ZADKINE

leren
denken
durven
doen

PHP regular expressions

1 februari 2016

Regular Expressions

PHP offers functions specific to two sets of regular expression functions, each corresponding to a certain type of regular expression. You can use any of them based on your comfort.

- POSIX Regular Expressions
- PERL Style Regular Expressions

POSIX Regular Expressions

The structure of a POSIX regular expression is not dissimilar to that of a typical arithmetic expression: various elements (operators) are combined to form more complex expressions.

The simplest regular expression is one that matches a single character, such as g, inside strings such as g, haggle, or bag.

Regular Expressions

Expression	Description
[0-9]	It matches any decimal digit from 0 through 9.
[a-z]	It matches any character from lowercase a through lowercase z.
[A-Z]	It matches any character from uppercase A through uppercase Z.
[a-Z]	It matches any character from lowercase a through uppercase Z.

Quantifiers

The frequency or position of bracketed character sequences and single characters can be denoted by a special character. Each special character having a specific connotation. The +, *, ?, {int. range}, and $ flags all follow a character sequence.

Expression	Description
p+	It matches any string containing at least one p.
p*	It matches any string containing zero or more p's.
p?	It matches any string containing zero or more p's. This is just an alternative way to use p*.
p{N}	It matches any string containing a sequence of N p's
p{2,3}	It matches any string containing a sequence of two or three p's.
p{2, }	It matches any string containing a sequence of at least two p's.
p$	It matches any string with p at the end of it.
^p	It matches any string with p at the beginning of it.

Expression	Description
[^a-zA-Z]	It matches any string not containing any of the characters ranging from a through z and A through Z.
p.p	It matches any string containing p, followed by any character, in turn followed by another p.
^.{2}$	It matches any string containing exactly two characters.
(.*)	It matches any string enclosed within and .
p(hp)*	It matches any string containing a p followed by zero or more instances of the sequence hp.

Predefined Character Ranges

For your programming convenience several predefined character ranges, also known as character classes, are available. Character classes specify an entire range of characters, for example, the alphabet or an integer set:

Expression	Description
[[:alpha:]]	It matches any string containing alphabetic characters aA through zZ.
[[:digit:]]	It matches any string containing numerical digits 0 through 9.
[[:alnum:]]	It matches any string containing alphanumeric characters aA through zZ and 0 through 9.
[[:space:]]	It matches any string containing a space.

PHP's Regexp POSIX Functions

PHP currently offers seven functions for searching strings using POSIX-style regular expressions:

Function	Description
ereg()	The ereg() function searches a string specified by string for a string specified by pattern, returning true if the pattern is found, and false otherwise.
ereg_replace()	The ereg_replace() function searches for string specified by pattern and replaces pattern with replacement if found.
eregi()	The eregi() function searches throughout a string specified by pattern for a string specified by string. The search is not case sensitive.
eregi_replace()	The eregi_replace() function operates exactly like ereg_replace(), except that the search for pattern in string is not case sensitive.

split()	The split() function will divide a string into various elements, the boundaries of each element based on the occurrence of pattern in string.
spliti()	The spliti() function operates exactly in the same manner as its sibling split(), except that it is not case sensitive.
sql_regcase()	The sql_regcase() function can be thought of as a utility function, converting each character in the input parameter string into a bracketed expression containing two characters.

Metacharacters

A metacharacter is simply an alphabetical character preceded by a backslash that acts to give the combination a special meaning.

For instance, you can search for large money sums using the '\d' metacharacter: **/([\d]+)000/**, Here **\d** will search for any string of numerical character.

Character	Description
.	a single character
\s	a whitespace character (space, tab, newline)
\S	non-whitespace character
\d	a digit (0-9)
\D	a non-digit
\w	a word character (a-z, A-Z, 0-9, _)
\W	a non-word character
[aeiou]	matches a single character in the given set
[^aeiou]	matches a single character outside the given set
(foo|bar|baz)	matches any of the alternatives specified

Modifiers

Modifier	Description
i	Makes the match case insensitive
m	Specifies that if the string has newline or carriage return characters, the ^ and $ operators will now match against a newline boundary, instead of a string boundary
o	Evaluates the expression only once
s	Allows use of . to match a newline character
x	Allows you to use white space in the expression for clarity
g	Globally finds all matches
cg	Allows a search to continue even after a global match fails

Function	Description
preg_match()	The preg_match() function searches string for pattern, returning true if pattern exists, and false otherwise.
preg_match_all()	The preg_match_all() function matches all occurrences of pattern in string.
preg_replace()	The preg_replace() function operates just like ereg_replace(), except that regular expressions can be used in the pattern and replacement input parameters.
preg_split()	The preg_split() function operates exactly like split(), except that regular expressions are accepted as input parameters for pattern.
preg_grep()	The preg_grep() function searches all elements of input_array, returning all elements matching the regexp pattern.
preg_ quote()	Quote regular expression characters

```
<?
$regex = "/World/";
$greeting = "Hello World";
if (preg_match($regex, $greeting)) {
    print "Het komt overeen";
} else {
    print "Het komt niet overeen";
}
```

Ook kan je andere tekens dan // gebruiken als 'matchtekens, bijvoorbeeld !. Dit is met name handig wanneer je een iets waar veel forward slashes (/) in zitten wil matches (bijvoorbeeld een URL).

```
[code]<?
preg_match("!World!", "Hello World");
preg_match("{World}", "Hello World");
?>
```

