Regex

Regular Expressions wie sie im Text-Editor Notepad++ verwendet werden. Andere Editoren oder Programmiersprachen verwenden eventuell eine geringfügig abweichende Syntax.

Übereinstimmung einzelner Zeichen

. \c

Der Punkt stimmt bei der Suche mit jedem Zeichen einer Zeile überein. Aktiviert man „findet \r\n“ kann man auch über Zeilengrenzen hinweg suchen.

\X

Findet ein einzelnes Zeichen gefolgt von beliebig vielen kombinierbarer Zeichen. Das trifft zu bei Unikode-Zeichen, die aus einem diakritischen Zeichen und dem eigentlichen Buchstaben zusammengesetzt sind.

\r

Der Backlash vor einem Zeichen erlaubt die Verwendung dieses Zeichens ohne dessen Bedeutung als Regex-Zeichen. Beispielsweise ist \[eine öffnende eckige Klammer und nicht der Beginn einer Zeichenklasse.

Nicht ASCII-Zeichen

\xnn

Einzelnes Zeichen mit dem Kode nn. Was das bedeutet, hängt von der Kodierung ab. Beispielsweise findet \xE9 ein é oder ein θ je nachdem welche Kode-Seite im Dokument verwendet wird.

\x{nnnn}

Findet ein 16-Bit-Unicode-Zeichen. In ANSI-Dokumenten ist dieses Konstrukt nicht erlaubt.

\Onnn

Findet ein Zeichen im Oktal-Kode

[[.Sortierreihenfolge.]]

Sortierreihenfolge festlegen. Beispielsweise wird die Zeichenkombination "ch" im Spanischen als einzelnes Zeichen betrachtet, obwohl sie aus zwei Zeichen besteht. Das Zeichen ist dann [[.ch.]]. Funktioniert auch mit symbolischen Namen wie zum Beispiel [[.BEL.]] mit dem Zeichenkode 0x07.

Steuerzeichen

\a

Steuerzeichen BEL 0x07 *(alarm)*.

\b

Steuerzeichen BS 0x08 *(backspace)*. Nur innerhalb einer Zeichenklassendefinition erlaubt, ansonsten hat es die Bedeutung einer „Wortgrenze“.

\e

Steuerzeichen ESC 0x1B *(escape)*

\f

Steuerzeichen FF 0x0C *(form feed)*.

\n

Steuerzeichen LF 0x0A *(line feed)*. Zeilenende in UNIX-Systemen.

\r

Steuerzeichen CR 0x0D *(carriage return)*. Teil der Zeilenende-Sequenz CR-LF in DOS/Windows und auch in Mac 9 und früher. Spätere Mac-OS benutzen \n.

\R

Irgend ein Zeilenendezeichen.

\t

Steuerzeichen TAB 0x09 *(tab, or hard tab, horizontal tab)*.

\C<character>

Das Steuerzeichen, das durch Isolierung der 6 niederwertigen Bit entsteht. Beispielsweise stehen \C1, \CA and \Ca für das Steuerzeichen SOH 0x01.

Zeichenklasse

[...]

Definiert eine Zeichenklasse. Beispielsweise bedeutet [abc] irgendeines der Zeichen a, b oder c. Man kann mit einem Bindestrich auch ganze Bereiche definieren, zum Beispiel [a-z] für irgendeinen Kleinbuchstaben. Man kann auch die Sortierreihenfolge in diesem Rahmen verwenden, wie [[.ch.]-[.ll.]]

[^...]

Komplementiert die Zeichenklasse, meint daher alle Zeichen, die in der Zeichenklassendefinition nicht enthalten sind. Beispielsweise bedeutet [^A-Za-z] jedes Zeichen außerhalb der Alpha-Zeichen. Vorsicht diese diesen Komplementär-Zeichenklassen, denn sie finden Zeichen immer auch über Zeilengrenzen hinweg. Will man das verhindern, muss man die Zeilenendezeichen in die Zeichenklasse aufnahmen. Beispiel: [^ABC\r\n].

