Regexp - säännölliset lausekkeet

> Tässä esitettyjä säännöllisiä lausekkeita voi hyödyntää sekä Laravelin kanssa että ilman.

- Wikipedia - Säännöllinen lauseke
- Säännöllinen lauseke = regular expression, RE
- Merkkijonon malli (pattern), joka kuvaa säännöllisyyden
- Voidaan etsiä ja korvata merkkijonoja **huomattavasti monipuolisemmin** kuin perinteisillä merkkojonohauilla (CTRL+F)
- RE on eräänlainen kaava, joka joko sopii (match) tai ei sovi yhteen tutkittavan merkkijonon kanssa.

Esimerkki

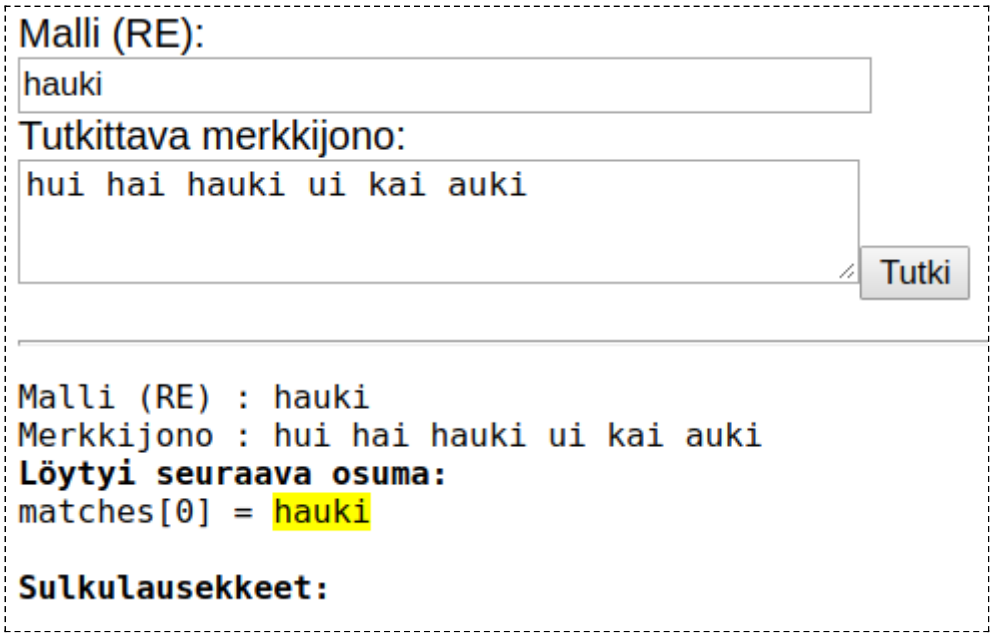
- HTML-dokumentissa runsaasti hyperlinkkejä: `"Juttua ... <a href="kohde1">Linkki1</a> juttua juttua <a href="kohde2">Linkki2</a> juttua juttua ..."`
- Haluasit löytää kaikki hyperlinkit, mutta perinteiset merkkijonofunktiot eivät osaa
- Hyperlinkin muoto noudattaa ilmiselvästi tiettyä säännöllisyyttä: `<a href="kohde">Linkki</a>`, vaikka jokaisessa linkissä on myös uniikki osuus.
- **Yllätys**: Säännöllisellä lausekkeella tällainen säännöllisyys voidaan kuvata :)
- => Voidaan löytää kaikki hyperlinkit.

PHP:n RE-funktiot

PHP käyttää nykyisin ohessa esitettyjä Perl-yhteensopivia lausekkeita. Myös Laravel käyttää sisäisesti preg_match()-funktiota syötteiden validoinnissa `regex`-säännöllä.

