ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

ΑΝΟΙΚΤΑ
ΑΚΑΔΗΜΑΪΚΑ
ΜΑΘΗΜΑΤΑ

Εκπαιδευτικά Περιβάλλοντα Διαδικτύου

Ενότητα 7: Σημειώσεις στην PHP

Θρασύβουλος-Κωνσταντίνος Τσιάτσος
Τμήμα Πληροφορικής

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «Ανοικτά Ακαδημαϊκά Μαθήματα στο Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

ΑΝΟΙΚΤΑ
ΑΚΑΔΗΜΑΪΚΑ
ΜΑΘΗΜΑΤΑ

Σημειώσεις στην PHP

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Περιεχόμενα ενότητας

1. PHP
    i. Tags
    ii. Εντολές
    iii. Μεταβλητές
    iv. Σχόλια
    v. Τύποι δεδομένων
    vi. Τελεστές
    vii. Δομές ελέγχου
    viii. Επανάληψη
    ix. Sessions

Σκοποί ενότητας

- Παρουσίαση της γλώσσας PHP & της MySQL
- Σύνδεση μεταξύ PHP-MySQL
- Χαρακτηριστικά και δομές της PHP

ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

PHP

MySQL

Εισαγωγή

Τι είναι η PHP 1/4

- Αρχικά σήμαινε Personal Home Page, αλλά μεταγενέστερα κατέληξε σε **P**hp **H**ypertext **P**reprocessor
- Open-source, server-side scripting γλώσσα σχεδιασμένη ειδικά για το Διαδίκτυο
- Μέσα σε μία απλή HTML σελίδα μπορεί να παρεμβληθεί κώδικας PHP, οποίος μεταφράζεται στον Web server σε HTML ή σε περιεχόμενο που μπορεί να δει ο χρήστης

Τι είναι η PHP 2/4

- Επινοήθηκε αρχικά το 1994 από τον Rasmus Lerdorf
- Υιοθετήθηκε από πολλούς ταλαντούχους ανθρώπους και υπέστη τρεις μεγάλες αναβαθμίσεις ώστε να γίνει το προϊόν που βλέπουμε σήμερα
- Έρευνα τον 10/2005 έδειξε ότι περισσότερα από 23 εκατομμύρια domains παγκοσμίως χρησιμοποιούσαν την PHP και αυτός ο αριθμός αυξανόταν ραγδαία

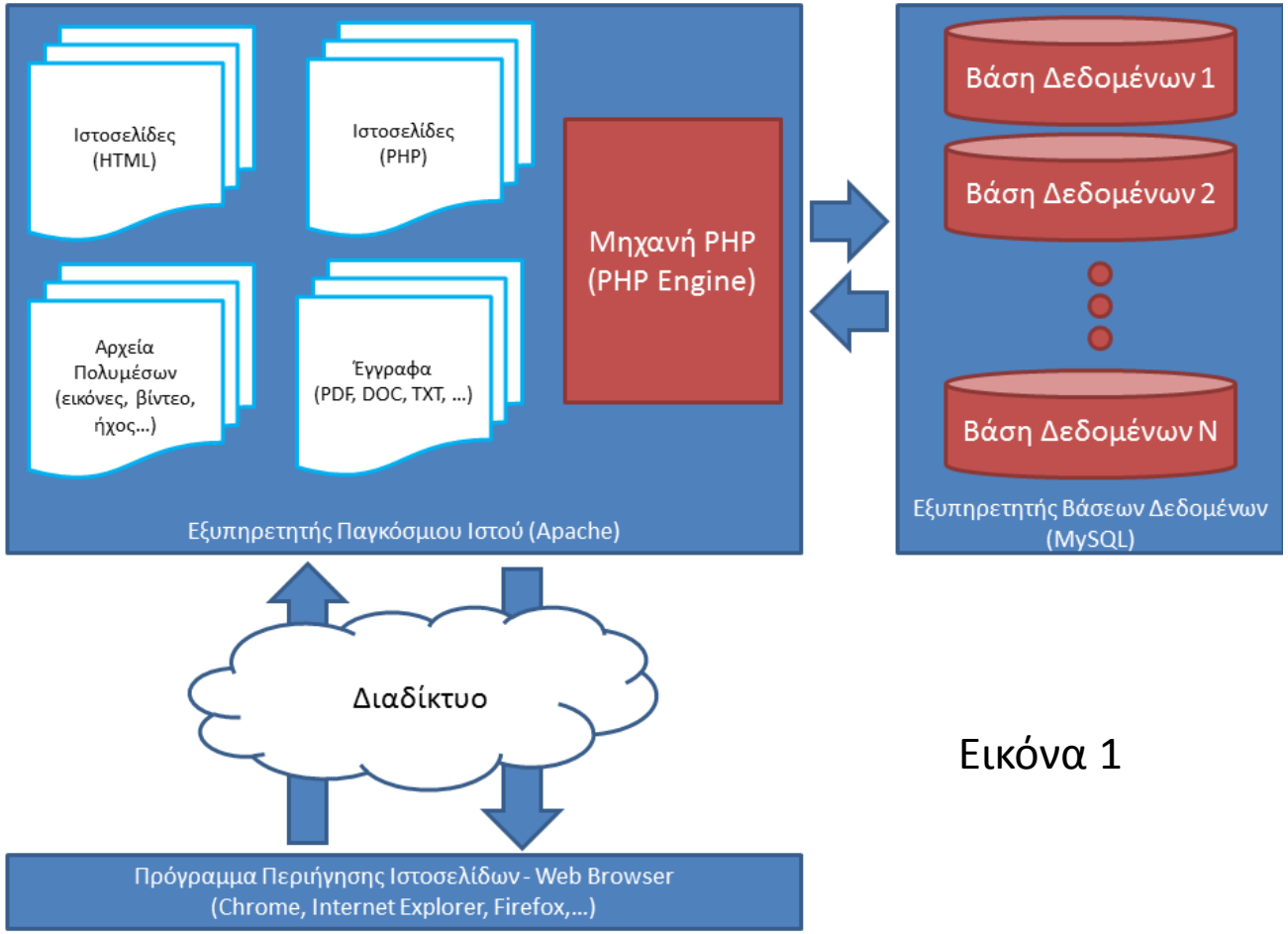
Τι είναι η PHP 3/4

- Πλεονεκτήματα:
  - Υψηλή απόδοση
  - Δυνατότητα σύνδεσης με διάφορα συστήματα διαχείρισης βάσης δεδομένων
  - Πληθώρα βιβλιοθηκών
  - Χαμηλό κόστος
  - Ευκολία εκμάθησης και χρήσης
  - Μεταφερσιμότητα
  - Πρόσβαση στον πηγαίο κώδικα

Τι είναι η PHP 4/4

- Είναι πιθανότατα η πιο διαδεδομένη γλώσσα script στον Παγκόσμιο Ιστό πληροφοριών από την πλευρά του εξυπηρετητή (server).
- Μπορεί να χρησιμοποιηθεί για την δημιουργία ιστοσελίδων πρόσβασης σε έναν ιστότοπο (login, sign-in), για λήψη πληροφοριών από φόρμες, δημιουργία αλληλεπιδραστικών και συνεργατικών εφαρμογών, όπως forum, παρουσίαση πληροφοριών από βάσεις δεδομένων και πολλές ακόμη λειτουργίες
- Ο κώδικας PHP εκτελείται από την πλευρά του εξυπηρετητή και δημιουργεί κώδικα HTML που παρουσιάζεται στον επισκέπτη μιας ιστοσελίδας

