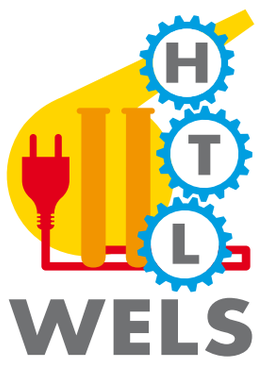
REGULÄRE AUSDRÜCKE

DI Thomas Helml
SEW 3



WAS SIND REGULÄRE AUSDRÜCKE?

- Ein regulärerer Ausdruck (regular expressions, regex) definiert ein Muster (pattern) zur der Suche einer Zeichenkette.
- das Muster kann mehrfach verwendet werden
- Somit kann z.B. die Eingabe eines Benutzer auf Richtigkeit geprüft werden
- Beispiele:
  - Schulnote
  - Postleitzahl
  - E-Mail-Adresse
  - Bestellnummern

REGELN

No	Die regulären Ausdruck	Die Bezeichnung
1	`.`	Irgendeinem Zeichen entsprechen (match)
2	`^regex`	Die reguläre Ausdruck muss bei der Startspunkt der Linie entsprechen
3	`regex$`	Die reguläre Ausdruck muss bei der Ende der Linie entsprechen
4	`[abc]`	Die Definition einstellen, die a oder b oder c entsprechen.
5	`[abc][vz]`	Die Definition einstellen, die a oder b oder c entsprechen, danach kommt v oder z.
6	`[^abc]`	Wenn das Zeichen ^ als den ersten Charakter in eckige Klammer auftritt, verneint es das Modell. Das kann allen Zeicher außer a oder b oder c entsprechen
7	`[a-d1-7]`	Der Raum: eine Buchstabe zwischen a und d und die Zahlen vom 1 bis 7 entsprechen
8	`X|Z`	X oder Z finden
9	`XZ`	X und folgend Z finden
10	`$`	Die Zeileende prüfen

11	`\d`	Irgendeines Digit, für `[0-9]` `v` verküzen
12	`\D`	Die nicht-Digit, für `[^0-9]` verküzen
13	`\s`	Ein Leerzeichen, für `[ \t\n\x0b\r\f]` verküzen
14	`\S`	Ein Nicht-Leerzeichen, für `[^\s]` verküzen
15	`\w`	Eine Buchstabe, für `[a-zA-Z_0-9]` verküzen
16	`\W`	Eine Nicht-Buchstabe, für `[^\w]` verküzen
17	`\S+`	Einige Nicht Leerzeichen (ein oder mehr)
18	`\b`	Das zeichen vom a-z oder A-Z oder 0-9 oder _, für `[a-zA-Z0-9_]` verküzen.

19	*	Zero oder mehrmals auftreten, für {0,} verküzen
20	+	Eins oder mehrmals auftreten, für {1,} verküzen
21	?	Zero oder einsmal auftreten ? für {0,1}. verküzen
22	{X}	X Mals auftreten {}
23	{X,Y}	Vom X zum Y Mals auftreten
24	*?	* bedeutet das Auftritt in zero oder mehrmals,? am hinter bedeutet die Suche nach der kleinste Übereinstimmung

EIN-ELEMENTIGE REGULÄRE AUSDRÜCKE

- Bsp. Schulnote in Österreich:

```
[12345]
```

- In eckigen Klammern folgt eine Auflistung von Zeichen, die erlaubt sind, der gesamte geklammerte Ausdruck steht für ein Zeichen
- Wenn die Zahlen von 1 bis 5 aufeinander folgen, gibt es eine Abkürzung

```
[1-5]
```

EIN-ELEMENTIGE REGULÄRE AUSDRÜCKE

- Bsp. Bahnhof mit 9 Gleisen:

  `[1-9]`

- Bsp. Gleis 4 ist gesperrt und somit nicht zulässig:

  `[1-35-9]` oder auch `[1-3  5-9]`

MEHRELEMENTIGE REGULÄRE AUSDRÜCKE

- Mehrere rechteckige Klammern => mehrere Zeichen
- das Muster wird von links nach rechts mit dem Ausdruck verglichen
- Bsp. Bahnsteige 1-9 haben noch jeweils Abschnitt "a" und „b"
- Gültige Werte sind also 1a, 1b, 2a, ….

```
[1-9][ab]
```

- auch zusammenhängende Buchstabenbereiche können abgekürzt werden:

```
[1-9][a-d]
```

