ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

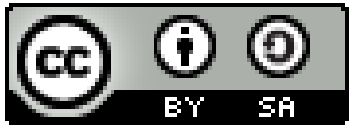
«Προχωρημένα Θέματα Προγραμματισμού με PHP για την Ανάπτυξη Δυναμικών Διαδικτυακών Εφαρμογών»

Γιάννης Σαμωνάκης
Πανεπιστήμιο Κρήτης

Σεμινάριο: Ανάπτυξη Διαδικτυακών Εφαρμογών με HTML(5) – CSS(2,3) – MYSQL - PHP
Ημερομηνία: 13/07/2015

ΜΟΝΑΔΕΣ ΑΡΙΣΤΕΙΑΣ
ΑΝΟΙΧΤΟΥ ΛΟΓΙΣΜΙΚΟΥ

ΣΗΜΕΙΩΜΑ ΑΔΕΙΟΔΟΤΗΣΗΣ



Περιεχόμενα

- Πρώτο Μέρος
  - Σύντομη επισκόπηση βασικών θεμάτων της PHP
  - Προχωρημένα θέματα της PHP, δίνοντας έμφαση στην ανάπτυξη διαδικτυακών εφαρμογών
- Δεύτερο Μέρος
  - Υλοποίηση χρήσιμων use cases που εμφανίζονται συχνά κατά την ανάπτυξη διαδικτυακών εφαρμογών

Στη συνέχεια…

- Πρώτο Μέρος
  - Σύντομη επισκόπηση βασικών θεμάτων της PHP
  - Προχωρημένα θέματα της PHP, δίνοντας έμφαση στην ανάπτυξη διαδικτυακών εφαρμογών
- Δεύτερο Μέρος
  - Υλοποίηση χρήσιμων use cases που εμφανίζονται συχνά κατά την ανάπτυξη διαδικτυακών εφαρμογών

Τι είναι η PHP

- Server-side scripting γλώσσα προγραμματισμού
- Μπορεί επίσης να χρησιμοποιηθεί και ως γλώσσα προγραμματισμού γενικού σκοπού