Πως συνεργάζονται



Εικόνα 1

ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

Βασική Σύνταξη

PHP

Βασική Σύνταξη 1/3

- Ένα πρόγραμμα (script) PHP εκτελείται στον server και η HTML που δημιουργείται αποστέλλεται στον browser
- Ένα script της PHP πάντα ξεκινά (ανοίγει) με <?php και κλείνει με ?> εναλλακτικά ένα script μπορεί να ανοίγει με <? και να τελειώνει (κλείνει) με ?>
- Το πιο κλασικό παράδειγμα σε οποιαδήποτε γλώσσα είναι η εμφάνιση της φράσης “Hello World” στην οθόνη του υπολογιστή
- Για την εμφάνιση του μηνύματος θα χρησιμοποιηθεί η εντολή echo. Το τέλος κάθε εντολής σε PHP υποδηλώνεται με το σύμβολο “;” (semicolon)
- Στην PHP όλες οι λέξεις κλειδιά (δεσμευμένες λέξεις) με τις οποίες συνάσσουμε τις εντολές του προγράμματος, δεν διαφέρουν για πεζούς ή κεφαλαίους χαρακτήρες, άρα η εντολή echo και η εντολή ECHO είναι ισοδύναμες

Βασική Σύνταξη 2/3

```
<!DOCTYPE html>
<html>
<body>
<h1>My first PHP page</h1>
<?php
echo "Hello World!";
?>
</body>
</html>
```

Εικόνα 2

Βασική Σύνταξη 2/3

- Παρατηρήσεις:
  - η χρήση των διπλών ή των μονών εισαγωγικών παίζει ρόλο και δίνει διαφορετική έξοδο όταν μέσα στο string της echo παρεμβάλλονται και άλλα στοιχεία εκτός από απλό κείμενο όπως για παράδειγμα μεταβλητές
  - Όπως και στην HTML έτσι και εδώ οι συνεχόμενοι κενοί χαρακτήρες αγνοούνται
  - Αν θέλουμε μέσα στο string της echo να εμφανίζονται ειδικοί χαρακτήρες τότε πριν από αυτούς πρέπει να υπάρχει το σύμβολο “\”

Σχόλια

- Ένα σχόλιο στην PHP είναι μια ή πολλές γραμμές που δεν εκτελούνται σαν μέρος του script
- Ο μοναδικός σκοπός τους είναι για να διαβαστούν από κάποιον που διαβάζει τον κώδικα ώστε να καταλάβει ποιος είναι ο σκοπός του ή για να δρα ως υπενθύμιση όταν γράφετε το συγκεκριμένο κομμάτι κώδικα και ποια είναι η λειτουργία του

```
<?php

// Αυτό είναι ένα σχόλιο που καταλαμβάνει μια σειρά

# Αυτό είναι επίσης ένα σχόλιο που καταλαμβάνει μια σειρά

/*
Αυτό είναι ένα σχόλιο
που μπορεί να καταλαμβάνει
παραπάνω από μια σειρά

στο PHP script
*/
```

Μεταβλητές και σταθερές 1/6

- Κανόνες για μεταβλητές της PHP:
  - Μια μεταβλητή πάντα ξεκινάει με τον χαρακτήρα “$” και ακολουθείται από το όνομα της μεταβλητής
  - Το όνομα μιας μεταβλητής πάντα ξεκινάει με ένα γράμμα ή τον χαρακτήρα “_”(underscore/κάτω παύλα)
  - Το όνομα μιας μεταβλητής δεν μπορεί να ξεκινάει με αριθμό
  - Μια μεταβλητή μπορεί να περιέχει στο όνομα της μόνο αλφαριθμητικούς χαρακτήρες και κάτω παύλες (underscores)
  - Το όνομα μιας μεταβλητής περιέχει διάκριση πεζών-κεφαλαίων, δηλαδή η μεταβλητή $car είναι διαφορετική από την μεταβλητή $CaR
  - Δεν χρειάζεται να δηλώσουμε μια μεταβλητή στην αρχή του script. Η δημιουργία και ο τύπος της γίνεται την ώρα της κλήσης της στον κώδικα

Μεταβλητές και σταθερές 2/6

- Οι μεταβλητές στην PHP μπορούν να αποθηκεύσουν διαφορετικούς τύπους τιμών:
  - String
  - Integer
  - Float (floating point numbers - also called double)
  - Boolean
  - Array
  - Object
  - NULL
  - Resource

Μεταβλητές και σταθερές 3/6

- Για τις πράξεις μεταξύ μεταβλητών (αριθμητικών ή τύπου string) χρησιμοποιούμε τελεστές:
  - Αριθμητικοί τελεστές
  - Τελεστές Ανάθεσης
  - Τελεστές Σύγκρισης
  - Τελεστές Αύξησης/Μείωσης
  - Τελεστές Λογικής
  - Τελεστές Αλφαριθμητικών
  - Τελεστές Πινάκων

Μεταβλητές και σταθερές 4/6

- Η εμβέλεια μιας μεταβλητής ορίζεται από το σημείο που δημιουργείται και ορίζει το μέρος του script που μπορεί αυτή η μεταβλητή να χρησιμοποιηθεί
- Αν μια μεταβλητή έχει δημιουργηθεί εκτός μιας συνάρτησης (function), η εμβέλειά της θεωρείται καθολική (global), οπότε δεν μπορεί να χρησιμοποιηθεί μέσα στην συνάρτηση
- Αντίστοιχα αν μια μεταβλητή δημιουργηθεί μέσα σε μια συνάρτηση, η εμβέλειά της θεωρείται τοπική (local), τότε αν χρησιμοποιηθεί έξω από αυτήν θα εμφανιστεί μήνυμα λάθους
- Για να καλέσουμε την τιμή μιας καθολικής μεταβλητής μέσα σε μια συνάρτηση θα πρέπει πρώτα να προσθέσουμε κατά τη δημιουργία της τη λέξη κλειδί global

Μεταβλητές και σταθερές 5/6

- Κανονικά όταν μια συνάρτηση έχει εκτελεστεί οι τοπικές μεταβλητές διαγράφονται
- Υπάρχουν περιπτώσεις όμως που δεν είναι επιθυμητό μια μεταβλητή να διαγραφεί, τότε χρησιμοποιείται η λέξη κλειδί static
- Οι σταθερές λειτουργούν παρόμοια με τις μεταβλητές με τη διαφορά ότι δεν μπορούν να αλλάξουν τιμή κατά την εκτέλεση του php script
- Έτσι μπορούμε να πούμε ότι μια σταθερά είναι ένα αναγνωριστικό για μια συγκεκριμένη τιμή
- Οι σταθερές έχουν αυτόματα καθολική εμβέλεια από τη στιγμή που δημιουργούνται
- Για τη δημιουργία μιας σταθεράς χρησιμοποιείται η συνάρτηση define()
  - define(name, value, case-insensitive)