MEHRELEMENTIGE REGULÄRE AUSDRÜCKE

- reguläre Ausdrücke sind case-sensitive
- voriges Beispiel mit Groß-/Kleinschreibung:

```
[1-9][a-dA-D]
```

OPTIONEN

- Fragezeichen hinter einem Element:
  - das vorhergehende Element *kann* vorkommen, muss aber nicht

- Bsp:
  - Hausnummern bestehen aus 1-3 Ziffern, gefolgt von einem Zeichen (a-z). Eine Ziffer ist notwendig, der Rest optional:

```
[1-9][0-9]?[0-9]?[a-z]?
```

WIEDERHOLUNGEN

- bei x Ziffern wird das sehr unübersichtlich
- daher Kurzschreibweise:
  - a{1,3}h
  - Gültige Eingaben: "ah" oder „aaah"
  - Angabe in Klammern steht für {minimale, maximale} Anzahl Zeichen

WIEDERHOLUNGEN

- Bsp. Hausnummern:
  - [1-9][0-9]{0,2}[a-z]?
- Bsp. lange Hausnummern:
  - [1-9][0-9]{1,4}[a-z]?
    - fünfstelliger Bereich, aber mindestens zweistellig

OPTIONEN

- Beispiel: 5-stellige Zahl
  - [0-9]{5}
- Geschwungene Klammer mit nur einer Zahl x:
  - x-malige Wiederholung des davor stehenden Ausdrucks
- "mindestens" 3-stellige Ziffer
  - [0-9]{3,}

BELIEBIGE WIEDERHOLUNGEN

- Telefonnummern sollen von diesem Format sein dürfen:
  - 0651/55541-36
  - 0049 160 555678
  - 0180.23.555.63
- neben Zahlen dürfen auch Bindestriche, Querstriche, Leerzeichen und Punkte vorkommen
- *Ein* gültiges Element wäre also
  - [0-9/. –]
  - (Also: Dieses *eine* Zeichen besteht aus einer Zahl zwischen 0 und 9 *oder* einem Slash *oder* einem Punkt *oder* einem Leerzeichen *oder* einem Minus.)

BELIEBIGE WIEDERHOLUNGEN

- [0-9/. –]
- Achtung! Der Bindestrich kommt 2x vor!
  - Zahlenbereich: 0-9
  - als Zeichen -> dieses muss mit Backslash „maskiert“ werden
- [0-9/. \–]

BELIEBIGE WIEDERHOLUNGEN

- Plus +:
  - Zeichen kommt 1..n mal
  - `[0-9/. \—]+`

- Stern *:
  - Zeichen kommt 0..n mal
  - `[0-9/. \—]*`

PLATZHALTER

- Beispiel Bibliothek:
  - Suche nach Autor: Vorname und Nachname ist bekannt, aber nicht ob er einen 2. Vornamen hat
- Nach Vornamen können **beliebige Zeichen** kommen und danach dann erst der Nachname.
  - `Markus .*Ostermann`

- Erklärung
  - `Punkt` = beliebiges Zeichen
  - `Stern` = 0..n mal beliebiges Zeichen

ZEICHENKLASSEN NEGIEREN

- Negieren/Ausschließen: ^

- zweiter Vornamen des Autors enthält garantiert kein q und kein z

- `Markus [^qz]+ Ostermann`

- verlangt nach einem weiteren Namen (deswegen das Plus) und lässt dazu ein beliebiges Zeichen zu, das **nicht** q oder z ist.

KLAMMERN

- Klammern: fasst längere Ausdrücke zu einem Element zusammen

- `Markus (Müller )?Ostermann`
  - "Markus Müller Ostermann" oder "Markus Ostermann"
    - Fragezeichen bezieht sich auf Müller plus Leerzeichen, 0..1 mal

- Beispiel
  - Ba(na)*ne

ALTERNATIVEN

- Alternativen in einem Teilausdruck:
  - `Das Wetter ist (toll|richtig schlecht)`

VERSCHACHTELUNGEN

- Klammern können geschachtelt werden:
  - `(VW (Golf|Polo)|Fiat (Punto|Panda))`
  - "VW Golf", "VW Polo", "Fiat Punto" oder "Fiat Panda"
- Wiederholung von Alternativen:
  - `(10|01)+`
  - Folge aus Nullen und Einsen, in der maximal 2 Nullen oder Einsen aufeinander folgen.

REGULÄRE AUSDRÜCKE PRÜFEN

- http://www.zytrax.com/tech/web/regex.htm
- http://www.regexe.de
-

REGEX IN JAVA

- siehe Buch „Java 7 - Fortgeschrittene Programmierung“, Kapitel 2, Seite 10ff