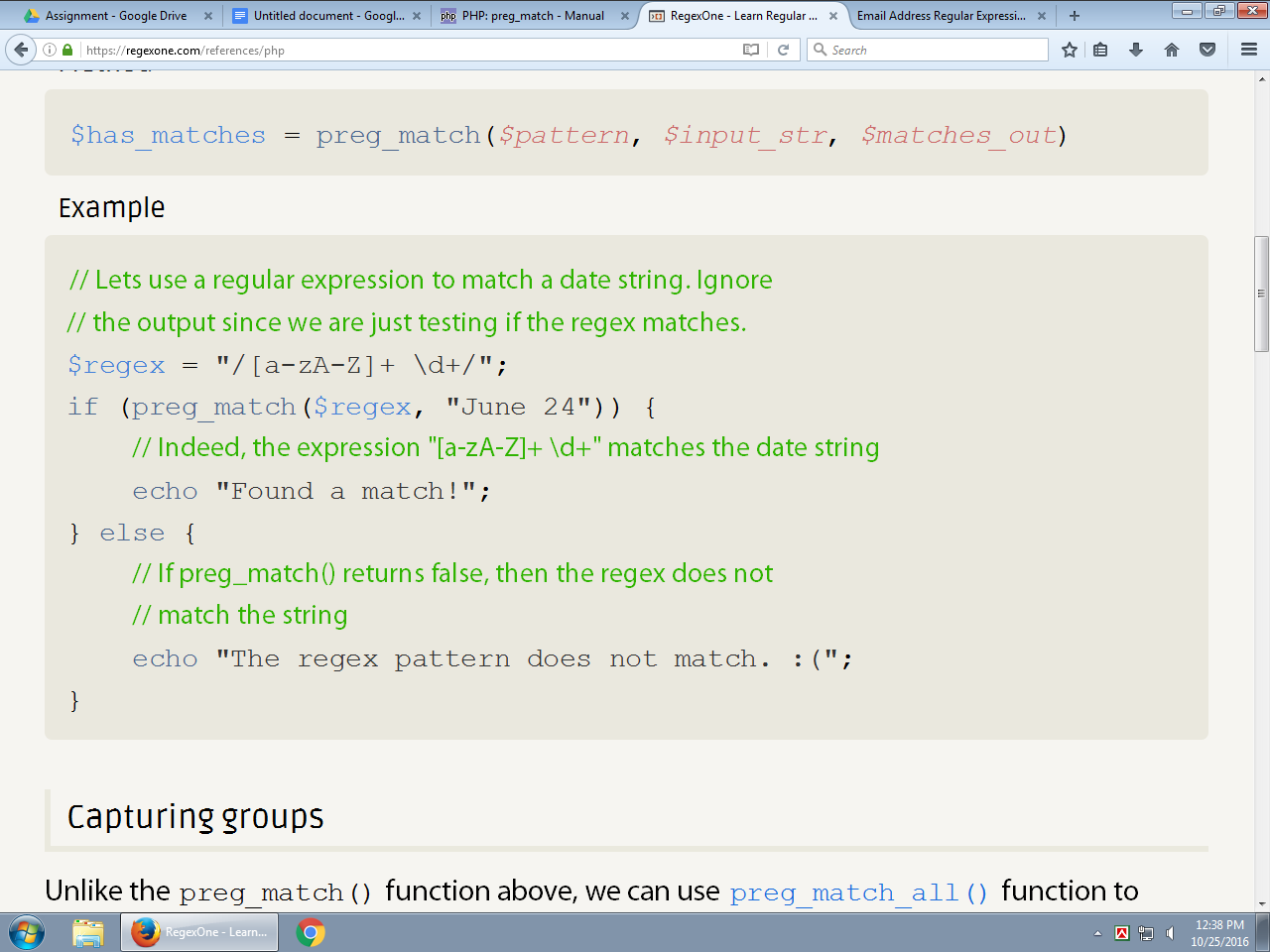
1. Installation and learning php screenshot within assignment folder
2. <http://www.w3schools.com/php/php_forms.asp>
3. 

