Reguläre Ausdrücke
- eine Einführung -

Andreas Romeyke

Leipzig, Oktober/November 2002

Diese Präsentation ist eine kurze Einführung in die Welt der regulären Ausdrücke mit Schwerpunkt Perl.
Die Präsentation dauert knapp 45 min und soll Anregung sein, sich mehr mit Perl, Makroprozessing und Unix zu beschäftigen.

Inhalt

1 Was sind Reguläre Ausdrücke?

Reguläre Ausdrücke sind:

- Suchmuster
- Vorlagen, Schablonen
- Textmanipulationsfilter
- endliche Automaten
- LEGO^{TM}*-Steine

2 Basisbausteine

- patterns (oder auch Muster)
- quantifiers (oder auch Wiederholungssymbole)
- anchors (oder auch Anker)

- buffers (oder auch Puffer)
- classificators (oder auch Symbolklassen zB. Ziffern)
- area controller (bzw. Symbole um Bedingungen abzuklären)

Ein sehr einfacher Ausdruck ist:

“stop” → “non***stop***”

Hier nochmal das Beispiel mit den Dateien:

		“test.bat”
“*.exe”	→	“***test.exe***”
		“test.png”

Streng genommen müsste der Reguläre Ausdruck so lauten:

		“test.bat”
“[^\.]+\.exe”	→	“***test.exe***”
		“test.png”

Was sind denn nun Pattern? Nun ein Pattern kann:

- ein Zeichen oder ein Symbol
- eine Gruppe von Symbolen (String)
- eine Klasse von Symbolen

sein. Pattern kann man auch Unterausdruck nennen.

Eine Gruppe von Symbolen wird in runden Klammern eingeschlossen:

"$(symbol)$" → gruppiert zum String "symbol"

Wohingegen eine Klasse von Symbolen in eckigen Klammern eingeschlossen wird:

"$[symbol]$" → Trifft auf 's', 'y', 'm', 'b','o' oder 'l' zu

Um eine Klasse zu negieren, benutzt man das Zirkumflex:

"$[\hat{}symbol]$" $\rightarrow$ Trifft nicht auf 's', 'y', 'm', 'b','o' oder 'l' zu

Eine Gruppe von Pattern oder Ausdrücken wird auch in Klammern eingeschlossen:

"$(sym\,[bol])$" $\rightarrow$ Trifft auf Ausdrücke "symb","symo" oder "syml" zu.

Daneben gibt es noch die Quantifiers:

* kein Unterausdruck oder einer und mehrere Unterausdrücke

+ einer und mehrere Unterausdrücke

? kein Unterausdruck oder nur einer

$\{2\}$ nur zwei Unterausdrücke (nich mehr, nicht weniger)

$\{2,\}$ zwei und mehr Unterausdrücke

$\{2,5\}$ zwei bis zu fünf Unterausdrücke

$\{,5\}$ keiner oder eins bis fünf Unterausdrücke

Auch hierzu wieder ein Beispiel:

"12?" → "*011*" oder "*012*", nicht "*122*"

Was bisher noch fehlte sind die Anker:

^ markiert den Beginn des Strings

$ markiert das Ende des Strings

"^script" →"***script****um*", nicht "*postscript*"

"tum$" →"*scrip****tum***", nicht "*tumbre*"

"^script" →"***script***", nicht "*postscriptum*"

"script" →"*post****script****um*"

3 Wichtige Regeln!

1. Der Treffer gewinnt, der am frühesten beginnt
2. Manche Quantifiers sind gefräßig

   Ein Beispiel:
   "[0-9]+" matcht auf "September␣**2001**"

Manche Quantifiers sind gefräßig

Perl bietet den *'nicht gierig'*-Operator.

"`S.*?e`" matcht auf
"**Se**ptember␣2001"

4 Die geheimnisvolle Maschine

- NFA
- DFA
- POSIX-NFA

5 Geschwindigkeit ist keine Hexerei

- DFA besser als POSIX-NFA besser als NFA...
- Wissen ist Macht!

  "`Sch[:alpha:]+`" statt "`Sch.*`" um nach "Schneider" zu suchen
- keine Alternativen!
- Kampf der Gefräßigkeit!
- Anchor halten das Schiff ruhig!
- Negativ ist Positiv!

6 Auf zu anderen Ufern

. . . oder Notation von RegEx in anderen Programmen

	Theorie	Perl	Grep	Shell	SQL
leere Menge	$\oslash$				
einzelnes Zeichen	a	a	a	a	a
Konkatenation	ab	ab	ab	ab	ab
Alternative	a | b	a | b	a | b		
Klammer	(. . .)	(. . .)	\(. . . \)		
i-te Klammer					
Zeichen aus Menge	a | b | c	[abc]			
Zeichen aus Nichtmenge		[^abc]	[^abc]	[^abc]	
beliebiger Char		.	.	_	?
beliebiger String		.*	.*	*	%
Kleenescher Abschluß	a*	a*	a*		
optionaler Ausdruck	(| x)	a?	a\?		
1.. fache Wiederholung	xx*	x+	x+		

7 Was Perls RegExer noch so bietet

- positive Look-aheads `?=`
- negative Look-aheads `?!`
- positive Look-behinds `?<=`
- negative Look-behinds `?<!`
- Rückbezüge auf Klammern `\1` ... `\n`

- Ersatzsymbole für
  - Wörter `\w`
  - Wortgrenzen `\b`
  - Whitespace `\s`
  - Digits `\d`
- Abkürzungen für POSIX-Zeichenklassen `[:alpha:]`
- Identifier für nicht gieriges Verhalten `a*?b`

8 Literatur – Wie geht es weiter?

- Friedl, Jeffrey E. F.:Mastering Regular Expressions; O'Reilly & Associates, Inc; 1998
- Schwartz, Randal L.; Christiansen, Tom:Learning Perl, Second Edition; O'Reilly & Associates, Inc;1997
- man perlfaq, man perlre
- Die Perl-Seite `http://perl.com`
- GAOS e.V. `http://gaos.org/`
- `http://andreas-romeyke.de`
- `http://txt2regex.sourceforge.net`