Server side scripting languages

9 Session & Cookies στη PHP

Β’ Τεχνική Σχολή Λεμεσού

Σχολική χρονιά 2019 – 2020

Άριστος Πασιάς

Cookies

Τα cookies χρησιμοποιούνται για να αναγνωρίζουμε χρήστες. Είναι ένα μικρό αρχείο το οποίο ο server ενσωματώνει στον υπολογιστή του χρήστη. Κάθε φορά που ο ίδιος υπολογιστής αιτείται μια σελίδα μέσω του browser, στέλνεται και το cookie μαζί. Με την php μπορούμε να δημιουργούμε και να ανακτούμε τις τιμές ενός cookie. Πρέπει να μπει πριν το <html> tag

```
setcookie(name,value,expire,path,domain);
```

```
<?php
setcookie(”user”,”Alex Porter”,time () + 3600);
?>
<html>
....
```

Cookies - Παράδειγμα

- Τα cookies ορίζονται πριν τη δήλωση του <html> tag.
- Ορισμός: setcookie(<όνομα>,<τιμή>,<λήξη>)
- Το cookie καταστρέφεται ανάλογα με την λήξη του.
- Για να καταστρέψουμε ένα cookie μπορούμε να το ορίσουμε χρησιμοποιώντας αρνητική λήξη.

```
<?php
setcookie(“username”, “Dimitrios”, time()+3600);
?>
<html>
<body>
</body>
</html>
```

Ανάκτηση τιμής ενός cookie: $_COOKIE


```
<html>
<body>
<?php
if (isset($_COOKIE[”user”]))
echo ”Welcome” . $_COOKIE[”user”] . ”!<br/>”;
else
echo ”Welcome guest!<br/>”;
// A way to view all cookies
print_r($_COOKIE);
?>
</body>
</html>
```

Διαγραφή cookie:

```
setcookie(”user”,””,time() -3600);
```

Sessions

Η session (σύνοδος) αποτελεί τον τρόπο με τον οποίο μπορούμε να αποθηκεύσουμε πληροφορία (με τη μορφή μεταβλητών) ώστε να μπορεί να χρησιμοποιηθεί μεταξύ πολλών σελίδων (username, shopping items κτλ). Αντίθετα με τα cookies, η σχετική πληροφορία των μεταβλητών δεν αποθηκεύεται στον υπολογιστή των χρηστών. Επίσης διαφέρει από τις υπόλοιπες μεταβλητές με την έννοια ότι δεν τις περνάμε ξεχωριστά σε κάθε σελίδα, αλλά τις ανακτούμε από τη σύνοδο (session), την οποία ανοίγουμε στην αρχή κάθε σελίδας. Ωστόσο, αυτού του είδους οι πληροφορίες είναι προσωρινές και συνήθως διαγράφονται πολύ σύντομα μόλις ο χρήστης αφήσει το website που χρησιμοποιεί sessions.

session start

Πριν αποθηκεύσουμε πληροφορίες στη σύνοδο, πρέπει πρώτα να την ξεκινήσουμε. Αυτό γίνεται όσο πιο νωρίς είναι δυνατό στον κώδικα, πριν οποιοδήποτε HTML κείμενο σταλεί. Το αποτέλεσμα του παρακάνω κώδικα είναι:
Pageviews = 1

```
<?php
session_start ( ) ;
$_SESSION[’views’]=1; // store session data
echo ”Pageviews = ” . $_SESSION[’views’]; //retrieve data
?>
```

Προκειμένου να πάρουμε την τιμή μιας μεταβλητής συνόδου, καλό θα ήταν να δούμε πρώτα αν έχει πάρει τιμή με τη συνάρτηση isset

```
<?php
session_start();
if (isset($_SESSION[’views’]))
$_SESSION[’views’]=$_SESSION[’views’] + 1;
else
$_SESSION[’views’]=1;
echo ”views = ” . $_SESSION[’views’];
?>
```

session destroy

Αν θέλουμε να διαγράψουμε ορισμένα δεδομένα από τη σύνοδο, μπορούμε να χρησιμοποιήσουμε την unset()

```
<?php
unset($_SESSION['views']);
?>
```