Laten we nu eens kijken welke reguliere expressies zouden overeenkomen met "Hello World".

```
<?
preg_match("/world/", "Hello World"); // Matched niet
preg_match("/o W/", "Hello World");   // Matched
preg_match("/oW/", "Hello World");    // Matched niet
preg_match("/World /", "Hello World");// Matched niet
?>
```

De eerste regexp mached niet omdat regexp hoofdlettergevoelig zijn. De tweede matched omdat 'o W' voorkomt in 'Hello World'. De spatie ' ' wordt gewoon beschouwd als een spatie en heeft geen speciale betekenis. Zonder de spatie zou deze niet werken, dit is te zien in de 3e regel, 'oW' matched niet, en in het vierde voorbeeld is te zien dat wanneer er een spatie te veel staat aan het eind van de regex. De les is dat de regexp [b]precies[/b] moet matchen om overeen te komen.

Wanneer een regex meer dan een keer in een string voorkomt zal php altijd de eerste matchen.

```
<?
preg_match("/o/", "Hello World");      // Matched de eerste 'o' in 'Hello'.
preg_match("/hat/","That hat is red"); // Matched 'hat' in 'That'
?>
```

Wat verder belangrijk is is dat een paar tekens ([b]metacharacters[/b]) zijn gereserveerd voor regexp notatie. De metacharacters zijn:

{}[]()^$|*+?\

```
<?
preg_match("/2+2/","2+2=4");  // Matched niet, + is een metacharacter.

preg_match("/2\+2/","2+2=4");  // Matched nu wel

preg_match("/[0,1)./", "De interval is [0,1)");    // Ongeldige regexp syntax!

preg_match("/\[0,1\)\./", "De interval is [0,1)"); // matched

preg_match("/http:\/\/www.phphulp.nl\/","http://www.phphulp.nl"); // matched

preg_match("!http://www.phphulp.nl!","http://www.phphulp.nl");    // matched
?>
```

forward slashes de delimiter (/)
vervangen door een uitroepteken of een ander teken om de leesbaarheid te vergroten.

Ook de blackslash \ is een metacharacter en moet ook geescaped worden:

```
<?
preg_match('/C:\\WIN/', 'C:\WIN');
?>
```

In alle bovenstaande regexps geld: als de regexp ook maar ergens in de string voorkomt was de expressie geldig. Soms wil je aangeven **[b]waar[/b]** de string de regexp overeen zou moeten komen.

Om dit te doen zijn de **(anchor)** metacharacters ^ en en ^ in het leven geroepen.

De **anchor ^** betekend dat hij aan het begin van de string zou moeten matchen en de

anchor $ betekend dat hij aan het eind van de string zou moeten matchen of voor een newline aan het eind van de string.

```
<?
preg_match("/keeper/", "housekeeper");    // matches

preg_match("/^keeper/", "housekeeper");   // matched niet

preg_match("/keeper$/", "housekeeper");   // matches

preg_match("/keeper$/", "housekeeper\n"); // matches
?>
```

Het tweede voorbeeld matched niet omdat het ^ teken forceerd dat keeper alleen aan het begin matched, terwijl in "housekeeper" keeper in het midden begint. De derde regexp matched omdat keeper aan het einde van "housekeeper" staat.

Wanneer zowel ^ als $ in een regexp gebruikt worden moet de regexp zowel aan het begin als aan het eind matchen. Met andere woorden: de gehele string moet matchen.

```
<?
preg_match("/^keep$/", "keeper");   // Matched niet

preg_match("/^keeper$/", "keeper"); // Matched

preg_match("/^$/", "");             // Matched een lege string
?>
```

Maak een php script

- Voeg de tekst van een artikel in als string

- Je krijgt de volgende zoekopdrachten:

- Bestaat het woord “Burger”
- Hoevaak wordt “restaurant” genoemd
- Welke zinnen beginnen met “In”
- Welke zinnen eindigen op “investment”
- Komt er een zin voor begint met “ en eindigt met “

Burger King Nederland krijgt een nieuwe eigenaar. De huidige aandeelhouder Citoyen Food Group verkoopt het bedrijf aan investeringsmaatschappij Standard Investment.

Er werden geen financiële details bekendgemaakt.

Burger King Nederland bestaat uit 27 restaurants met in totaal veertienhonderd werknemers. Het concern is een franchisenemer van de Amerikaanse fastfoodketen.

Onder andere de vestiging bij Schiphol, naar omzet de grootste vestiging van Burger King ter wereld, valt onder het bedrijf.

In totaal zijn er zestig vestigingen van Burger King in Nederland. De rest van de resturants hebben een andere eigenaar.

Nieuwe restaurants

Standard Investments zegt met Burger King meer nieuwe restaurants te willen openen. "Het is voor ons een kans om een goedlopende onderneming naar een hoger niveau te tillen", aldus Hendrik Jan ten Have, partner bij Standard Investment.

BurgerKing maakt wereldwijd al een sterke groei door. Waar de keten in 2009 nog 12.000 restaurants omvatte, is dat in 2014 toegenomen tot bijna 14.500

Maak een php script , maar nu met een file-upload form
en tenminste één zoekargument.

Je krijgt de volgende zoekopdrachten:

- Bestaat het woord “the”
- Hoevaak wordt “fuck” genoemd
- Welke zinnen beginnen met “As”
- Welke zinnen eindigen op “thank you”
- Komt er een woord voor dat begint met A .. En eindigd met “see”?