



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

Webbasierte Anwendungen

Reguläre Ausdrücke

Prof. Dr. Ludger Martin

Gliederung



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Einführung
- PCRE
- Unterschiede POSIX zu PCRE
- Programmierung



- Engl: regular expression
- (komplexe) Suchmuster
- Zwei Standards
 - PCRE (Perl Compatible Regular Expressions)
 - POSIX (Portable Operating System Interface for uniX)
 - Basic Regular Expression
 - Extended Regular Expressions
- JavaScript – PCRE
- HTML5 – PCRE
- SQL (MariaDB ab Version 10.0.8) – PCRE
- SQL (MySQL, Oracle) – POSIX Standard
- PHP – PCRE und POSIX (ab PHP 5.3.0 veraltet)

PCRE



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Reguläre Ausdrücke werden zwischen **/** notiert:

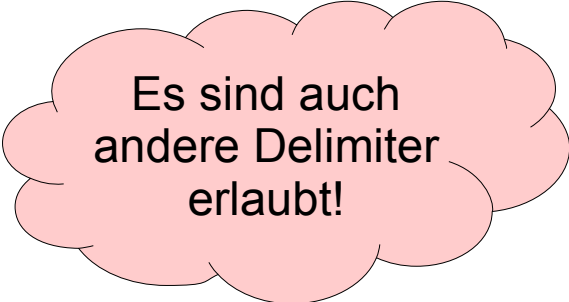
/Regulärer Ausdruck**/**

- Beispiel:

– Begriffe: "Maus", "Braut", "Haus"

– Ausdruck: /aus/

– Treffer: "Maus", ~~"Braut"~~, "Haus"



Es sind auch
andere Delimiter
erlaubt!



- Metazeichen (Wildcard):
 - Suchen von variablen Zeichen
 - Ein `.` steht für *ein* beliebiges Zeichen
 - Beispiel:
 - Begriffe: `"Fest"`
`"Test"`
`"Wertebereich"`
`"Robert"`
 - Ausdruck: `/e.t/`
 - Treffer: `"Fest"`
`"Test"`
`"Wertebereich"`
`"Robert"`
 - Soll ein `.` gefunden werden muss `\.` geschrieben werden
 - Für `\` ist `\\` zu nutzen



- Quantifizierer:
 - Wie oft soll ein Zeichen gefunden werden?
 - **+**: mindestens einmal
 - **?**: keinmal oder einmal
 - *****: keinmal, einmal oder mehrmals
 - Beispiel:
 - Begriffe: "Haus"
"Hans"
"Hannes geht nach Hause"
 - Ausdruck: `/a.+s/`
 - Treffer: "H**au**s"
"H**an**s"
"H**anne**s geht nach H**au**s"

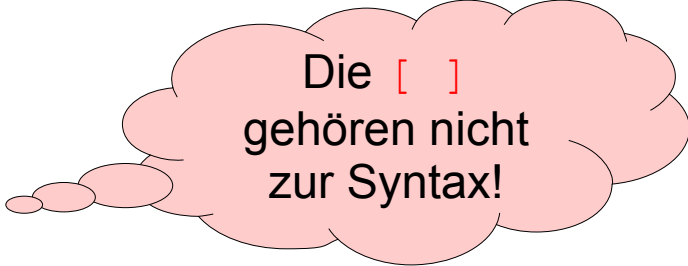
- Quantifizierer:

- Häufigkeitsangabe

`{<min>[, [<max>]]}`

- Beispiele:

- `/x{2,4}/:` mindestens 2, maximal 4 x
 - `/x{2,}/:` mindestens 2 x
 - `/x{2}/:` genau 2 x



Die `[]`
gehören nicht
zur Syntax!