Που μπορεί να χρησιμοποιηθεί

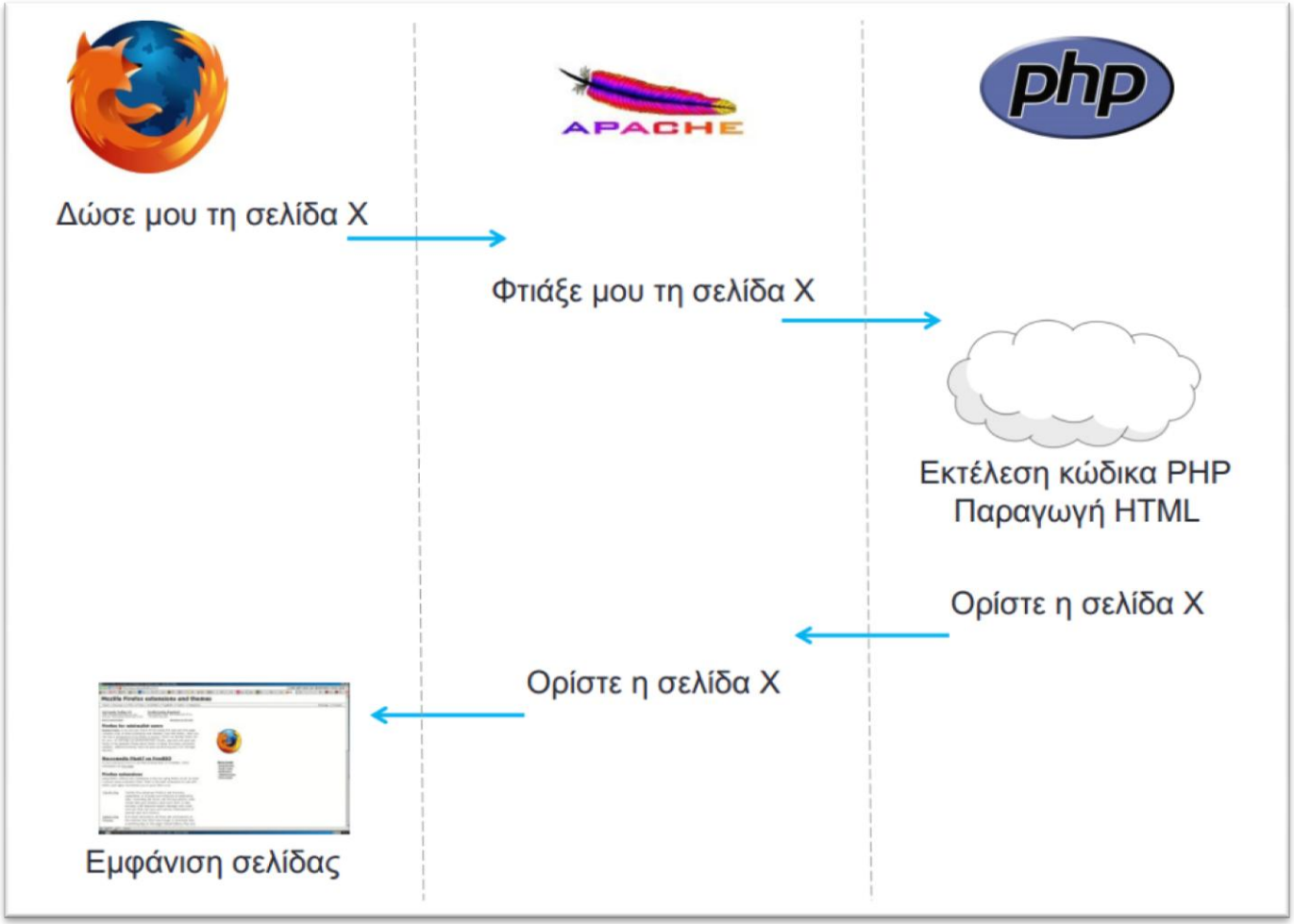
- Στην ανάπτυξη διαδικτυακών εφαρμογών (server-side scripting)
- Στην ανάπτυξη εφαρμογών που τρέχουν από τη γραμμή εντολών. Κατάλληλο για την εκτέλεση επαναλαμβανόμενων εργασιών περιβάλλον Linux και Windows
- Στην ανάπτυξη desktop εφαρμογών, με τη βοήθεια σχετικών επεκτάσεων (PHP-GTK)

Τι θα χρειαστούμε

Για την ανάπτυξη μιας διαδικτυακής εφαρμογής θα πρέπει:

- Να είναι εγκατεστημένη η PHP στο μηχάνημα που θα τρέχει η εφαρμογή
- Να είναι εγκατεστημένος ένας web server (λχ τον Apache HTTP server), στο μηχάνημα που θα τρέχει η εφαρμογή
- Να έχουμε πρόσβαση σε μια βάση δεδομένων, στην περίπτωση που θέλουμε να αναπτύξουμε δυναμικές διαδικτυακές εφαρμογές
- Να έχουμε στη διάθεσή μας ένα φυλλομετρητή ιστού

Ένα τυπικό σενάριο



Βασικές έννοιες της PHP

- Τύποι δεδομένων: String, Integer, Float, double, Boolean, Array, Object, NULL
- Τελεστές: Αριθμητικοί, λογικοί, ανάθεσης, σύγκρισης, αυξομείωσης, συμβολοσειρών, τελεστές πάνω σε arrays
- Συναρτήσεις
- Εμβέλεια μεταβλητών: Local, Global, Static
- Εντολές διακλάδωσης
- Εντολές επανάληψης
- Λειτουργίες διασύνδεσης με βάσεις δεδομένων

Στη συνέχεια…

- Πρώτο Μέρος
  - Σύντομη επισκόπηση βασικών θεμάτων της PHP
  - Προχωρημένα θέματα της PHP, δίνοντας έμφαση στην ανάπτυξη διαδικτυακών εφαρμογών
- Δεύτερο Μέρος
  - Υλοποίηση χρήσιμων use cases που εμφανίζονται συχνά κατά την ανάπτυξη διαδικτυακών εφαρμογών

Superglobal Variables

- Πρόκειται για built-in μεταβλητές που είναι διαθέσιμες από παντού.
- Μπορούν να προσπελαστούν από όλες τις συναρτήσεις, κλάσεις και αρχεία, χωρίς να χρειάζεται να κάνουμε κάτι το ιδιαίτερο