Vordefinierte Zeichenklassen

[[:alnum:]]

ASCII-Zeichen und Zahlen

[[:alpha:]]

ASCII-Zeichen

[[:blank:]]

Zwischenraum aber kein Zeilenende

[[:cntrl:]]

Steuerzeichen

[[:d:]] [[:digit:]]

Dezimalzahl

[[:graph:]]

Grafikzeichen

[[:l:]] [[:lower:]]

Kleinbuchstabe

[[:print:]]

Druckbares Zeichen

[[:punct:]]

Sonderzeichen: , " ' ? ! ; : # $ % & ( ) * + - / < > = @ [ ] \ ^ _ { } | ~

[[:s:]] [[:space:]]

Zwischenraum

[[:u:]] [[:upper:]]

Großbuchstabe

[[:unicode:]]

Irgendein Zeichen mit einem Kodewert über 255

[[:w:]] [[:word:]]

Wortzeichen

[[:xdigit:]]

Hexadezimal-Zeichen

[[:\pshort name:]] [[\p{name}:]]

Identisch mit [[:name:]]. Beispielsweise stehen \pd und \p{digit} für eine Ziffer \d.

[[:Pshort name:]] [[:\P{name}:]]

Identisch mit [^[:name:]] (gehört nicht zu der Zeichenklasse).

\d

Ziffer 0-9 range, entspricht [[:digit:]].

\D

Keine Ziffer, entspricht [^[:digit]].

\l

Kleinbuchstabe, entspricht [a-z] oder [[:lower:]]. Wenn die Option "Match case" ausgeschaltet ist, entspricht das einem Wortzeichen \w

\L

Kein Kleinbuchstabe

\u

Großbuchstabe, entspricht **[[:upper:]].**

\U

Kein Großbuchstabe

\w

Wortzeichen. Entweder ein Buchstabe, eine Zahl oder ein Unterstrich. Entspricht [[:word:]].

\W

Kein Wortzeichen. Entspricht [[:alnum:]] mit der Ergänzung des Unterstrichs.

\s

Abstandzeichen; enthält SP, TAB und EOL. Entspricht [[:space:]].

\S

Kein Abstandzeichen.

\h

Horizontales Abstandzeichen. Enthält SP, TAB und LF.

\H

Kein Horizontales Abstandzeichen

\v

Vertikales Abstandzeichen. Enthält die Steuerzeichen VT, FF und CR = 0x0B (vertical tab), 0x0D (carriage return) und 0x0C (form feed).

\V

Kein Vertikales Abstandzeichen.

[[=primary key=]]

Alle Zeichen, die sich von „primary key“ durch die Schreibweise (groß/klein), den Akzent oder eine ähnliche Variation unterscheiden. Beispiel: [[=a=]] findet irgendeines der Zeichen: a, À, Á, Â, Ã, Ä, Å, A, à, á, â, ã, ä and å.

Vervielfachende Operatoren

+

Findet eines oder mehrere des vorigen Zeichens, so viele wie möglich. Beispiel: Sa+m findet Sam, Saam, Saaam.... [aeiou]+ findet aufeinander folgende Selbstlaute.

*****

Findet kein oder mehrere des vorigen Zeichens, so viele wie möglich. Beispiel: Sa*m findet Sm, Sam, Saam...

?

Findet keines oder eines des vorigen Zeichens. Beispiel: Sa?m findet Sm and Sam, aber nicht Saam.

+?

Findet eines oder mehrere des vorigen Zeichens, aber so wenig wie möglich. Beispiel: Sa+?m findet Sam, Saam, Saaam.... [aeiou]+ findet aufeinander folgende Selbstlaute.

***?**

Findet kein oder mehrere des vorigen Zeichens, so wenig wie möglich.

Beispiel: m.*?o findet im Text “margin-bottom: 0;” margin-bo, während m.*o findet margin-botto.

{n}

Findet n Elemente.

{n,}

Findet n oder mehr Elemente, so viele wie möglich.

{n,}?

Findet n oder mehr Elemente, so wenig wie möglich.

{m,n}

Findet m bis n Elemente, so viele wie möglich.

{m,n}?

Findet m bis n Elemente, so wenig wie möglich.

Anker

^

Findet den Zeilenbeginn (außer innerhalb einer Zeichenklasse [...].

$

Findet das Zeilenende.

\<

Findet einen Wortanfang.

\>

Findet ein Wortende.

\b

Findet entweder Wortanfang oder Wortende.

\B

Findet weder Wortanfang noch Wortende.

\A \'

Der Beginn des Suchstrings.

\z \`

Das Ende des Suchstrings.

Gruppen

(.)

Klammern markieren eine Gruppe, einen Teil des Suchbegriffs. Der so markierte Teilstring kann bei Ersetzungsvorgängen als Variable verwendet werden. Gruppen können verschachtelt werden.