Μεταβλητές και σταθερές 6/6

```
<?php

$x = 5; // καθολική εμβέλεια

function Test1() {
 // χρησιμοποιώντας την μεταβλητή χ εδώ θα οδηγήσει σε μήνυμα λάθους
    echo "<p>Variable x inside function is: $x</p>";
}
Test1();
 // χρησιμοποιώντας την μεταβλητή χ εδώ θα λάβουμε την τιμή 5
echo "<p>Variable x outside function is: $x</p>";
?>
```

```
<?php
function Test2() {
    $x = 5; // τοπική εμβέλεια
    echo "<p>Variable x inside function is: $x</p>";
}
Test2();

 // χρησιμοποιώντας την μεταβλητή χ εδώ θα οδηγήσει σε μήνυμα λάθους
echo "<p>Variable x outside function is: $x</p>";
?>
```

Εικόνα 3

```
<?php
$x = 5;
$y = 10;

function Test3() {
    global $x, $y;
    $y = $x + $y;
}

Test3();
echo $y; // η έξοδος θα είναι 15
?>
```

Αριθμητικοί τελεστές

Τελεστής	Τελεστής	Όνομα	Παράδειγμα	Αποτέλεσμα
+	+	Πρόσθεση	$x + $y	Άθροισμα $x και $y
-	-	Αφαίρεση	$x - $y	Διαφορά $x και $y
*****	*	Πολλαπλασιασμός	$x * $y	Πολ/μός $x και $y
/	/	Διαίρεση	$x / $y	Διαίρεση $x και $y
%	%	Ακέραιο υπόλοιπο	$x % $y	Υπόλοιπο διαίρεσης $x και $y
******	**	Ύψωση σε δύναμη	$x ** $y	Το αποτέλεσμα του να υψώσουμε το $x στην $y'ωστή δύναμη

Εικόνα 4

Τελεστές ανάθεσης

Τελεστής	Όμοιος με..	Αποτέλεσμα
x=y	x=y	Στον αριστερό τελεστή ανατίθεται η τιμή της έκφρασης στα δεξιά
x+=y	x=x+y	Πρόσθεση
x-=y	x=x-y	Αφαίρεση
x*=y	x=x*y	Πολλαπλασιασμός
x/=y	x=x/y	Διαίρεση
x%=y	x=x%y	Ακέραιο υπόλοιπο

Εικόνα 5

Τελεστές Σύγκρισης

Τελεστής	Όνομα	Παράδειγμα	Αποτέλεσμα
==	Ισότητα	$x==$y	Αληθής όταν το $x είναι ίσο με το $y
===	Ταυτόσημα	$x===$y	Αληθής όταν το $x είναι ίσο με το $y και είναι και ίδιου τύπου
!=	Ανισότητα	$x!=$y	Αληθής όταν το $x δεν είναι ίσο με το $y
<>	Ανισότητα	$x<>$y	Αληθής όταν το $x δεν είναι ίσο με το $y
!==	Μη ταυτόσημα	$x!==$y	Αληθής όταν το $x δεν είναι ίσο με το $y ή δεν είναι ίδιου τύπου
>	Μεγαλύτερο	$x>$y	Αληθής όταν το $x είναι μεγαλύτερο από το $y
<	Μικρότερο	$x<$y	Αληθής όταν το $x είναι μικρότερο από το $y
>=	Μεγαλύτερο ή ίσο	$x>=$y	Αληθής όταν το $x είναι μεγαλύτερο ή ίσο από το $y
<=	Μικρότερο ή ίσο	$x<=$y	Αληθής όταν το $x είναι μικρότερο ή ίσο από το $y

Εικόνα 6

Τελεστές Αύξησης/Μείωσης

Τελεστής	Όνομα	Περιγραφή
++$x	Αύξηση πριν	Αύξηση του $x κατά ένα, επιστροφή του $x
$x++	Αύξηση μετά	Επιστροφή του $x, αύξηση του $x κατά ένα
--$x	Μείωση πριν	Μείωση του $x κατά ένα, επιστροφή του $x
$x--	Μείωση μετά	Επιστροφή του $x, μείωση του $x κατά ένα

Εικόνα 7

Λογικοί Τελεστές

Τελεστής	Όνομα	Παράδειγμα	Αποτέλεσμα
And	Και	$x and $y	Αληθής όταν και το $x και το $y είναι αληθή
Or	Ή	$x or $y	Αληθής όταν είτε το $x ή το $y είναι αληθή
Xor	Xor	$x xor $y	Αληθής όταν είτε το $x ή το $y είναι αληθή, αλλά όχι μαζί
&&	Και	$x && $y	Αληθής όταν και το $x και το $y είναι αληθή
||	Ή	$x || $y	Αληθής όταν είτε το $x ή το $y είναι αληθή
!	Όχι	!$x	Αληθής όταν το $x δεν είναι αληθές

Εικόνα 8

Τελεστές Αλφαριθμητικών

Τελεστής	Όνομα	Παράδειγμα	Αποτέλεσμα
.	Συνένωση	$txt1.$txt2	Συνένωση του $txt1 με το $txt2
.=	Προσάρτηση	$txt1.=$txt2	Προσάρτηση του $txt2 στο $txt1

Εικόνα 9

Τελεστές Πινάκων

Τελεστής	Όνομα	Παράδειγμα	Αποτέλεσμα
+	Ένωση	$x + $y	Ένωση $x και $y
==	Ισότητα	$x==$y	Επιστρέφει αληθής όταν το $x και το $y έχουνε τα ίδια ζευγάρια κλειδιών/τιμών
===	Ταυτόσημα	$x===$y	Επιστρέφει αληθής όταν το $x και το $y έχουνε τα ίδια ζευγάρια κλειδιών/τιμών, με την ίδια σειρά και ίδιους τύπους
!=	Ανισότητα	$x!=$y	Επιστρέφει αληθής όταν το $x δεν είναι ίσο με το $y
<>	Ανισότητα	$x<>$y	Επιστρέφει αληθής όταν το $x δεν είναι ίσο με το $y
!==	Μη ταυτόσημα	$x!==$y	Επιστρέφει αληθής όταν το $x δεν είναι ταυτόσημο με το $y

Εικόνα 10

Συναρτήσεις για δεδομένα τύπου string 1/7

- Η συνάρτηση strlen();
  - Η συνάρτηση της PHP strlen(); επιστρέφει τον αριθμό χαρακτήρων μιας συμβολοσειράς ή μιας μεταβλητής που την εμπεριέχει
  - Παράδειγμα:

```
<?php
echo strlen("Hello world!"); // Η έξοδος θα είναι 12
?>
<?php
$teststring = “Hello there!”;
echo strlen($teststring); // Η έξοδος θα είναι 11
?>
```

Συναρτήσεις για δεδομένα τύπου string 2/7

- Η συνάρτηση str_word_count();
  - Επιστρέφει τον αριθμό λέξεων μίας συμβολοσειράς ή μιας μεταβλητής που την εμπεριέχει
  - Παράδειγμα:

```
<?php
echo str_word_count ("Hello world!"); // Η έξοδος θα είναι 2
?>


<?php
$teststring = “Hello there!”;
echo str_word_count ($teststring); // Η έξοδος θα είναι 2
?>
```