- Σημαντικές superglobal variables στην PHP:
  - $GLOBALS : Περιέχει όλες τις global μεταβλητές που δηλώνουμε
  - $_SERVER: Περιέχει πληροφορίες για headers, paths, script locations
  - $_POST: Συγκεντρώνει δεδομένα μετά την υποβολή φορμών με τη μέθοδο post
  - $_GET: Συγκεντρώνει δεδομένα μετά την υποβολή φορμών με τη μέθοδο get
  - $_COOKIE: Περιέχει πληροφορίες για τα cookies της εφαρμογής μας
  - $_SESSION: Περιέχει πληροφορίες για το session της εφαρμογής μας

PHP Cookies (i)

- Το cookie είναι ένα μικρό αρχεία που ο server τοποθετεί στον υπολογιστή του χρήστη. Έτσι κάθε φορά που ο υπολογιστής αυτός ζητά μια σελίδα μέσω ενός browser, στέλνει επίσης και το cookie.
- Τα cookies χρησιμοποιούνται για να αναγνωρίζουν ένα χρήστη και τις προτιμήσεις του, διευκολύνοντας έτσι την πλοήγησή του στο web
- Με την PHP μπορεί κανείς να δημιουργεί cookies και να αντλεί τις τιμές τους.

PHP Cookies (ii)

- Δημιουργία, τροποποίηση και διαγραφή ενός cookie:

  setcookie(name, value, expire, path, …);

- Ανάγνωση τιμών cookie: χρησιμοποιούμε τη global variable $_COOKIE, αφού πρώτα ελέγξουμε με την isset(), αν υπάρχει το συγκεκριμένο cookie

PHP Cookies (iii)

Παράδειγμα

```
<!DOCTYPE html>
<?php
$cookie_name = "user";
$cookie_value = "John Doe";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/"); // 86400 = 1 day
?>
<html>
<body>

<?php
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}
?>

<p><strong>Note:</strong> You might have to reload the page to see the value of the cookie.</p>

</body>
</html>
```

Σημείωση: Η συνάρτηση setcookie() πρέπει να καλείται πριν το <html> tag.

PHP Sessions (i)

- Κάθε φορά που ένας χρήστης ανοίγει μια εφαρμογή, δημιουργείται ένα session, το οποίο λήγει όταν πλέον ο χρήστης την κλείσει
- Στην περίπτωση των διαδικτυακών εφαρμογών, δεν υπάρχει τρόπος να το γνωρίζουμε αυτό γιατί το HTTP πρωτόκολλο δεν διατηρεί το state
- Το πρόβλημα αυτό λύνεται με τη $_SESSION η οποία μπορεί να φιλοξενήσει πληροφορίες του χρήστη (username, προτιμήσεις, κλπ) σε όλες τις σελίδες της εφαρμογής

PHP Sessions (ii)

- Ένα session δημιουργείται με τη συνάρτηση session_start() που πρέπει να καλείται πριν το <html> tag.
- Τροποποίηση μεταβλητών ενός session: χρησιμοποιούμε τη global variable $_SESSION

```
$_SESSION["favcolor"] = "yellow";
```

- Ανάγνωση μεταβλητών ενός session: χρησιμοποιούμε τη global variable $_SESSION

```
echo “Welcome" . $_SESSION[“username"] . ".<br>";
```

- Διαγραφή ενός session:

```
// remove all session variables
session_unset();

// destroy the session
session_destroy();
```

Submitting PHP Forms (i)

Το αρχείο index.php

```
<html>
<body>
<form action="login.php" method="post">
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>
</body>
</html>
```

Το αρχείο login.php

```
<html>
<body>
Welcome <?php echo $_POST["name"]; ?><br>
Your email address is: <?php echo $_POST["email"]; ?>
</body>
</html>
```

Submitting PHP Forms (ii)

Το αρχείο index.php

```
<html>
<body>
<form action="login.php" method="get">
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>
</body>
</html>
```

Το αρχείο login.php

```
<html>
<body>
Welcome <?php echo $_GET["name"]; ?><br>
Your email address is: <?php echo $_GET["email"]; ?>
</body>
</html>
```