- `preg_ match(), preg_ match_all()` - etsi
- `preg_replace()` - etsi ja korvaa
- `preg_split()` - Jaa merkkijono RE:n määräämissä osissa taulukon alkioiksi.
- `preg_last_error()` - Näyttää viimeisen virheilmoituksen

Seuraavalla ohjelmanpätkällä voit testailla erilaisia säännöllisiä lausekkeita. Ohessa tutkittavassa merkkijonossa uiva `hauki` on löytynyt




```
<form method="get" action="<?php echo $_SERVER['PHP_SELF'];?>
Malli (RE):<br>
<input type="text" name="malli" size="40"
value="<?php if (isset($_GET['malli']))echo htmlspecialchars(
<br>

Tutkittava merkkijono:<br>
<?php

echo "<textarea name='mjono' rows='3' cols='40'>";
if (isset($_GET['mjono']))echo htmlspecialchars($_GET['mjono'
echo "</textarea>";
?>
<input type="submit" value="Tutki">
</form>
<hr>
<pre>
<?php

//Tulosten havainnollistamista varten

$s = <<< EOSSPAN
<span style="color:black;background-color:#ffff00">
EOSSPAN;
```

```php
$e = "</span>";


if (isset($_GET['malli']) AND isset($_GET['mjono']))
{
   $n_malli = htmlspecialchars($_GET['malli']);
   $n_mjono = htmlspecialchars($_GET['mjono']);
   echo "Malli (RE) : $n_malli\n";
   echo "Merkkijono : $n_mjono\n";

   $pattern = "/{$_GET['malli']}/";
   if (preg_match($pattern, $_GET['mjono'], $matches))

        {
      echo "<strong>Löytyi seuraava osuma:</strong>\n";
     $match = htmlspecialchars($matches[0]);
      echo "matches[0] = $s$match$e\n\n";
      echo "<strong>Sulkulausekkeet:</strong>\n";
      $i = 0;
      foreach ($matches as $osuma)
      {
         if($i != 0) echo "matches[$i] = $s$osuma$e<br>";
         $i++;
      }
   }
        else
   {
      echo "<strong>Ei osumaa!</strong>";
   }
}
?>
</pre>
```

Erikoismerkit

Merkki	Nimitys	Merkitys
|	putkimerkki	Vaihtoehto
^	sirkumfleksi	Seuraava lauseke merkkijonon alussa
$	dollarimerkki	Edeltävä lauseke merkkijonon lopussa
.	piste	Mikä tahansa yksittäinen merkki

Merkki	Nimitys	Merkitys
*	asteriski	Edellisen lausekkeen toisto 0 tai useamman kerran. Sama kuin {0,}
+	plusmerkki	Edellisen lausekkeen toisto 1 tai useamman kerran. Sama kuin {1,}
?	kysymysmerkki	Edellisen lausekkeen toisto 0 tai 1 kertaa. Sama kuin {0,1}
()	kaarisulut	Lausekkeen ryhmittely, myös varastointi
[]	hakasulut	Merkkijoukko
{}	aaltosulut	Edellisen lausekkeen toisto vähintään n ja enintään m kertaa {n,m}
\	kenoviiva	Metamerkin erikoismerkityksen poisto

Yhteen merkkiin viittaaminen

RE	merkkijono	sopii
e	Perl	1
\?	?	1
\?	\?	1
\\	ma\ri	1
a.i	ari	1
a.i	arsi	0
a\.i	a.i	1
a.\.i	ma.ire	1

Merkkijoukko

RE	merkkijono	sopii
[abcd]	marine	1
[a-d]	marine	1
[0123456789]	Johnsson 70 hp	1
[0-9]	Johnsson 70 hp	1
[^0123456789]	Johnsson 70 hp	1
[^0-9]	12345	0

Tavallinen merkkijono ja vaihtoehdot

RE	merkkijono	sopii
Perl	tulkitsee Perl-kieltä	1

RE	merkkijono	sopii
Perl|PHP	tulkitsee PHP-kieltä	1
Perl|PHP	kääntää C-kieltä	0
ari|ma	mari	1
ari|ma	arima	1

