x) oft vorkommen	\w.*	Rechererer er robert licem
?	(Ungreedy)	\w.?	Rechererer er robert licem
+	Schau dir den vorherigen Charakter an, dieser muss einmal vorkommen, darf aber auch öfters vorkommen (1-x)	r+	Rechererer er robert licem
{x,y}	Schau dir den vorherigen Charakter an, dieser muss mindestens x mal vorkommen, und darf höchstens y vorkommen.	r{1,2}	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 , Birnenkuchen , kuche , Kuchen
[...-...]	Gib alle Charkter in diesem Bereich an. z.B. 0-9A-Za-z	[a-z]	Apfelkuchen , Birnenkuchen , 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 **ApolloPad.com**
 Everyone has a novel in them. Finish Yours!
<https://apollopad.com>