Συναρτήσεις για δεδομένα τύπου string 3/7

- Η συνάρτηση strrev();
  - Αντιστρέφει μια συμβολοσειρά ή μια μεταβλητή που την εμπεριέχει
  - Παράδειγμα:

```
<?php
echo strrev("Hello world!"); // Η έξοδος θα είναι !dlrow olleH
?>


<?php
$teststring = “Hello there!”;
echo strrev($teststring); // Η έξοδος θα είναι !ereht olleH
?>
```

Συναρτήσεις για δεδομένα τύπου string 4/7

- Η συνάρτηση strpos();
  - Ψάχνει για συγκεκριμένο κείμενο μέσα σε μια συμβολοσειρά και αν το βρει επιστρέφει την θέση του χαρακτήρα όπου βρέθηκε το κείμενο για πρώτη φορά, αλλιώς επιστρέφει την τιμή FALSE
  - Παράδειγμα:

```
<?php
echo strpos("Hello world!", "world"); // Η έξοδος θα είναι 6
?>
```

```
<?php
$teststring = “Hello there!”;
$string_to_find = “ there”;
echo strpos($teststring, $string_to_find);
// Η έξοδος θα είναι 6
?>
```

Συναρτήσεις για δεδομένα τύπου string 5/7

- Μεταβλητές τύπου Array
  - Μερικές φορές χρειάζεται να αποθηκευτούν παραπάνω από μια τιμές σε μια μεταβλητή.
  - Σε αυτήν την περίπτωση πρέπει να χρησιμοποιηθεί ένας πίνακας με την χρήση μιας μεταβλητής τύπου array
  - Για τη δήλωση μιας μεταβλητής τύπου array υπάρχουν δυο τρόποι:

Συναρτήσεις για δεδομένα τύπου string 6/7

- Μεταβλητές τύπου Array (συν.)

```
<?php
$fruit = array("Apple", "Pear", "Banana");
?>
```

```
<?php
$fruit[0] = "Apple";
$fruit[1] = "Pear";
$fruit[2] = "Banana";
?>
```

Συναρτήσεις για δεδομένα τύπου string 7/7

- Για την προσπέλαση (διάσχιση) ενός array και για την εκτύπωση των τιμών μπορεί να χρησιμοποιηθεί μια δομή επανάληψης

```
<?php
$fruit = array("Apple", "Pear", "Banana");
$length_of_array = count($fruit);

for($x = 0; $x < $length_of_array; $x++) {
    echo $fruit[$x];
    echo "<br>";
}
?>
```

Δομές επανάληψης 1/3

- Ο βρόχος do…while
  - Πάντα εκτελεί μια φορά το μπλοκ κώδικα που εμπεριέχει και μετά ελέγχει την συνθήκη που τέθηκε
  - Αν η συνθήκη είναι αληθής τότε ο κώδικας που εμπεριέχεται θα συνεχίσει να εκτελείται μέχρι η συνθήκη να είναι ψευδής.
  - Παράδειγμα:

```
<?php

$x = 1;

do {
   echo $x;
   $x++;
} while ($x <= 5);

// Η έξοδος θα είναι 1, 2, 3, 4, 5

?>
```

Δομές επανάληψης 2/3

- Ο βρόχος for
  - Χρησιμοποιείται όταν είναι γνωστό εκ των προτέρων το πόσες φορές θα πρέπει να εκτελεστεί ένα κομμάτι κώδικα
  - Παράδειγμα :

```
<?php
for ($x = 0; $x <= 10; $x++) {
    echo "The number is: $x <br>";

// Η έξοδος θα είναι από το 0 μέχρι και το 10
}
?>
```

Δομές επανάληψης 3/3

- Ο βρόχος foreach
  - Χρησιμοποιείται μόνο σε μεταβλητές τύπου array και χρησιμοποιείται για να προσπελάσει ζευγάρια θέσης/τιμής
  - Στο παρακάτω παράδειγμα επιδεικνύεται πως με έναν βρόχο foreach εκτυπώνονται οι τιμές της μεταβλητής τύπου array ”fruit”
  - Παράδειγμα:

```
<?php
$fruit = array("Apple", "Pear", "Banana");


foreach ($fruit as $value) {
   echo "$value <br>";

// Η έξοδος θα είναι Apple, Pear, Banana
}
?>
```

Δομές επιλογής 1/5

- Η δομή ελέγχου if και οι παραλλαγές της
  - Η δομή ελέγχου if στην πιο απλή μορφή της χρησιμοποιείται για τον έλεγχο αν ένα τμήμα κώδικα θα εκτελεστεί σύμφωνα με μια συνθήκη
  - Εναλλακτικά με την παραλλαγή if…else ορίζεται μια συνθήκη και τα τμήματα κώδικα που θα εκτελεστούν ανάλογα με το αν αυτή η συνθήκη είναι αληθής ή ψευδής
  - Η τελευταία παραλλαγή είναι η if…elseif…else , που χρησιμοποιείται όταν πέρα από την πρώτη συνθήκη πρέπει να ελεγχθεί άλλη μια (ή και παραπάνω καθώς δεν υπάρχει περιορισμός στο πόσες συνθήκες elseif μπορούν να παρεμβληθούν) συνθήκη πριν την εκτέλεση της else που σημαίνει ότι όλες οι προηγούμενες συνθήκες ήταν ψευδείς

Δομές επιλογής 2/5

```
<?php
$time_hour = date("H");

if ($time_hour < "20")
 {
    echo "καλημέρα!";
/* Το κομμάτι κώδικα θα εκτελεστεί μόνο αν η ώρα του υπολογιστή είναι πριν τις 8 το βράδυ
*/
}
?>


<?php
$ time_hour = date("H");

if ($time_hour < "20")
 {
    echo "Καλημέρα!";
/* Το κομμάτι κώδικα θα εκτελεστεί μόνο αν η ώρα του υπολογιστή είναι πριν τις 8 το βράδυ
*/
} else
 {
    echo "Καληνύχτα!";
/* Το κομμάτι κώδικα θα εκτελεστεί αν η ώρα του υπολογιστή είναι μετά τις 8 το βράδυ */
}
?>
```

Εικόνα 11

Δομές επιλογής 3/5

```
<?php
$ time_hour = date("H");

if ($time_hour < "10")
 {
    echo "Καλημέρα!";
/* Το κομμάτι κώδικα θα εκτελεστεί μόνο αν η ώρα του υπολογιστή είναι πριν τις 10 το πρωί
*/
} elseif ($time_hour < "20")
{
    echo "Καλό μεσημέρι/απόγευμα!";
/* Το κομμάτι κώδικα θα εκτελεστεί μόνο αν η ώρα του υπολογιστή είναι πριν τις 8 το βράδυ
και η πρώτη συνθήκη είναι αναληθής άρα είναι μετά τις 10 το πρωί */
} else
 {
    echo "Καληνύχτα!";
/* Το κομμάτι κώδικα θα εκτελεστεί αν η ώρα του υπολογιστή είναι μετά τις 8 το βράδυ */
}
?>
```