Μπορούμε να τελειώσουμε μια σύνοδο, χρησιμοποιώντας τη συνάρτηση session_destroy. Όταν γίνει αυτό, σημαίνει ότι όλα τα δεδομένα που αποθηκεύτηκαν στη σύνοδο θα χαθούν!

```
<?php
session_start();
session_destroy();
?>
```

session Παράδειγμα

Φόρμα Συμπλήρωσης Στοιχείων

```
<form name="login_form"
method="post"
action="login.php">
Email: <input type="text"
name="take_email"><br>
Password: <input
type="password"
name="take_password"><br>
Sex: <input type="radio"
name="sex"
value="yes">Male</input>
<input type="radio" name="sex"
value="no">Female</input>
<input type="submit"
value="Login">
</form>
```

```
<?session_start();?> ....
<?
function process_form(){
switch ($_POST["sex"]) {
case "yes": print "Καλωσήρθατε Κύριε
(".$_SESSION["take_email_session"].")";
break;
case "no": print "Καλωσήρθατε Κυρία
(".$_SESSION["take_email_session"].")";
break;
default:
print "Καλωσήρθατε (".$_SESSION["take_email_session"].")";
break; } }
?>
<body>
<?
if(!isset($_SESSION["take_email_session"]) ){
if (((strlen($_POST["take_email"])<6) ||
(strlen($_POST["take_password"])<6))){
print "Δεν έχετε δώσει σωστά τα δεδομένα";
exit(); }
else{
$_SESSION["take_email_session"]=$_POST["take_email"];
$_SESSION["sex_session"]=$_POST["sex"]; }
process_form(); }
else
process_form();
?>
<a href=logoff.php>LOGOFF</a>
```

include - require

Μπορούμε να εισάγουμε τα περιεχόμενα ενός PHP αρχείου σε ένα άλλο, πριν ο server ξεκινήσει να τα εκτελεί. Αυτό γίνεται με τις:

- include() σε περίπτωση σφάλματος δημιουργεί μια προειδοποίηση, αλλά ο κώδικας συνεχίζεται να εκτελείται
- require() σε περίπτωση σφάλματος δημιουργεί ένα fatal error και ο κώδικας σταματάει να εκτελείται.

```
<?php include(”header.php”);?>
<h1>Welcome to my home page!</h1>
<p>Some text.</p>
```

Error Handling

Η διαχείριση σφαλμάτων είναι σημαντικό μέρος της ανάπτυξης web εφαρμογών. Αν ο κώδικας δεν ελέγχει για τυχόν σφάλματα, υπάρχει πιθανότητα να προκύψουν θέματα ασφάλειας δεδομένων.
Μέθοδοι αντιμετώπισης σφαλμάτων:

- δηλώσεις τύπου ”die()”
- αναφορά σφάλματος
- καθορισμός τύπων σφαλμάτων

Error Handling

```
<?php
$file=fopen(”welcome.txt”,”r”);
?>
Warning : fopen (welcome.txt) [function.fopen]:failed to open
stream
:
No such file or directory in C:\webfolder\test.php on line 2
```

```
<?php
if (!file_exists(”welcome.txt”)) { die (”File not found”); }
else
{ $file=fopen(”welcome.txt”,”r”);} ?>
File not found
```

Debugging

- Κοιτάμε τον κώδικα!
- Κοιτάμε το log του server (σε unix συνήθως είναι στο /var/log/apache2/error.log).
- Χρησιμοποιούμε echo, print_r, var_dump για να δούμε αν μεταβλητές ή αντικείμενα έχουν τις τιμές που θέλουμε.
- Ελέγχουμε αν όλα τα απαραίτητα πακέτα είναι εγκατεστημένα στον server.
- Χρησιμοποιούμε το logging σύστημα του framework.


942
314
403
644
172
821
1,540
13,752
PHP
ASP.NET
Ruby
Perl
Python
ColdFusion
Java
Other
php
MySQL
Toolbox