Merkkijonon alku ja loppu

RE	merkkijono	sopii
^foo	foofo	1
fo$	foofo	1
^foofo$	foofo	1
^fo$	foofo	0

Toistaminen

RE	merkkijono	sopii
^Ki*r	Kiiir	1
^Ki*r	Kr	1
^Ki*r	Kiar	0
^Ki{0,}r	Kiiir	1
^Ki{0,}r	Kr	1
^Ki{0,}r	Kiar	0

RE	merkkijono	sopii
^Ki+r	Kiiir	1
^Ki+r	Kr	0
^Ki+r	Kiar	0
^Ki{1,}r	Kiiir	1
^Ki{1,}r	Kr	0
^Ki{1,}r	Kiar	0

RE	merkkijono	sopii
^Ki?r	Kir	1
^Ki?r	Kiiir	0
^Ki?r	Kr	1
^Ki?r	Kiar	0
^Ki{0,1}r	Kir	1
^Ki{0,1}r	Kiiir	0

RE	merkkijono	sopii
^Ki{3}r	Kir	0
^Ki{3}r	Kiiir	1
^Ki{3}r	Kiaar	0
^Ki{2,4}r	Kir	0
^Ki{2,4}r	Kiiir	1

Ryhmittely suluilla

- Jos sulkulauseke edeltää toisto-operaattoria (`*+?{}`), toisto-operaattorin vaikutus kohdistuu koko suluilla ryhmiteltyyn lausekkeeseen
- Etsinnän tuloksen varastona sulkulausekkeen osalta.

RE	merkkijono	sopii
(ari)+	ariari	1
(ari)+	arii	0
(ari)*	arii	1
(ari)*	halibatsui	1
(ari)*(ma)	ariiima	1
(ari)+(ma)	ariiima	0

Esimerkkilausekkeita

RE	merkkijono	sopii	Kommentti
^(ar)+$	arar	1	
[a-d123]	rtoc	1	Viimeinen c sopii
^[a-d123]	rtoc	0	Ensimmäisenä merkkinä ei mikään ehdotetuista
[a-d123]$	rtoc	1	c vimeisenä merkkinä
^[a-d123]$	rtoc	0	Vain yhden merkin pituinen merkkijono voisi sopia
[a-c]+x$	Labax	1	Sopii: abax
^[a-d123]+$	a1bc	1	Kaikki merkit täyttävät sopivuusehdon
^[a-d123]+$	anna	0	Kaikki merkit eivät (nn) täytä sopivuusehtoa
^[^a-d123]+$	monni	1	Mikään poissuljettavista merkeistä ei löydy
^[^a-d123]+$	mon2ni	0	Poissuljettavista merkeistä löytyy merkki 2
^(yli|ylin) (nopisto|opisto)$	ylinopisto	1	Löytyy: 1) "yli" ja 2) "nopisto"

RE	merkkijono	sopii	Kommentti
^(yli|yli\ )(\ opisto|opisto)$	yli opisto	1	Löytyy: 1) "yli" ja 2) " opisto"
^(yli|yli\ )(\ opisto|opisto)$	yliopisto	1	Löytyy: 1) "yli" ja 2) "opisto"

Esimerkki: Kokonaisluku

```
^[+-]?[0-9]+$
```

sopii mihin tahansa kokonaislukua esittävään merkkijonoon. esim. 36, -12, +0, -0, 0, 175, jne.