Εικόνα 12

Δομές επιλογής 4/5

- Η δομή επιλογής switch
  - Χρησιμοποιείται όταν χρειάζεται να εκτελεστούν διαφορετικά τμήματα κώδικα ανάλογα με το αποτέλεσμα μιας συνθήκης (παρόμοια αλλά όχι ίδια με μια δομή if...elseif…else).
  - Στην πράξη υπάρχει μια συνθήκη (συνήθως μια μεταβλητή) η οποία αφού επιστρέψει μια τιμή , συγκρίνεται με τις τιμές για κάθε περίπτωση (case)
  - Αν βρεθεί κάποια ισότητα τότε εκτελείται το αντίστοιχο τμήμα κώδικα και η δομή σταματάει να εκτελείται χωρίς να προχωρήσει στην επόμενη σύγκριση (αυτή είναι η διαφορά με την δομή if…elseif…else)
  - Σε περίπτωση που καμία από τις περιπτώσεις δεν ταιριάζει τότε εκτελείται το τμήμα κώδικα που αντιστοιχεί στην περίπτωση default. Στο παρακάτω παράδειγμα παρουσιάζεται η σύνταξη της δομής switch

Δομές επιλογής 5/5

```
<?php
$best_fruit = "Apple";

switch ($best_fruit) {
   case "Apple":
      echo "Το καλύτερο φρούτο είναι το μήλο!";
      break;
   case "Pear":
      echo "Το καλύτερο φρούτο είναι το αχλάδι!";
      break;
   case "Banana":
      echo "Το καλύτερο φρούτο είναι η μπανάνα!";
      break;
   default:
echo "Το καλύτερο φρούτο δεν είναι ούτε το μήλο ούτε το αχλάδι ούτε και η
μπανάνα!";
}
```

Οι δηλώσεις include και require 1/3

- Οι δηλώσεις include και require αντιγράφουν όλο τον κώδικα που υπάρχει σε ένα συγκεκριμένο αρχείο και τον παραθέτουν στο αρχείο που χρησιμοποιεί την δήλωση
- Αυτό είναι πολύ χρήσιμο στην περίπτωση που χρειάζεται σε πολλαπλές σελίδες ενός ιστοτόπου να υπάρχει το ίδιο κομμάτι κώδικα
- Η διαφορά μεταξύ της include και της require είναι ότι όταν ένα αρχείο που δηλώνεται σε μια δήλωση include δεν μπορεί να εντοπιστεί από την PHP, ο κώδικας που ακολουθεί θα εκτελεστεί, ενώ στην περίπτωση της require με το που εκτελεστεί η εντολή και το αρχείο δεν εντοπιστεί, η εκτέλεση θα σταματήσει και θα εμφανιστεί μήνυμα λάθους

Οι δηλώσεις include και require 2/3

- Κανόνες για τις δηλώσεις include και require
  - όταν ένα αρχείο είναι σημαντικό και πρέπει οπωσδήποτε να υπάρχει για την ομαλή λειτουργία του script χρησιμοποιείται η δήλωση require
  - όταν ένα αρχείο δεν είναι σημαντικό και η συνέχεια του script δεν επηρεάζεται από την απουσία του χρησιμοποιείται η δήλωση include
- Αντί σε κάθε ξεχωριστή ιστοσελίδα να πρέπει να γραφεί ο ίδιος κώδικας, η χρήση της δήλωσης require θα κάνει πιο εύκολη και την τροποποίηση του χωρίς να χρειάζεται να γίνουν αλλαγές σε κάθε μια ιστοσελίδα ξεχωριστά

Οι δηλώσεις include και require 3/3

```
<?php
echo '<a href="/default.asp">Home</a> -
<a href="/html/default.asp">HTML Tutorial</a> -
<a href="/css/default.asp">CSS Tutorial</a> -
<a href="default.asp">PHP Tutorial</a>';
?>
```

Αρχείο “menu.php”

Εικόνα 13

Εικόνα 15


```
<html>
<body>

<div class="menu">
<?php require 'menu.php';?>
</div>

<h1>Καλώς ήρθατε στην προσωπική μου ιστοσελίδα!</h1>
<p>Κείμενο.</p>
<p>Κείμενο.</p>

</body>
</html>
```

Αρχείο “index.php”

Εικόνα 14

Sessions 1/3

- Τα sessions (σύνοδοι) είναι ένας τρόπος για να αποθηκεύονται προσωρινά μεταβλητές και δεδομένα για την αξιοποίηση τους σε πολλαπλές σελίδες ενός ιστοτόπου
- Τα δεδομένα αυτά από την στιγμή που θα δημιουργηθούν αποθηκεύονται προσωρινά στην μνήμη του υπολογιστή μέχρι ο χρήστης να κλείσει τον περιηγητή ιστσελίδων ή ο προγραμματιστής να δηλώσει το κλείσιμο του session σε κάποιο σημείο του κώδικα (πχ. κατά την αποσύνδεση του χρήστη, log out από ένα ιστότοπο.
- Ένα session ξεκινάει με την συνάρτηση session_start() και οι μεταβλητές υποδηλώνονται με την μορφή $_SESSION["variable_name"]
- Η συνάρτηση session_start() πρέπει πάντα να δηλώνεται στην αρχή του εγγράφου πριν από τα HTML tags
- Για την διαγραφή του session και των δεδομένων που εμπεριέχονται σε αυτό χρησιμοποιούνται οι συναρτήσεις session_unset() session_destroy()

Sessions 2/3

- Δημιουργία μιας ιστοσελίδας με όνομα “session_test1.php”

```
<?php
// Εκκίνηση του session
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Δήλωση μεταβλητών

$_SESSION["name"] = "James";
$_SESSION["occupation"] = "Programmer";
echo "Session variables are set.";
?>

</body>
</html>
```

Sessions 3/3

- Στην επόμενη ιστοσελίδα με όνομα “session_test2.php” χρησιμοποιείται η εντολή echo για την εμφάνιση των μεταβλητών που δηλώθηκαν στην session_test1

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>
<?php
/* Εμφάνιση των μεταβλητών του session που δηλώθηκαν στην προηγούμενη ιστοσελίδα */
echo "Your name is " . $_SESSION["name"] . ".<br>";
echo "Your occupation is " . $_SESSION["occupation"] . ".";
?>

</body>
</html>
```

Φόρμες και επικύρωση δεδομένων 1/14

- Οι φόρμες είναι ο τρόπος με τον οποίο γίνεται η συλλογή δεδομένων από τον χρήστη στην PHP.
- Μπορεί να είναι ένα απλό πλαίσιο κειμένου, μια σειρά από πλαίσια ελέγχου (checkboxes), κουμπιά επιλογής (radial buttons) και άλλα
- Σε ένα απλό παράδειγμα φόρμας ο χρήστης εισάγει το όνομα του σε ένα πλαίσιο κειμένου και αφού επιλέξει το κουμπί “submit” τα δεδομένα αποστέλλονται με την μέθοδο HTTP POST σε ένα άλλο αρχείο “sentdatatest.php” για επεξεργασία ή στην συγκεκριμένη περίπτωση απλή εμφάνιση

Φόρμες και επικύρωση δεδομένων 2/14

- Ο κώδικας του αρχείου με την φόρμα έχει ως εξής:

```
<html>
<body>
<form action=" sentdatatest.php" method="post">
Name: <input type="text" name="name"><br>
<input type="submit">
</form>
</body>
</html>
```

- Και ο κώδικας του αρχείου “sentdatatest.php” θα είναι:

```
<html>
<body>
Καλώς ήρθατε <br>Το όνομα σας είναι : <?php echo $_POST["name"]; ?><br>
</body>
</html>
```

Φόρμες και επικύρωση δεδομένων 3/14

- Υπάρχει άλλη μια μέθοδος με την οποία αποστέλλονται τα δεδομένα μεταξύ ιστοσελίδων που έχει σύνταξη ίδια με την $_POST αλλά:
  - Η μέθοδος $_GET μεταφέρει δεδομένα σαν παραμέτρους του URL σε αντίθεση με την $_POST που τα μεταφέρει μέσω της μεθόδου HTTP POST
  - Δεδομένα που έχουν σταλεί με την μέθοδο $_GET είναι ορατά σε όλους καθώς εμφανίζονται στο URL αλλά αυτό είναι θεμιτό σε μερικές περιπτώσεις που τα δεδομένα δεν είναι ευαίσθητα και υπάρχει περίπτωση να χρειάζεται η ιστοσελίδα με τα συγκεκριμένα δεδομένα να μπει σαν σελιδοδείκτης (π.χ. bookmark σε συγκεκριμένο αντικείμενο ενός e-shop)
  - Η μέθοδος $_GET έχει όριο 2000 χαρακτήρων για τις μεταβλητές/δεδομένα που αποστέλλονται μέσω αυτής
  - Η μέθοδος $_POST δεν έχει όριο χαρακτήρων και τα δεδομένα που αποστέλλονται μέσω αυτής είναι αόρατα προς άλλους οπότε είναι ιδανική για ευαίσθητα δεδομένα όπως κωδικοί

Φόρμες και επικύρωση δεδομένων 4/14

- Ένα σημαντικό κομμάτι όσον αφορά την ασφάλεια στις φόρμες και τα δεδομένα είναι η επικύρωση τους
- Όταν υπάρχει η ανάγκη για υποχρεωτικά πεδία σε φόρμες είναι χρήσιμη η καθολική μεταβλητή $_SERVER[“PHP_SELF”] η οποία αποστέλλει τα δεδομένα στην ιστοσελίδα που έγινε η υποβολή τους, επιτρέποντας τον έλεγχο για λάθη και για δεδομένα που ήταν υποχρεωτικά αλλά λείπουν
- Η $_SERVER[“PHP_SELF”] όμως από μόνη της αφήνει ένα σημαντικό κενό ασφάλειας στην μορφή του cross site scripting (XSS), γι’ αυτό είναι σημαντικό τα δεδομένα να περνάνε πρώτα από την συνάρτηση htmlspecialchars() η οποία μετατρέπει ειδικούς χαρακτήρες που χρησιμοποιούνται σε επιθέσεις XSS σε οντότητες HTML και τους καθιστά άχρηστους

Φόρμες και επικύρωση δεδομένων 5/14

- Έστω ότι υπάρχει η ιστοσελίδα “test.php” με την ακόλουθη φόρμα:
  - <form method="post" action="<?php echo $_SERVER["PHP_SELF"];?>">
- Αν ένας χρήστης εισάγει στο URL την ιστοσελίδα “www.webserver.com/test.php” ο παραπάνω κώδικας θα μεταφραστεί σε:
  - <form method="post" action="test.php">
- Έστω όμως ότι ο χρήστης εισάγει στο URL τα επόμενα:
  - http://www.webserver.com/test.php/%22%3E%3Cscript%3EMalicious_script%3C/script%3E
- Στην παραπάνω περίπτωση λόγω της $_SERVER[“PHP_SELF”] ο κώδικας θα μεταφραστεί σε:
  - <form method="post" action="test_form.php/"><script>malicious_script</script>

Φόρμες και επικύρωση δεδομένων 6/14

- Όπου malicious_script, μπορεί να είναι κώδικας βλαβερός προς άλλους χρήστες.
- Αν όμως χρησιμοποιηθεί η htmlspecialchars() όπως παρακάτω:
  - <form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
- Τότε ο κώδικας του URL μεταφράζεται σε:
  - <form method="post" action="test_form.php/"><script>(malicious_script)</script>">
- που δεν έχει ιδιαίτερο νόημα και η επίθεση αποτυγχάνει

Φόρμες και επικύρωση δεδομένων 7/14

- Παρακάτω αναδεικνύεται ο τρόπος με τον οποίο λειτουργεί η επικύρωση δεδομένων σε μια φόρμα με τρία στοιχεία, δυο πλαίσια κειμένου για όνομα και e-mail και μια επιλογή από 2 κουμπιά επιλογής για φύλο
  - Παράδειγμα:
  - Η φόρμα στην απλή της μορφή θα έχει ως εξής:

```
<h2>Παράδειγμα επικύρωσης δεδομένων</h2>
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
  Name: <input type="text" name="name" value">
  <br></br>
  E-mail: <input type="text" name="email" value">
   Φύλο:
  <input type="radio" name="gender" value="female">Γυναίκα
  <input type="radio" name="gender" value="male">Άνδρας
  <input type="submit" name="submit" value="Submit">
</form>
```

Φόρμες και επικύρωση δεδομένων 8/14

- Το επόμενο βήμα είναι η δημιουργία μιας συνάρτησης που θα λαμβάνει δεδομένα και:
  - χρησιμοποιεί την συνάρτηση htmlspecialchars() για να μειώσει το ρίσκο επίθεσης μέσω δεδομένων
  - χρησιμοποιεί την συνάρτηση stripslashes() για την αφαίρεση ανάστροφων καθέτων από τα δεδομένα
  - χρησιμοποιεί την συνάρτηση trim() για την αφαίρεση περιττών χαρακτήρων (ειδικούς χαρακτήρες, στηλοθέτες(tab) και άλλα)
- Η συνάρτηση θα έχει ως εξής:

Φόρμες και επικύρωση δεδομένων 9/14

```
<?php
// Θέτουμε τις μεταβλητές σε κενή κατάσταση
$name = $email = $gender = “”;

if ($_SERVER["REQUEST_METHOD"] == "POST") {
  $name = test_input($_POST["name"]);
  $email = test_input($_POST["email"]);
  $gender = test_input($_POST["gender"]);
}

function test_input($data) {
  $data = trim($data);
  $data = stripslashes($data);
  $data = htmlspecialchars($data);
  return $data;
}
?>
```

Εικόνα 16

Φόρμες και επικύρωση δεδομένων 10/14

- Έτσι τώρα κάθε φορά που ο χρήστης πατάει το κουμπί submit τα δεδομένα πρώτα περνάνε από τον έλεγχο της συνάρτησης test_input()
- Επίσης, το $_SERVER["REQUEST_METHOD"] == "POST" ενεργοποιείται αν ο χρήστης πατήσει το κουμπί submit αλλιώς το κομμάτι κώδικα αγνοείται και εμφανίζεται η κενή φόρμα (π.χ. την πρώτη φορά που ο χρήστης θα εισέλθει στην ιστοσελίδα)
- Το επόμενο βήμα είναι η μετατροπή των πεδίων που θεωρούνται σημαντικά σε υποχρεωτικά και η δημιουργία μηνυμάτων λάθους στην περίπτωση που ο χρήστης δεν εισάγει δεδομένα ή εισάγει λάθος δεδομένα σε ένα υποχρεωτικό πεδίο

Φόρμες και επικύρωση δεδομένων 11/14

- Στον παρακάτω κώδικα εισάγονται μερικές καινούριες μεταβλητές ($nameErr, $emailErr και $genderErr) που θα περιέχουν τα μηνύματα λάθους, και κανόνες για την περίπτωση που ο χρήστης αφήσει ένα πεδίο που είναι υποχρεωτικό κενό:

```
<?php
// Θέτουμε τις μεταβλητές σε κενή κατάσταση
$nameErr = $emailErr = $genderErr = "";
$name = $email = $gender = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
  if (empty($_POST["name"])) {
    $nameErr = "Name is required";
  } else {
    $name = test_input($_POST["name"]);
  }

  if (empty($_POST["email"])) {
    $emailErr = "Email is required";
  } else {
    $email = test_input($_POST["email"]);
  }
  if (empty($_POST["gender"])) {
    $genderErr = "Gender is required";
  } else {
    $gender = test_input($_POST["gender"]);
  }
}
?>
```

Εικόνα 17

Φόρμες και επικύρωση δεδομένων 12/14

- Επιπλέον, στο κομμάτι κώδικα της φόρμας προστίθενται τμήματα PHP κώδικα για την εμφάνιση του μηνύματος λάθους αν ο χρήστης δεν εισάγει δεδομένα:

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

Name: <input type="text" name="name">
<span class="error">* <?php echo $nameErr;?></span>
<br><br>
E-mail:
<input type="text" name="email">
<span class="error">* <?php echo $emailErr;?></span>
<br><br>
Gender:
<input type="radio" name="gender" value="female">Female
<input type="radio" name="gender" value="male">Male
<span class="error">* <?php echo $genderErr;?></span>
<br><br>
<input type="submit" name="submit" value="Submit">

</form>
```

Εικόνα 18

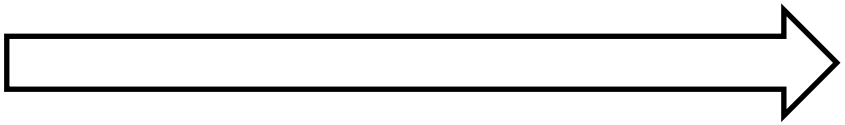
Φόρμες και επικύρωση δεδομένων 13/14

- Στο επόμενο βήμα πρέπει να ελεγχθεί αν το όνομα και το email είναι σωστά
- Το όνομα θα πρέπει να περιέχει μόνο χαρακτήρες και κενά και το email να έχει σωστή μορφή
- Για το όνομα θα χρησιμοποιηθεί μια κανονική έκφραση και η συνάρτηση της PHP “preg_match()” για τον έλεγχο των χαρακτήρων και για το email η συνάρτηση της PHP “filter_var()”

Φόρμες και επικύρωση δεδομένων 14/14

```
<?php
// Θέτουμε τις μεταβλητές σε κενή κατάσταση
$nameErr = $emailErr = $genderErr = "";
$name = $email = $gender = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
  if (empty($_POST["name"])) {
    $nameErr = "Name is required";
  } else {
    $name = test_input($_POST["name"]);
    // ελέγχουμε το όνομα να περιέχει μόνο χαρακτήρες και κενά
    if (!preg_match("/^[a-zA-Z ]*$/",$name)) {
      $nameErr = "Only letters and white space allowed";
    }
  }
```



Εικόνα 19

```
  if (empty($_POST["email"])) {
    $emailErr = "Email is required";
  } else {
    $email = test_input($_POST["email"]);
    // Ελέγχουμε αν το email έχει σωστή μορφή
    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
      $emailErr = "Invalid email format";
    }
  }
  if (empty($_POST["gender"])) {
    $genderErr = "Gender is required";
  } else {
    $gender = test_input($_POST["gender"]);
  }
}
?>
```

Μεταφόρτωση αρχείων στην PHP 1/6

- Ένα είδος φόρμας που έχει όμως αρκετή διαφορά με τα είδη που παρουσιάστηκαν παραπάνω είναι η φόρμα μεταφόρτωσης (uploading) αρχείων και ο κώδικας που είναι απαραίτητος για την διαχείρισή/αποθήκευσή τους.
- Στο παρακάτω απόσπασμα κώδικα παρουσιάζεται το html τμήμα της φόρμας μεταφόρτωσης αρχείων (στο συγκεκριμένο παράδειγμα μόνο εικόνες)

Μεταφόρτωση αρχείων στην PHP 2/6

```
<!DOCTYPE html>
<html>
<body>

<form action="upload.php" method="post"
enctype="multipart/form-data">
    Select image to upload:
    <input type="file" name="fileToUpload" id="fileToUpload">
    <input type="submit" value="Upload Image"
name="submit">
</form>

</body>
</html>
```

Μεταφόρτωση αρχείων στην PHP 3/6

- Σημαντικό είναι να σημειωθεί ότι αν δεν τεθεί το method=”post” και το enctype=”multipart/for-data” η μεταφόρτωση δεν λειτουργεί.
- Παρακάτω παρουσιάζεται το αρχείο “upload.php” που αναλαμβάνει την μεταφόρτωση του αρχείου που επιλέχθηκε από την παραπάνω φόρμα:

Μεταφόρτωση αρχείων στην PHP 4/6

```
<?php
$target_dir = "uploads/";
$target_file = $target_dir .
basename($_FILES["fileToUpload"]["name"]);
$uploadOk = 1;
$imageFileType =
pathinfo($target_file,PATHINFO_EXTENSION);
// Check if image file is a actual image or fake image
if(isset($_POST["submit"])) {
   $check =
getimagesize($_FILES["fileToUpload"]["tmp_name"]);
   if($check !== false) {
      echo "File is an image - " . $check["mime"] . ".";
      $uploadOk = 1;
   } else {
      echo "File is not an image.";
      $uploadOk = 0;
   }
}
// Check if file already exists
if (file_exists($target_file)) {
   echo "Sorry, file already exists.";
   $uploadOk = 0;
}
}
```

Εικόνα 20

```
// Check file size
if ($_FILES["fileToUpload"]["size"] > 5000000) {
   echo "Sorry, your file is too large.";
   $uploadOk = 0;// Allow certain file formats
if($imageFileType != "jpg" && $imageFileType != "png"
&& $imageFileType != "jpeg"
&& $imageFileType != "gif" ) {
   echo "Sorry, only JPG, JPEG, PNG & GIF files are
allowed.";
   $uploadOk = 0;
}
// Check if $uploadOk is set to 0 by an error
if ($uploadOk == 0) {
   echo "Sorry, your file was not uploaded.";
// if everything is ok, try to upload file
} else {
   if
(move_uploaded_file($_FILES["fileToUpload"]["tmp_nam
e"], $target_file)) {
      echo "The file ". basename(
$_FILES["fileToUpload"]["name"]). " has been uploaded.";
   } else {
      echo "Sorry, there was an error uploading your file.";
   }
}
?>
```

Μεταφόρτωση αρχείων στην PHP 5/6

- Η μεταβλητή $target_dir καθορίζει τον φάκελο στον οποίο θα αποθηκευτεί το αρχείο
- Ο φάκελος αυτός θα πρέπει, στην συγκεκριμένη περίπτωση, να είναι στο ίδιο μέρος που βρίσκεται το αρχείο “upload.php” και να έχει δημιουργηθεί από διαχειριστή γιατί η PHP δεν έχει δικαιώματα δημιουργίας φακέλων για λόγους ασφάλειας
- Η μεταβλητή $target_file περιέχει την διαδρομή και το όνομα του αρχείου προς μεταφόρτωση
- Η μεταβλητή $imageFileType περιέχει την επέκταση του αρχείου προς μεταφόρτωση και χρησιμοποιείται για να καθορίσουμε ότι το αρχείο είναι στην πραγματικότητα αρχείο εικόνας
- Η συνθήκη file_exists($target_file) ελέγχει αν υπάρχει αρχείο με ίδιο όνομα στον φάκελο με τις μεταφορτώσεις.
- Αν υπάρχει εμφανίζει μήνυμα λάθους

