PHP

PHP Session Handling techniques

Cookies [1]

- What is a Cookie?
  - A cookie is often used to identify a user.
  - A cookie is a small file that the server embeds on the user's computer.
  - Each time the same computer requests a page with a browser, it will send the cookie too.
  - With PHP, you can both create and retrieve cookie values.

Create Cookies With PHP[1]

- A cookie is created with the setcookie() function.
- Syntax

  setcookie(*name, value, expire, path, domain, secure, httponly*);

- Only the *name* parameter is required.
- All other parameters are optional.

expire	Optional. Specifies when the cookie expires. The value: time()+86400*30, will set the cookie to expire in 30 days. If this parameter is omitted or set to 0, the cookie will expire at the end of the session (when the browser closes). Default is 0
path	Optional. Specifies the server path of the cookie. If set to "/", the cookie will be available within the entire domain. If set to "/php/", the cookie will only be available within the php directory and all sub-directories of php. The default value is the current directory that the cookie is being set in
domain	Optional. Specifies the domain name of the cookie. To make the cookie available on all subdomains of example.com, set domain to "example.com". Setting it to www.example.com will make the cookie only available in the www subdomain
secure	Optional. Specifies whether or not the cookie should only be transmitted over a secure HTTPS connection. TRUE indicates that the cookie will only be set if a secure connection exists. Default is FALSE
httponly	Optional. If set to TRUE the cookie will be accessible only through the HTTP protocol (the cookie will not be accessible by scripting languages). This setting can help to reduce identity theft through XSS attacks. Default is FALSE

PHP Create/Retrieve a Cookie[1]

- The following example creates a cookie named "user" with the value “Parth Patel". The cookie will expire after 30 days (86400 * 30).
  - 86400 => 1 day
- The "/" means that the cookie is available in entire website (otherwise, select the directory you prefer).
- We then retrieve the value of the cookie "user" (using the global variable $_COOKIE).
- We also use the isset() function to find out if the cookie is set:

- Example 1

Modify a Cookie Value[1]

- To modify a cookie, just set (again) the cookie using the setcookie() function:

- Example 2

Delete a Cookie [1]

- To delete a cookie, use the setcookie() function with an expiration date in the past:

- Example 3

Check if Cookies are Enabled [1]

- The following example creates a small script that checks whether cookies are enabled. First, try to create a test cookie with the setcookie() function, then count the $_COOKIE array variable:

- Example 4

Sessions [1]

- A session is a way to store information (in variables) to be used across multiple pages.
- Unlike a cookie, the information is not stored on the users computer.

What is a PHP Session? [1]

- When you work with an application, you open it, do some changes, and then you close it. This is much like a Session. The computer knows who you are. It knows when you start the application and when you end. But on the internet there is one problem: the web server does not know who you are or what you do, because the HTTP address doesn't maintain state.
- Session variables solve this problem by storing user information to be used across multiple pages (e.g. username, favorite color, etc). By default, session variables last until the user closes the browser.
- So; Session variables hold information about one single user, and are available to all pages in one application.
- **Tip:** If you need a permanent storage, you may want to store the data in a database.

Start a PHP Session[1]

- A session is started with the session_start() function.
- Session variables are set with the PHP global variable: $_SESSION.
- Now, let's create a new page called “5.php". In this page, we start a new PHP session and set some session variables:
- Example 5
- **Note:** The session_start() function must be the very first thing in your document. Before any HTML tags.

Get PHP Session Variable Values[1]

- Next, we create another page called “6.php". From this page, we will access the session information we set on the first page (“5.php").
- Notice that session variables are not passed individually to each new page, instead they are retrieved from the session we open at the beginning of each page (session_start()).
- Also notice that all session variable values are stored in the global $_SESSION variable:
- Example 6

Destroy a PHP Session[1]

- To remove all global session variables and destroy the session, use session_unset() and session_destroy()

- Example 8

Examples [1]

- Using Cookie:
  - Login.html, Login.php, Logout.php
- Using Session:
  - SLogin.html, SLogin.php, SLogout.php

Include Files

Include Files [1]

- The include (or require) statement takes all the text/code/markup that exists in the specified file and copies it into the file that uses the include statement.
- Including files is very useful when you want to include the same PHP, HTML, or text on multiple pages of a website.

include and require Statements [1]

- It is possible to insert the content of one PHP file into another PHP file (before the server executes it), with the include or require statement.
- **The include and require statements are identical, except upon failure:**
  - require will produce a fatal error (E_COMPILE_ERROR) and stop the script
  - include will only produce a warning (E_WARNING) and the script will continue

include and require Statements [1]

- Example 9 -> include
- Example 10 -> require
- Example 11 -> variable include

Regular Expression in PHP

What is Regular Expression in PHP?

- Regular expressions are powerful pattern matching algorithm that can be performed in a single expression.
- Two Types of Regular Expressions in PHP
    - POSIX Regular Expressions
    - PERL Style Regular Expressions

Why to use regular expressions?

- Finding patterns in strings by calling single function.
- Searching a particular string inside another string.
- Replacing one string by another string.
- You can split a string into many chunks.
- Highlighting keywords in search results.

Why to use regular expressions?

- When validating user input such as email address, domain names, telephone numbers, IP addresses,
- When creating a custom HTML template. Regular expressions can be used to identify the template tags and replace them with actual data.