Submitting PHP Forms (iii)

- Στην PHP τόσο με τη POST όσο και με τη GET δημιουργείται ένα array που περιέχει τα ονόματα και τις αντίστοιχες τιμές των μεταβλητών που είναι της μορφής: array(key => value, key2 => value2, key3 => value3, ...)
- Οι τιμές των μεταβλητών είναι προσπελάσιμες από τις $_POST και $_GET αντίστοιχα

HTTP Request Methods (i)

- Οι πιο συχνά χρησιμοποιούμενες μέθοδοι είναι η POST και η GET
- Με τη GET ζητάμε να αντλήσουμε δεδομένα για ένα πόρο.
- Παράδειγμα GET request:
  /test/demo_form.php**?name1=value1&name2=value2**
- Με την POST υποβάλλουμε δεδομένα προς επεξεργασία για ένα πόρο
- Παράδειγμα POST request

  POST /test/demo_form.asp HTTP/1.1
  Host: w3schools.com
  name1=value1&name2=value2

HTTP Request Methods (ii)

- Τα αιτήματα με GET:
  - Μπορούν να διατηρηθούν στην cash
  - Μπορούν να διατηρηθούν στο ιστορικό του browser
  - Μπορούν να προστεθούν στα bookmarks
  - Δεν πρέπει ποτέ να χρησιμοποιούνται όταν έχουμε να κάνουμε με ευαίσθητα δεδομένα
  - Πρέπει να χρησιμοποιούνται μόνο για άντληση πληροφορίας
  - Έχουν περιορισμούς στο μήκος του αιτήματος

HTTP Request Methods (iii)

- Τα αιτήματα με POST:
  - Δεν μπορούν να διατηρηθούν στην cash
  - Δεν μπορούν να διατηρηθούν στο ιστορικό του browser
  - Δεν μπορούν να προστεθούν στα bookmarks
  - Δεν έχουν περιορισμούς στο μήκος του αιτήματος
  - Είναι πιο ασφαλή από τα αντίστοιχα αιτήματα με GET
  - Είναι κατάλληλα για τις περιπτώσεις που το αίτημά μας πρόκειται να επιφέρει κάποια αλλαγή στα δεδομένα (ενημέρωση/διαγραφή)

Form Submitting - Security Tips (i)

- Cross site scripting (XSS): επιτρέπει στους hackers να εμφυτεύσουν επικίνδυνα scripts στη σελίδα που βλέπει ο χρήστης.
- Παράδειγμα

  Η δήλωση:

  <form method="post" action="<?php echo $_SERVER["PHP_SELF"];?>">

  Μεταφράζεται σε:

  <form method="post" action="login.php">

  Αυτό σημαίνει ότι ο hacker μπορεί να δώσει το URL:

  http://www.example.com/login.php/%22%3E%3Cscript%3Ealert('hacked')%3C/script%3E

  Που μεταφράζεται σε:

  <form method="post" action="login.php/"><script>alert('hacked')</script>

Form Submitting - Security Tips (ii)

- Λύση στο cross site scripting (XSS): χρήση της συνάρτησης htmlspecialchars()
- htmlspecialchars(): μετατρέπει τους ειδικούς χαρακτήρες σε HTML entities
- Παράδειγμα

Δηλώνοντας:

```
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
```

Και ο hacker δώσει το URL:

```
http://www.example.com/login.php/%22%3E%3Cscript%3Ealert('hacked')%3C/script%3E
```

Θα μεταφραστεί τελικά σε:

```
<form method="post"
    action="login.php/&quot;&gt;&lt;script&gt;alert('hacked')&lt;/script&gt;">
```

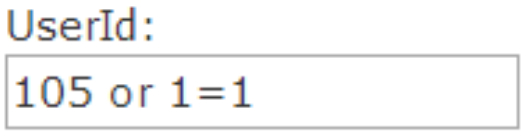
Form Submitting - Security Tips (iii)

SQL Injection

- Έστω ότι έχουμε το sql statement:

```
$userid = $_POST('UserId');
$stmt = "SELECT * FROM Users WHERE id= " + $userid;
```

- Και στην HTML form εισάγουμε:



- Οπότε το sql statement γίνεται:

```
SELECT * FROM Users WHERE id= 105 or 1=1
```