- Gruppierung:
 - Mit `()` können Zeichen gruppiert werden
 - Beispiel:

`/(aua)+/`

erkennt alle Zeichenreihen, die die Buchstaben `aua` mindestens einmal enthalten:

"Der Patient ruft `auaaua!`"

- Alternativen:
 - mit `|` kann eine Alternative angegeben werden
 - Beispiel: `/(htm)| (html)/`

- Zeichenklassen:
 - Zeichen, die wahlweise gefunden werden sollen, werden in `[]` notiert:
`/[abcdeABCDE]/`
 - Verkürzte Schreibweise:
`/[a-eA-E]/`
 - Zeichen, die nicht gefunden werden dürfen, werden in `[^]` notiert:
`/[^0-9]/`
- Abkürzungen:
 - `\d` findet Ziffern (`\D` keine Ziffern)
 - `\w` findet Wortzeichen (`\W` keine Wortzeichen)
`[a-zA-Z0-9_]`
 - `\s` findet Whitespace (`\S` kein Whitespace)

- Spezielle Suchabkürzungen:

- `^` – Anfang einer Zeichenreihe oder Zeile
- `$` – Ende einer Zeichenreihe oder Zeile
- `\b` – Wortgrenze (`\B` keine Wortgrenze)
- `\A` – Anfang von Zeichenreihe
- `\Z` – Ende einer Zeichenreihe oder Zeilenende
- `\z` – Ende einer Zeichenreihe

Nicht in JavaScript nutzbar

- Beispiel:

`/^\w+/` – Erstes Wort in Zeichenreihe oder Zeile



- Modifizierer für reguläre Ausdrücke:
 - Modifizierer werden nach dem `/ . . . /` angegeben:
`/aua/i` findet auch `Aua` und `AUA`
 - `i` – ignoriert Groß- und Kleinschreibung
 - `u` – ein Ausdruck wird als UTF-8 Zeichenreihe behandelt
 - `s` – `.` beinhaltet auch den Zeilenumbruch

Unterschiede POSIX zu PCRE



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Pattern werden in POSIX nicht mit Delimitern eingeschlossen.
- POSIX hat eine spezielle Funktion `[. .]` für case-insensitive.
z.B. `[.Ch .]`
- POSIX versucht die von links aus längste Zeichenkette zu finden, PCRE findet die erste valide Zeichenkette.
- Zeichenklassen werden in POSIX mit `[: :]` definiert.
z.B. `[:digit:]`
- Beispiel
`[:alpha:] [[:alpha:] [:digit:]] *`



- JavaScript

- Finden innerhalb einer Zeichenreihe. Gibt die erste Fundstelle zurück.

```
aString.search(regex);
```

Beispiel

```
var s = "JavaScript ist eine Programmiersprache.";
s.search(/script/i);
// Rückgabe 4
```

- Ersetzen innerhalb einer Zeichenreihe. Liefert die neue Zeichenreihe zurück.

```
aString.replace(regex, replacement);
```

Beispiel

```
var s = "JavaScript ist eine Programmiersprache.";
s.replace(/javascript/i, "JS");
// Rückgabe "JS ist eine Programmiersprache."
```



- HTML5
 - Validierung von Eingaben

```
<input type="text" name="text" pattern="[a-z]+" />
```

ABC

Bitte benutzen Sie das korrekte
Format



- MySQL

- Suche nach regulärem Ausdruck `muster` in `spalte`
- Erlaubt in `WHERE`-Teil
- `spalte REGEXP muster`
- Beispiel

```
SELECT * FROM medienartikel WHERE titel REGEXP '[:,digit:]]+'
```

a_nr	titel	jahr
0001-E	LaTeX in 21 Tagen	2004
0016-D	Star Trek - 40 Years	2006



- PHP

- Auf passende reguläre Ausdrücke prüfen:

```
int preg_match(string regexp, string str)
```

Beispiel:

```
if (preg_match("/aus/", "Haus")) {  
    ...  
}
```

- Alle Stellen finden:

```
int preg_match_all (string regexp, string str, array matches)
```

Die gefundenen Stellen werden in einem Array zurückgegeben

- Mit regulärem Ausdruck ersetzen:

```
mixed preg_replace(mixed regexp, mixed replacement, mixed subject)
```

Als Parameter können entweder Zeichenreihen oder Arrays übergeben werden



- Flanagan, D.: JavaScript – Das umfassende Referenzwerk, Auflage 6, 1. korr. Nachdruck, O'Reilly, 2014
- Lubkowitz, M: Webseiten programmieren und gestalten, 2. Auflage, Galileo Press, 2006
- The Open Group: The Open Group Base Specification Issue 6,
<http://pubs.opengroup.org/onlinepubs/009695399/>, abgerufen 03.12.2015
- Firas Dib: Online regex tester and debugger: <https://regex101.com/> abgerufen 13.05.2024