POSIX Regular Expression

- The structure of a POSIX regular expression is similar 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.

POSIX Regular Expression

- Explanation for few concepts being used in POSIX regular expression
- Brackets: used to find a range of characters.

Expression	Description
[0-9]	It matches any decimal digit from 0 through 9.
[a-z]	It matches any character from lower-case 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.

POSIX Regular Expression

- 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.

POSIX Regular Expression

- Quantifiers

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 one p's.
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.

POSIX Regular Expression

- Examples

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.
<b>(.*)</b>	It matches any string enclosed within <b> and </b>.
p(hp)*	It matches any string containing a p followed by zero or more instances of the sequence php.

POSIX Regular Expression

- Predefined Character Ranges:
  - 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

Expression	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.

PHP's Regexp POSIX Functions

Expression	Description
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.

PERL Style Regular Expressions

- Perl-style regular expressions are similar to their POSIX counterparts.
- The POSIX syntax can be used almost interchangeably with the Perl-style regular expression functions.

PERL Style Regular Expressions

- Meta characters
  - A meta character 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' meta character: **/([\d]+)000/**, Here **\d** will search for any string of numerical character.
  - Following is the list of meta characters which can be used in PERL Style Regular Expressions.

PERL Style Regular Expressions

Expression	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

PERL Style Regular Expressions

- Modifiers

Expression	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

PHP's Regexp PERL Compatible Functions

- PHP offers following functions for searching strings using Perl-compatible regular expressions –

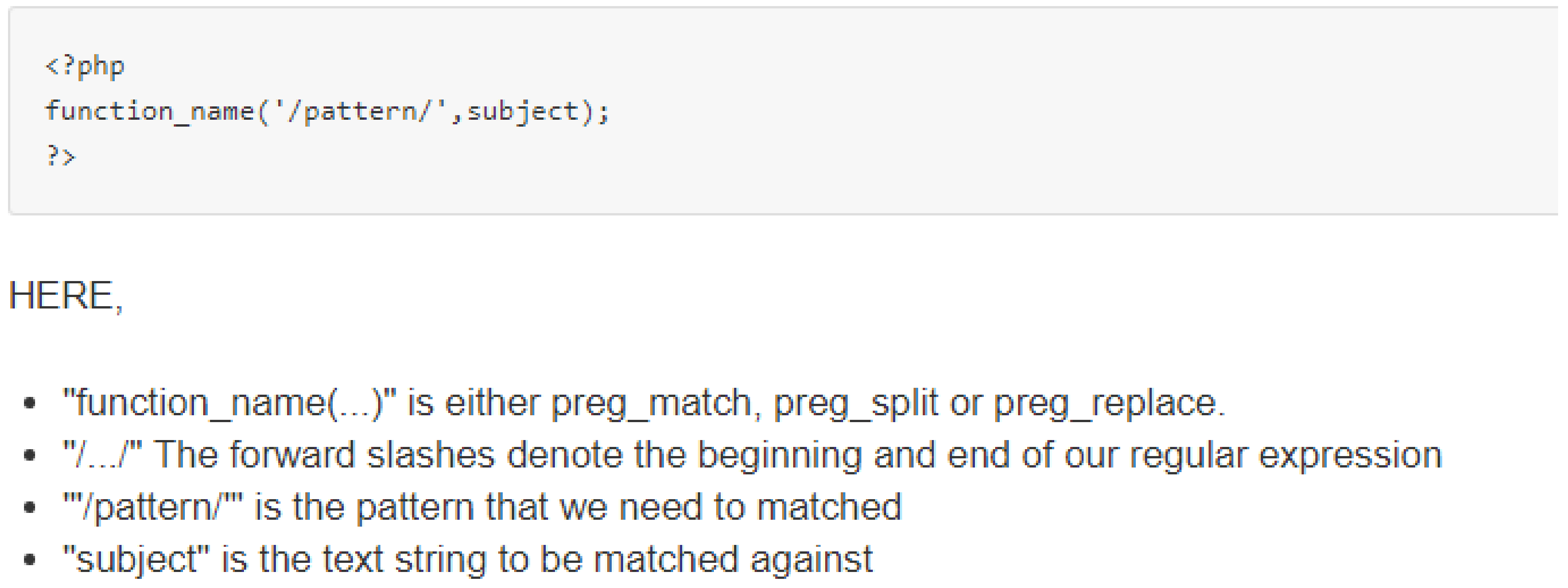
Expression	Description
preg_match()	The preg_match() function searches string for pattern, returning true if pattern exists, and false otherwise.
Preg_match_all()	The preg_match_all() function matches all occurrences of pattern in string.

PHP's Regexp PERL Compatible Functions

Expression	Description
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

PHP's Regexp PERL Compatible Functions

- Below is the syntax for a regular expression function such as preg_match,preg_split or preg_replace.



```
<?php
function_name('/pattern/',subject);
?>
```

HERE,

- "function_name(...)" is either preg_match, preg_split or preg_replace.
- "/.../" The forward slashes denote the beginning and end of our regular expression
- "'/pattern/'" is the pattern that we need to matched
- "subject" is the text string to be matched against

References

1. https://www.w3schools.com/php/

Thank you….