Form Submitting - Security Tips (iv)

ΑντιμετώπισηSQL Injection

- Χρησιμοποιούμε SQL parameters
- Παράδειγμα:

```
$stmt = $pdo->prepare("INSERT INTO Customers
(CustomerName, Address, City) VALUES (:nam, :add, :cit)");

$stmt->bindParam(':nam', $txtNam);
$stmt->bindParam(':add', $txtAdd);
$stmt->bindParam(':cit', $txtCit);

$stmt->execute();
```

Form Submitting - Security Tips (v)

- trim() και stripslashes(): για να αφαιρέσουμε άχρηστους χαρακτήρες (extra spaces, tabs, newlines) και τα backslashes (\) από το user input
- md5(): MD5 message-digest κρυπτογραφικός αλγόριθμος παίρνει σαν είσοδο ένα αλφαριθμητικό και παράγει ένα αποτύπωμα 128-bit. Μπορούμε να το χρησιμοποιούμε για τα passwords

PHP Filters (i)

- Τα φίλτρα της PHP χρησιμοποιούνται για να
  - Δια πιστώσουμε αν η μορφή των δεδομένων μας είναι η σωστή
  - Καθαρίσουμε τα δεδομένα μας από ανεπιθύμητους χαρακτήρες
- Θα πρέπει να φιλτράρουμε τα δεδομένα που λαμβάνουμε από εξωτερικές πηγές (forms, cookies, web services, BDs, …) γιατί:
  - Μη έγκυρα δεδομένα είναι δυνατό να δημιουργήσουν προβλήματα ασφάλειας της εφαρμογής μας
  - Μη έγκυρα δεδομένα είναι δυνατό να προκαλέσουν «κρασάρισμα » της εφαρμογής μας
  - Είναι σημαντικό η εφαρμογή μας να επεξεργάζεται έγκυρα δεδομένα

PHP Filters (ii)

- Χρησιμοποιούμε τη συνάρτηση filter_var()
- Παραδείγματα:
  - $str = "<h1>Hello World!</h1>";
    $newstr = filter_var($str, FILTER_SANITIZE_STRING); //Sanitize a string
  - $int = 100;
    if (!filter_var($int, FILTER_VALIDATE_INT) === false) { // Check if is integer
    echo("Integer is valid");
    } else {
    echo("Integer is not valid");
  - $ip = "127.0.0.1";
    if (!filter_var($ip, FILTER_VALIDATE_IP) === false) { // Check if is valid IP address
    echo("$ip is a valid IP address");
    } else {
    echo("$ip is not a valid IP address");
    }

PHP Filters (iii)

- Παραδείγματα (συνέχεια):
  - $email = "john.doe@example.com";
    // Remove all illegal characters from email
    $email = filter_var($email, FILTER_SANITIZE_EMAIL);
    // Validate e-mail
    if (!filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
      echo("$email is a valid email address");
    } else {
      echo("$email is not a valid email address");
    }

PHP Exceptions (i)

- Το exception είναι ένα γεγονός που συμβαίνει κατά την εκτέλεση του προγράμματος, το οποίο διακόπτει την κανονική ροή εκτέλεσης των εντολών.
- Χρησιμοποιούνται για να συλλάβουν σφάλματα ή άλλα εξαιρετικά γεγονότα αναπροσαρμόζοντας τη ροή εκτέλεσης του προγράμματος
- Συχνά χρησιμοποιούνται για να εντοπίσουν ένα «εξωτερικό» γεγονός το οποίο εμποδίζει την κανονική εκτέλεση του προγράμματός μας. Λχ
  - Αστοχία σύνδεσης με βάση δεδομένων
  - Ανεπάρκεια αποθηκευτικού χώρου
  - Σφάλμα κατά την άντληση δεδομένων από έναν εξυπηρετητή
  - Αδυναμία σύνδεσης στο δίκτυο

PHP Exceptions (ii)

- Δύο βασικά στάδια:
  - Παραγωγή exception, με το thrown

```
function checkNum($number) {
    if($number>1) {
      throw new Exception("Value must be 1 or below");
    }
    return true;
}
```