(?<some name>.) (?'some name'.)
(?(some name) .)

Benennung von Gruppen.

\gn \g{n}

Die n-te Gruppe (d.h. eingeklammerte Gruppe). Die zweite Form hat Vorteile, weil n auch größer als 9 werden kann. Wenn n' negative ist, warden die Gruppen nach rückwärts gezählt, sodass \g-2 die vorletze gefundene Gruppe ist.

\g{something} \k<something>

Findet die mit **something** benannte Gruppe.

\1 \2...

Ersetzung: **\1** findet die ersten angegeben Gruppe. Beispiel**: ([Cc][Aa][Ss][Ee])** findet alle denkbaren Schreibweisen von *Case*. Ein Regex-Ausdruck kann mehrere Untergruppen **\2**, **\3** usw. haben.

Verbesserung der Lesbarkeit

(:.)

Eine Gruppierung, zur Erleichterung der Lesbarkeit.

(?#...)

Kommentar. Comments. Eventuell gemeinsam mit dem x-Modifizierer ein guter Ansatz zur Erleichterung der Lesbarkeit.

Änderungen der Suchregeln

\Q

Behandelt Zeichen wörtlich bis zum Auftreten von \E

\E

Beendet die wörtliche Betriebsart. Beispiel: "\Q*+\Ea+" findet "*+aaaa".

(?:flags-not-flags ...) (?:flags-not-flags:...)

Definiert Bedingungen (setzt oder löscht Flags) für eine Suche:
i schreibweisenunabhägnig (default: off)
m **^** und **$** finden eingebettete Zeilenunbrüche.
s Punkt findet Zeilenumbruch.
x ignoriert Spaces sofern sie nicht mit Backslash eingeleitet werden (default: off)

Alternierungskonstrukte

|

Findet einen von mehreren optionalen Suchbegriffen. Beispiel: eins|zwei|drei findet entweder "eins", "zwei" oder "drei". Der Vergleich findet von links nach rechts statt. Mit (?:) kann auch ein leerer String als Alternative angegeben werden.

\K

Löscht den bereits gefundenen Text exakt an diesem Punkt. Beispiel: Suche nach *„Apfel\Kbaum“* findet zwar *„Apfelbaum“* schränkt aber das Gesuchte auf *„baum“* ein.

Ersetzungen

\a,\e,\f,\n,\r,\t,\v

Setzt das entsprechende Steuerzeichen ein, also BEL, ESC, FF, LF, CR, TAB und VT.

\Ccharacter" \xnn \x{nnnn}

Das Steuerzeichen, das durch Isolierung der 6 niederwertigen Bit entsteht, wird eingefügt. Beispielsweise stehen **\C1**, **\CA** and **\Ca** für das Steuerzeichen SOH 0x01.

nnnn erfordert Unicode Kodierung.

\l

Nächstes Zeichen als Kleinbuchstaben ausgeben.

\L

Alle nächsten Zeichen werden bis zum Auftreten eines \E als Kleinbuchstaben ausgeben.

\u

Nächstes Zeichen als Großbuchstaben ausgeben.

\U

Alle nächsten Zeichen werden bis zum Auftreten eines \E als Großbuchstaben ausgeben.

\E

Beendet die erzwungene Schreibeweise, die durch **\L** oder **\U** eingeleitet wurden.

$& $MATCH ${^MATCH}

Der gesamte gefundene Text.

$` $PREMATCH ${^PREMATCH}

Der Text zwischen der vorigen und der aktuellen Übereinstimmung oder . wenn es die erste Übereinstimmung ist, der Text vor der Übereinstimmung.

$" $POSTMATCH ${$POSTMATCH}

Der Text nach der aktuellen Übereinstimmung.

$LAST_SUBMATCH_RESULT $^N

Der Text, mit der der letzte Suchbegriff übereingestimmt hat.

$+ $LAST_PAREN_MATCH

Der Text, mit der der letzte Suchbegriff im Suchmuster übereingestimmt hat.

$$

Das Dollarzeichen wird zurückgegeben.

$n ${n} \n

Der Text, der mit dem n—ten Suchbegriff übereingestimmt hat..

$+{name}

Der Text, der mit dem mit **name** benannten Suchbegriff übereingestimmt hat..

Download

Webversion

http://d.pcnews.at/ins/pcn/139/002500/

- RegularExpressions.docx