The following link will guide you how to solve the problem

<http://www.hackingwithphp.com/4/8/2/novice-regexes>

<http://www.phpforkids.com/php/php-functions-regular-expressions.php>

<http://www.hackingwithphp.com/4/8/3/advanced-regexes>

1. Regular xpression cheat sheet
2. <http://regexlib.com/CheatSheet.aspx?AspxAutoDetectCookieSupport=1>

<http://php.net/manual/en/function.preg-match.php>

<?php

$str = 'foobar: 2008';

preg\_match('/(?P<name>\w+): (?P<digit>\d+)/', $str, $matches);

/\* This also works in PHP 5.2.2 (PCRE 7.0) and later, however

\* the above form is recommended for backwards compatibility \*/

// preg\_match('/(?<name>\w+): (?<digit>\d+)/', $str, $matches);

print\_r($matches);

?>

The above example will output:

Array  
(  
 [0] => foobar: 2008  
 [name] => foobar  
 [1] => foobar  
 [digit] => 2008  
 [2] => 2008  
)

**Write a program that can determine whether a input grammar is left associative or right associative using regular expression in php**

More examples

<http://lzone.de/examples/PHP%20preg_match>

# **PHP Preg\_match Examples**

Use preg\_match() to match strings with regular expressions. Check the return value for true to see if the expression did match.

### **1. Syntax of preg\_match**

While full syntax is

int preg\_match ( string $pattern , string $subject   
 [, array &$matches [, int $flags = 0 [, int $offset = 0 ]]] )

you propably will use preg\_match() mostly with two parameters for simply matching checks or with three to extract matches. You probably won't use the 4th and 5th parameter which can be used to return match offsets and limit matching to a given offset in the string.

### **2. Simple String Checks**

Here are some syntax examples that check strings for certain content: **Basic matching**

preg\_match("/PHP/", "PHP") # Match for an unbound literal  
preg\_match("/^PHP/", "PHP") # Match literal at start of string  
preg\_match("/PHP$/", "PHP") # Match literal at end of string  
preg\_match("/^PHP$/", "PHP") # Match for exact string content  
preg\_match("/^$/", "") # Match empty string

**Using different regex delimiters**

preg\_match("/PHP/", "PHP") # / as commonly used delimiter  
preg\_match("@PHP@", "PHP") # @ as delimiter  
preg\_match("!PHP!", "PHP") # ! as delimiter

**Changing the delimiter becomes useful in some cases**

preg\_match("/http:\/\//", "http://"); # match http:// protocol prefix with / delimiter  
preg\_match("#http://#", "http://") # match http:// protocol prefix with # delimiter

**Case sensitivity**

preg\_match("/PHP/", "PHP") # case sensitive string matching  
preg\_match("/php/i", "PHP") # case in-sensitive string matching

**Matching with wildcards**

preg\_match("/P.P/", "PHP") # match a single character  
preg\_match("/P.\*P/", "PHP") # match multipe characters  
preg\_match("/P[A-Z]P/", "PHP") # match from character range A-Z  
preg\_match("/[PH]\*/", "PHP") # match from character set P and H  
preg\_match("/P\wP/", "PHP") # match one word character  
preg\_match("/\bPHP\b/", "regex in PHP") # match the word "PHP", but not "PHP" as larger string

**Using quantifiers**

preg\_match("/[PH]{3}/", "PHP") # match exactly 3 characters from set [PH]  
preg\_match("/[PH]{3,3}/", "PHP") # match exactly 3 characters from set [PH]  
preg\_match("/[PH]{,3}/", "PHP") # match at most 3 characters from set [PH]  
preg\_match("/[PH]{3,}/", "PHP") # match at least 3 characters from set [PH]

### **3. Extracting Data**

To extract data using regular expression we have to use capture/grouping syntax.

Some basic examples

# Extract everything after the literal "START"  
preg\_match("/START(.\*)/", $string, $results)   
  
# Extract the number from a date string   
preg\_match("/(\d{4})-(\d{2})-(\d{2})/", "2012-10-20", $results)  
  
# Nesting of capture groups, extract full name, and both parts...  
preg\_match("/name is ((\w+), (\w+))/", "name is Doe, John", $results)

So you basically just enclose the sub patterns you want to extract with braces and fetch the results by passing a third parameter which preg\_match() will fill as an array.

Named Capture Groups

# Extract the number from a date string   
preg\_match("/(?P<year>\d{4})-(?P<month>\d{2})-(?P<day>\d{2})/", "2012-10-20", $results)

Now the $result array will additionally to the position matches 1, 2 and 3 contain the keys "year", "month" and "day". The advantage is never having to think of the capture positions anymore when you modify the expression!

### **Result:**

1

Array  
(  
 [0] => http://preg\_match.onlinephpfunctions.com  
 [1] => preg\_match.onlinephpfunctions.com  
)

### **PHP call:**

<?php

$pattern = '@^(?:http://)?([^/]+)@i';

$subject = 'http://preg\_match.onlinephpfunctions.com/index.php';

$result = preg\_match( $pattern, $subject , $matches );

echo $result;

print\_r($matches);

?>

<?php

$pattern = '/()()/';

$subject = '';

$result = preg\_match( $pattern, $subject , $matches );

echo $result;

print\_r($matches);

echo "<br>".$matches[1];

?>