  - Σύλληψη του exception, με το try-catch

```
try {
    checkNum(2);
    //If the exception is thrown, this text will not be shown
    echo 'If you see this, the number is 1 or below';
}
   //catch exception
    catch(Exception $e) {
    echo 'Message: ' .$e->getMessage();
}
```

PHP Exceptions (iii)

- Μπορούμε να δημιουργήσουμε τις δικές μας κλάσεις για τα exceptions:

```
class customException extends Exception {
    public function errorMessage() {
      //error message
      $errorMsg = 'Error on line '.$this->getLine().' in '.$this->getFile()
      .': <b>'.$this->getMessage().'</b> is not a valid E-Mail address';
      return $errorMsg;
    }
  }

  function checkEmail($email) {
    if(filter_var($email, FILTER_VALIDATE_EMAIL) === FALSE) {
      //throw exception if email is not valid
      throw new customException($email);
    }
    return true;
  }

  $email = "someone@example...com";
  try {
    //check if is valid email address
        checkEmail($email);

  }
  catch (customException $e) {
    //display custom message
    echo $e->errorMessage();
  }
```

Εργαλεία Debugging (i)

- Η ενεργοποίηση μηνυμάτων της PHP σε περίπτωση σφαλμάτων πραγματοποιείται από το php.ini file, με τις ακόλουθες οδηγίες:
  - error_reporting: είναι το λεγόμενο error reporting level
  - display_errors: θέλουμε να εμφανίζονται τα μηνύματα λάθους ή όχι
  - display_startup_errors: θέλουμε να εμφανίζονται τα μηνύματα λάθους κατά την εκκίνησή της PHP ή όχι
  - log_errors: αν θέλουμε ή όχι να καταγράφονται τα μηνύματα λάθους σε log files
  - html_errors: ενεργοποιεί/απενεργοποιεί HTML tags που διευκολύνουν την προβολή των μηνυμάτων
- Υπάρχουν αντίστοιχες συναρτήσεις που θέτουν τις οδηγίες του php.ini file κατά το runtime

Εργαλεία Debugging (ii)

- Γνώμονας μας είναι σε ποιο mode (production/development) τρέχει η εφαρμογή μας
- Καλή πρακτική:

```php
// toggle this to change the setting
define('DEBUG', true);
// you want all errors to be triggered
error_reporting(E_ALL);

if(DEBUG == true)
{
    // you're developing, so you want all errors to be shown
    display_errors(true);
    // logging is usually overkill during dev
    log_errors(false);
}
else
{
    // you don't want to display errors on a prod environment
    display_errors(false);
    // you definitely wanna log any occurring
    log_errors(true);
}
```

Εργαλεία Debugging (iii)

- Χρήσιμες συναρτήσεις για την παρακολούθηση των τιμών των μεταβλητών: var_dump(), print_r();
- Σε λειτουργίες που περιλαμβάνουν σύνδεση με βάση δεδομένων, είναι καλό να ελέγχουμε αν όλα πήγαν καλά

Development Tips

- Δίνουμε έμφαση στην ασφάλεια
- Κάνουμε validate το user input
- Εντοπίζουμε ποια τμήματα της HTML σελίδας πρέπει να γίνουν δυναμικά και προχωράμε σταδιακά στην υλοποίησή τους
- Χρησιμοποιούμε το charset=utf8
- Κάνουμε clear τα cashes των browsers

Στη συνέχεια…

- Πρώτο Μέρος
  - Σύντομη επισκόπηση βασικών θεμάτων της PHP
  - Προχωρημένα θέματα της PHP, δίνοντας έμφαση στην ανάπτυξη διαδικτυακών εφαρμογών
- Δεύτερο Μέρος
  - Υλοποίηση χρήσιμων use cases που εμφανίζονται συχνά κατά την ανάπτυξη διαδικτυακών εφαρμογών

Παραπομπές

- W3schools.com - PHP Tutorial:
  - http://www.w3schools.com/php/default.asp
- Εισαγωγικό tutorial για PHP:
  - http://devzone.zend.com/6/php-101-php-for-the-absolute-beginner/
- Επίσημος ιστότοπος της PHP:
  - http://php.net
- Πληροφορίες για server-side scripting:
  - http://en.wikipedia.org/wiki/Server-side_scripting