Μεταφόρτωση αρχείων στην PHP 6/6

- Η συνθήκη ($_FILES["fileToUpload"]["size"] > 5000000) ελέγχει για το αν το μέγεθος του αρχείου είναι μεγαλύτερο από περίπου 5Mb και αν είναι εμφανίζει μήνυμα λάθους
- Η επόμενη συνθήκη μας επιτρέπει να θέσουμε συγκεκριμένους τύπους αρχείων εικόνας που είναι επιτρεπτοί για μεταφόρτωση και αν η εικόνα που μεταφορτώνεται δεν ανήκει σε κάποιον από αυτούς τότε εμφανίζεται μήνυμα λάθους
- Η τελευταία συνθήκη περιέχει την μεταβλητή $uploadOk που χρησιμοποιείται σαν σημαία για το αν συνέβη κάποιο πρόβλημα σε όλους τους προηγούμενους ελέγχους
- Τέλος, αφού ξεπεράστηκαν όλοι οι έλεγχοι η παρακάτω εντολή:
  - move_uploaded_file($_FILES["fileToUpload"]["tmp_name"], $target_file
- μεταφέρει το αρχείο στον προκαθορισμένο φάκελο
- Πάλι, αν υπάρξει κάποιο πρόβλημα (π.χ. δεν υπάρχει αρκετός χώρος στον σκληρό δίσκο του server) εμφανίζεται μήνυμα λάθους, αλλιώς εμφανίζεται μήνυμα επιτυχούς μεταφόρτωσης

Συναρτήσεις

- Η χρήση συναρτήσεων στην PHP παρουσιάζει πολλές ομοιότητες με άλλες γλώσσες προγραμματισμού
- Ο ορισμός μίας συνάρτησης μπορεί να γίνει σε οποιοδήποτε σημείο του κώδικα - ακόμα και μετά την κλήση της
- Ο ορισμός μίας συνάρτησης γίνεται με την εξής σύνταξη:

```
function ονομα_συνάρτησης ($όρισμα1, $όρισμα2, ... , $όρισμαN)
{
        μπλοκ εντολών της συνάρτησης;
        return τιμή;
}
```

Συναρτήσεις

- Η εντολή επιστροφής return δεν είναι υποχρεωτική
- Μπορούμε δηλαδή να έχουμε συναρτήσεις που δεν επιστρέφουν τιμή
- Επίσης μπορούμε να έχουμε συναρτήσεις που δεν δέχονται ορίσματα. Παράδειγμα:

```
function add($a,$b)
{
        return $a+$b;
}

$result = add(10,20);
echo “Το αποτέλεσμα είναι $result”;
```

Συναρτήσεις

- Προσέξτε ότι κατά την κλήση μίας συνάρτησης δεν υπάρχει το σύμβολο $
- Επίσης όπως συμβαίνει και στις άλλες γλώσσες προγραμματισμού, τα ονόματα των ορισμάτων μίας συνάρτησης δεν έχουν σχέση με τυχόν μεταβλητές που βρίσκονται εκτός της συνάρτησης και έχουν το ίδιο όνομα
- Οι μεταβλητές που χρησιμοποιούνται μέσα στον ορισμό μίας συνάρτησης είναι τοπικές και ισχύουν μόνο μέσα στα όρια της συνάρτησης
- Όταν καλούμε μία συνάρτηση δηλώνοντας ως όρισμα την τιμή μίας μεταβλητής τότε η συνάρτηση χρησιμοποιεί ένα αντίγραφο της μεταβλητής χωρίς να επηρεάζει την αρχική τιμή

Συναρτήσεις

```
function add_1($num)          //η συνάρτηση add_1 αυξάνει την τιμή που δέχεται ως
{                             //όρισμα κατά 1
        $num = $num +1;
        return $num;
}

$a = 10;
echo "Η τιμή της μεταβλητής α είναι $a <br>";

$b = add_1($a);
echo "Το αποτέλεσμα της συνάρτησης είναι $b <br>";

echo "Ωστόσο η τιμή της μεταβλητής α δεν άλλαξε και παραμένει $a <br>";
```

Συναρτήσεις

- Αν θέλουμε να περάσουμε ως όρισμα την ίδια την μεταβλητή και όχι απλώς ένα αντίγραφο της τιμής της, τότε πρέπει μπροστά από το όνομα της μεταβλητής στην κλήση της συνάρτησης να βάλουμε το σύμβολο “&”
- Για παράδειγμα:

```
$b = add_1(&$a);
echo “Η τιμή της μεταβλητής α είναι πλέον $a”;
```

Συναρτήσεις

- Αν θέλουμε να χρησιμοποιήσουμε μέσα στην συνάρτηση μία μεταβλητή που ορίστηκε εκτός αυτής ή αν θέλουμε να χρησιμοποιήσουμε εκτός συνάρτησης μία μεταβλητή που ορίστηκε μέσα σε αυτήν τότε τη δηλώνουμε ως global

```
$a = 10;   //εκτός συνάρτησης
function add($b)
{
        global $a, $c;
        $c = 20;
        return $a + $b + $c;
}
$sum = add(30);
echo “Το άθροισμα και των τριών αριθμών είναι $sum <br>”;
echo “Ενώ η τιμή της μεταβλητής c είναι $c”;
```

Βιβλιογραφία

- Τσιάτσος Θρασύβουλος. Εκπαιδευτικά Περιβάλλοντα Διαδικτύου. Ηλεκτρονικά Ακαδημαϊκά Συγγράμματα και Βοηθήματα για Επιστήμες Μηχανικών και Πληροφορική, 2015.

Σημείωμα Χρήσης Έργων Τρίτων

- Το Έργο αυτό κάνει χρήση των ακόλουθων έργων:
- Εικονες 1, Τσιάτσος Θρασύβουλος. Εκπαιδευτικά Περιβάλλοντα Διαδικτύου. Ηλεκτρονικά Ακαδημαϊκά Συγγράμματα και Βοηθήματα για Επιστήμες Μηχανικών και Πληροφορική, 2015
- Εικόνα 2: Στιγμιότυπο από υπολογιστή, από το προσωπικό αρχείο του συγγραφέα
- Εικονες 3-20, Τσιάτσος Θρασύβουλος. Εκπαιδευτικά Περιβάλλοντα Διαδικτύου. Ηλεκτρονικά Ακαδημαϊκά Συγγράμματα και Βοηθήματα για Επιστήμες Μηχανικών και Πληροφορική, 2015

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης

Τέλος ενότητας

Επεξεργασία: <Στέργιος Τέγος>
Θεσσαλονίκη, <26/05/2015>

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Σημειώματα

Σημείωμα Ιστορικού Εκδόσεων Έργου

Το παρόν έργο αποτελεί την έκδοση 1.00.

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.