Esimerkki: Desimaalimuotoinen reaaliluku

```
^[+-]?[0-9]+\.?[0-9]*$
```

Kaikki em. kokonaisluvut kelpaavat ja lisäksi esim. +0.754, -12., 0.12

Nimettyjä merkkikokoelmia

```
[[:digit:]]  - [0-9]
[[:alpha:]]  - [A-Za-z]
[[:alnum:]]  - [0-9A-Za-z]
[[:blank:]]  - välilyönti ja tabulaattori
[[:cntrl:]]  - kontrollimerkit (ASCII < 32 ja ASCII 127)
[[:print:]]  - tulostuvat merkit (ASCII: 32-126)
[[:graph:]]  - kuten [:print:] poislukien välilyönti
[[:punct:]]  - kuten [:print:] poislukien välilyönti ja a
[[:lower:]]  - pienet merkit
[[:upper:]]  - isot merkit
[[:space:]]  - välilyönti ja ASCII 9-13: TAB, LF, VTAB, F
[[:xdigit:]] - heksadesimaalit: [0-9a-fA-F]
[[:<:]]      - Sanan alku (Sana = alnum ja _)
[[:>:]]      - Sanan loppu
```

Esimerkki: Kokonainen sana

```
[[:<:]]sana[[:>:]]
```

sopii merkkijonoon "Miehen sana painaa", mutta ei merkkijonoon "Miehen sanaan voi luottaa".

Ahneus

- RE voi sopia useaan kohtaan tutkittavaa merkkijonoa
- Sovittava siitä, mitä onnistuneista osumista käytetään.
- Periaatteena maksimaalinen eli ahne sovitus
- Useista osumista valitaan se, joka
  - alkaa mahdollisimman aikaisin ja loppuu mahdollisimman myöhään, joka toisin ilmaistuna tarkoittaa
  - vasemmanpuoleisinta, pisintä mahdollista osumaa.
- Aikaisimpaan osumaan otetaan niin paljon merkkejä kuin mahdollista.
- Ensisijaisesti valitaan aina aikaisin osuma, vaikka myöhemmin löytyisi pitempiäkin osumia.

RE	Merkkijono	Osuma	Kommentti
k.*	ak111k222	k111k222	Aikaisin k, siihen pisin jatko
ari	mariamaria	ari	Osumana ensimmäinen ari
(ari)*	halibatsui	Tyhjä merkkijono	Jokaisen merkkijonon alussa on "tyhjä merkkijono", johon sovitusta yritetään ensimmäisenä. Tässä sovitus onnistuu tähän merkkijonon alkuun. Huomaa, että lopputulos olisi sama, vaikka tutkittava merkkijono olisi "mariari".
ari|ma	amaria	ma	Vaihtoehtoja sovitetaan vuorotellen ensin merkkijonon alkuun (tyhjä merkki), sitten ensimmäisestä merkistä lähtien (a). Nämä kumpikaan sovitus eivät tuo tulosta. Seuraavana sovitus tehdään toisesta merkistä (m) alkaen, johon "ma" sopii. Ensimmäisenä (ja laajentumattomana) osumana tämä jää voimaan

Sulkulausekkeet varastona

Taulukon elementteihin `$matches[1]` - `$matches[n]` tallennetaan mallissa käytettyjä sulkulausekkeita vastaavat osumat. Maksimissaan voidaan varastoida siis yhdeksän sulkulausekkeen osumat. Jos mallina (RE) on vaikkapa merkkijono

```
(ta)(vu)te(taan)
```

ja tutkittavana merkkijonona "tavutetaan", niin regs-taulukkoon tallennetut arvot ovat

- $matches[0] = tavutetaan
- $matches[1] = ta
- $matches[2] = vu
- $matches[3] = taan

Jos käytetään sisäkkäisiä sulkuja, uloimmat saavat aikaisemman järjestysnumeron. Jos mallina (RE) on vaikkapa merkkijono

```
((ta)(vu))te(taan)
```

ja tutkittavana merkkijonona "tavutetaan", niin `regs`-taulukkoon tallennetut arvot ovat

- $matches[0] = tavutetaan
- $matches[1] = tavu
- $matches[2] = ta
- $matches[3] = vu
- $matches[4] = taan

```
© #AriRantala
```